Linguistically Informed Language Generation: A Multi-faceted Approach

Dongyeop Kang

CMU-LTI-20-003

Language Technologies Institute
School of Computer Science
Carnegie Mellon University
5000 Forbes Ave., Pittsburgh, PA 15213
www.lti.cs.cmu.edu

Thesis Committee:
Eduard Hovy (Carnegie Mellon University, Chair)
Jeffrey Bigham (Carnegie Mellon University)
Alan W Black (Carnegie Mellon University)
Jason Weston (Facebook AI Research / New York University)
Dan Jurafsky (Stanford University)

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy
In Language and Information Technologies

Keywords: Natural language generation, Systemic Functional Linguistics, knowledge augmentation, structure imposition, stylistic variation, neural-symbolic integration, text planning, cross-style analysis

Abstract

Natural language generation (NLG) is a key component of many language technology applications such as dialogue systems, like Amazon's Alexa; question answering systems, like IBM Watson; automatic email replies, like Google's SmartReply; and story generation. NLG is the process of converting computer-internal semantic representations of content into the correct form of a human language, such as English or Korean, so that the semantics are accurately included. One might think that the only information an NLG system would need to produce is that contained explicitly in the utterance. However, there is a multitude of implicit information not explicitly obvious on the surface. For instance, many different surface sentences have the same meaning but still have slightly different surface outputs. Several kinds of parameters seem to be reflected in variations of an utterance: external knowledge, goals, interpersonal information, speaker-internal information, and more. In this work, we call each individual input parameter a *facet*. To generate appropriate and natural utterances as humans do, appropriate modeling of these facets is necessary, and the system needs to be effectively guided by these facets.

One of M. Halliday's linguistic theories, called Systemic Functional Linguistics (SFL), suggests that such parameters could be categorized into three meta-functions, where each contains separate types of information relevant to aural and written communication. We choose three facets of interest, one for each SFL meta-function, and repackaged them into three facet groups: *knowledge*, *structures*, and *styles*. The *knowledge* facet decides the basic semantics of the topic to be communicated, while the *structure* facet coherently arranges information guiding the structure of the (multi-sentence) communication. Finally, the *style* facet represents all the other additional kinds of information that direct the formulation of interpersonal communication.

We assume that the three facets are, more or less, individual, and they dynamically interact with each other instead of being a sequential process. One can develop a human-like NLG system that effectively reflects the three facets of communication and that simultaneously interact with each other, making a multifaceted system. In such systems, we believe that each facet of language has its own communicative goal, such that the *knowledge* facet is used to achieve factual goals, the *structure* facet is used to achieve coherence goals, and the *style* facet is used to achieve social goals.

To show the necessity and effectiveness of the multifaceted system, we have developed several computing methods for each facet from the following questions:

- Q1 "What" knowledge must be processed to make the model produce more factual text?
- Q2 "How" can the model compose multiple sentences coherently?
- Q3 How can the model produce stylistically appropriate output depending on "who" you are and "whom" you are talking to?

Acknowledgments

This thesis would not have been possible without the help and support of my advisor, thesis committee, mentors, friends, collaborators, faculty, and family.

I would like to express my deepest appreciation to my advisor, Professor Eduard Hovy. When I first started graduate school at LTI, I embarrassingly thought language was textual data for testing fancy machine learning algorithms. He opened my eyes to a new world of computational linguistics, including how to understand the semantics of language, why humans communicate via language, and how to generate human-like language. Among hundreds of inspirational pieces of his advice, my favorite is "deep thinking, not only deep learning." To me, he is the perfect example of a scholar. He thinks critically, is passionate about approaching problems, has the courage to delve into challenges, and is generous with his time. Although my time working with him resulted in solving only a tiny portion of the puzzles he set for me, I hope to continue relying on his wisdom to develop my career as a scientist. Professor Eduard Hovy, getting to know you has been the happiest and most precious part of my graduate study.

Additionally, I would like to thank Professor Jeffrey Bigham, Professor Alan W Black, Jason Weston, and Professor Dan Jurafsky for agreeing to be on my thesis committee. Your valuable advice and feedback on the proposal, defense, and final version of the thesis has made me realize how truly fortunate I was to have you all on my thesis committee.

Over the years of graduate school, I am grateful I could collaborate with amazing researchers at Microsoft Research, Allen Institute for AI, and Facebook AI Research. These experiences introduced me to many mentors: Patrick Pantel, Michael Gamon, Tushar Khot, Ashish Sabharwal, Peter Clark, Y-Lan Boureau, and Jason Weston. Your support and guidance helped broaden my perspective, both as it applies to research and the world in general. I would also like to thank my peers, including Madian Khabsa, Waleed Ammar, Bhavana Dalvi, Roy Schwartz, Anusha Balakrishnan, Pararth Shah, and Paul Crook.

I am extremely grateful to have made such amazing friends at CMU. Without your help, I would not have finished a single piece of work. I would particularly like to thank Hiroaki Hayashi and Dheeraj Rajagopal, who were always available whether I needed a random chat or to discuss research ideas. Whenever I felt down, you guys encouraged and motivated me again. I am also grateful to the Ed-visees members: Dheeraj Rajagopal, Varun Gangal, Naoki Otani, Evangelia Spiliopoulou, Hector Liu, Xuezhe Ma, Divyansh Kaushik, Qizhe Xie, Yohan Jo, and Maria Ryskina. I would also like to thank my friends, Hyeju Jang, Yohan Jo, Byungsoo Jeon, Kiryong Ha, Jay Lee, Seojin Bang, and Jinkyu Kim; you made me feel comfortable living in US. To my dear office mates, Volkan Cirik, Daniel Spokoyny, and Vidhisha Balachandran, thank you for sharing delicious snacks and providing helpful feedback. Yongjoon Kim and Junho Suh, my two oldest friends, thank you for treating me like a dumb high-school student. Your conversations helped me relax when I needed it the most.

I would also like to thank the faculty members, collaborators, and administrative staff of LTI who helped me, including Professor Robert Frederking, Professor Graham Neubig, Professor Yulia Tsvetkov, Professor Teruko Mitamura, Professor Eunsu Kang, Stacey Young, Salvador Medina, Michael Miller, Shirley Hayati, Paul Michel, Artidoro Pagnoni, Shruti Palaskar, and many others whose names I do not have space to list.

To my parents and sisters, thank you for your patience, endless love, and wisdom. It has been hard being apart for so long, but I sincerely appreciate everything you shared with me. I hope to soon fulfill my responsibilities as your son and brother. Finally, I would like to thank my wife, Taehee Jung. You made me realize what is most important in my life. Through the painful and hard times, you have kept me believing that going through the journey of life together will get us farther and happier than I ever could alone. Marrying you was the biggest achievement of my graduate life.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Natural Language Generation

Natural language generation (NLG) is the process of converting computer-internal semantic representations of content into the correct form of a language (e.g., English, Korean) so that the semantics are accurately included. NLG is often viewed as the opposite of natural-language understanding (NLU) or as a tightly coupled component with NLU in a large pipeline within a language-oriented Artificial Intelligence (AI) system. NLU systems understand an input sentence and disambiguate it to produce machine-readable forms, whereas NLG systems must make decisions about how to put the semantics into words with respect to situational context.

NLG is a key component of many language technology applications such as dialogue systems (e.g., Amazon Alexa, Google Assistant), question answering systems (e.g., IBM Watson, SQUAD (Rajpurkar et al., 2016)), automatic email replies (e.g., SmartReply (Kannan et al., 2016)), story generation (Schwartz et al., 2017), automated journalism (Graefe, 2016), personalized football reports (e.g., Yahoo! Fantasy Football), news summaries (Hovy et al., 1999), and more.

A sophisticated NLG system requires stages of planning and merging of information to produce more natural text. The stages in traditional NLG systems, as proposed by Reiter and Dale (1997), include content determination, document structuring, aggregation, lexical choice, referring expression generation, and realization. In contrast to this traditional pipeline, recent NLG systems are developed in end-to-end fashion, without separate stages. End-to-end systems have shown success in image captioning and machine translation tasks.

One might think that the only information an NLG system would need to produce is that contained explicitly in the utterance. However, there is a multitude of implicit information not explicitly obvious on the surface. For instance, many different surface sentences have the same meaning but still have slightly different surface outputs. Therefore, a deeper understanding of the implicit information that places it in context, detects intents behind semantics, and structures it into the correct form is crucial to a well-functioning NLG system.

1.1.1 Case Studies

To better understand the difference between human-written language and machine-written language (or machine language understanding), we conduct three case studies as follows.

Case #1 is presented below in Table 1.1, which shows a set of two tweets (top) and a set of three emails (bottom); the times within each set concern the same topic. Within each set, the examples all seem to have the same meaning, but they nonetheless have slightly different connotations. We first measure the semantic similarity between two sentences by using calculating the cosine similarity between two sentences' vector representations as measured by sentence-BERT, the state-of-the-art sentence representation model (Reimers and Gurevych, 2019). If we compare the two tweets in the first set, the best language understanding model says they are semantically identical, with 90% similarity score. In fact, however, depending on the relationship between the speaker and the subject of the tweet, the two tweets have totally different meanings; the former is sarcastic, because of the adversarial relationship between Hilary Clinton and Donald Trump (the two major opposing candidates for the 2016 United States presidential election), whereas the latter is admiring.

Case #1

Two tweets: Hilary Clinton: "*Great* year Trump, *great* year." Bob: "It was a great year, Trump!"
Three emails: **F**ormal (written to an unknown audience): "I am applying for the receptionist position advertised in the local paper. I am an excellent candidate for the job because of my significant secretarial experience, good language skills, and sense of organization." **S**emi-formal (written to an individual known to the writer): "I am applying for the receptionist position that is currently open in the company. As you are aware, I have worked as a temporary employee with your company in this position before. As a result, I not only have experience and knowledge of this position, but also already understand the company's needs and requirements for this job." **I**nformal (incorrect): "Hi! I read in the paper that ya'll were looking for a receptionist. I think that I am good for that job because I've done stuff like it in the past, am good with words, and am incredibly well organized."

Table 1.1: Top: two tweets about **President Donald Trump's 2019 State of the Union address report** (semantic similarity between the two tweets: **0.908**). Bottom: three emails for **job searching** (semantic similarities between formal/semi-formal: **0.729**, formal-informal: **0.909**, semi-formal/informal: **0.909**).

In the second set of examples, a group of three written emails related to job searching (Helder, 2011), the level of formality of the emails varies from formal to semi-formal to informal. Depending on the person to whom you write an email (to an unknown audience or an audience

known to you), the lexical and structural choices differ for each type of email. For the purpose of job searching, the third email is too informal to be appropriate. The program sentence-BERT's understanding of their semantics is again very high: 90.9% between formal and informal text.

But without understanding the underlying interpersonal context such as the relationship between the speaker and the listener, the level of formality of text, the speaker's relationship to the topics, and more, a machine cannot produce or understand human-like language.

Case #2

"Every natural text is written in some style. The style is formed by a complex combination of different stylistic factors, including formality markers, emotions, metaphors, etc. Some factors implicitly reflect the author's personality, while others are explicitly controlled by the author's choices in order to achieve some personal or social goal. One cannot form a complete understanding of a text and its author without considering these factors. The factors combine and co-vary in complex ways to form styles."	"Every natural text is written in some style. Unfortunately, the chosen style does not always make sense, even in texts with a relatively well-defined grammar. This is the issue that a definition of a dictionary fails to address. We should try to find an appropriate dictionary of natural text and then associate it with the written language. This is perhaps a slightly more involved project than we intend to make it out to be, so let's first define what we intend to do. A natural text is a collage of the words and sounds that make up a language."
GPT-LM Perplexity: 73.8 GLTR Top-k count(%): 64 22 11 2 Local Coherence (log P): -0.6083	GPT2-LM Perplexity: 30.1 GLTR Top-k count: 78 14 7 0 Local Coherence (log P): -0.5726

Table 1.2: A paragraph written by a human (left) and by a machine, GPT2 (right), when given the same prompt text "Every natural text is written in some style." We provide three automatic measures of their textual coherence.

Case #2, which appears in Table 1.2 below, shows two paragraphs, one written by a human writer (left) and another by the state-of-the-art text generator GPT2 (Radford et al., 2019), both created when given the same textual prompt, "Every natural text is written in some style." The human-written text conveys a specific intent (i.e., the complexity of style and how style forms in language), followed by a coherently structured discourse (e.g., definition of styles, elaboration, counter-examples, etc). In contrast, the machine-written text makes sense as an English text but only at the surface level, and its central message is hard to capture. In particular, the overall flow of text is not coherent at all: it first poses the issue of style, then proposes to use an appropriate dictionary of natural text, and finally defines natural text as a collage of words and sounds. Making text whose intent is clear and rendering its structure coherent are key elements in generating human-like multi-sentence text, while they are difficult to find from the machine-generated text.

Beside the qualitative analysis, we like to check how well existing automatic measures of textual coherence can detect such long-term coherence. We use three automatic measurements: *perplexity*, *top-k distribution*, and *local coherence*: In information theory, perplexity measures

how well a probability distribution or probability model predicts a sample. Low perplexity indicates that the probability distribution is good at predicting the sample. We use one of the largest probability models, GPT-LM, which was trained on billions of samples of text (Radford et al., 2018). Despite the poor discourse structure observed in our qualitative study, perplexity of the machine-generated text is much lower than in the human-generated text, indicating that how bad the perplexity of a language model is as a measure of long-term textual coherence.

Top-k distribution measures how each word prediction falls into different buckets of top-k prediction in the language model (i.e., k={10, 10^2, 10^3, 10^4}). We use the GLTR system (Gehrmann et al., 2019) with the same GPT2 language model (Radford et al., 2019). Gehrmann et al. (2019) show that machine-generated text has more skewed 10 or 10^2 distribution than human-generated text, indicating that the machine has made less diverse word choices than a human writer. As expected, in our two paragraphs in Table 1.2, the machine-generated text uses 78% of top-10 word choices, whereas the human text uses only 65%, showing the limitation of language modeling prediction as a coherence measure.

Finally, we use the entity-grid method (Lapata and Barzilay, 2005; Barzilay and Lapata, 2008), which measures the regularities (i.e., co-occurrence of entities over sentences) reflected in the distribution of entities over sentences. The method is based on the Centering theory (Grosz et al., 1995), whereby discourse segments in which successive utterances mention the same entities are *more coherent* than discourse segments in which multiple entities are discussed. The final probability is the log probability ($\log P$) of the entity distribution over sentences, the higher the better. Again, unfortunately, the local coherence score in the human-written text (-0.6083) is lower than the machine-written text (-0.5726), indicating the difficulty of capturing semantic coherence between texts.

In summary, none of the automatic metrics successfully measured the textual coherence between the human-written and machine-written text, while the differences in their textual coherence is clear in manual checking.

We often see some texts that are connected illogically or that includes false information, leading to generation of untruthful text. Untruthful text results when pieces of missing pieces of information between texts or illogical connections between them.

Case #3, in Table 1.3, shows an open-ended conversation question-and-answer conversation (top) and then three examples with their factuality scores predicted by the state-of-the-art text reasoner, ROVER (Clark et al., 2020). The ROVER system is based on the pre-trained transformer language model (Devlin et al., 2019), fine-tuned with formal logic. In the first conversation, in order to correctly answer Alice's question as "blacktop," not "sand" the system needs to retrieve external commonsense knowledge, such as the idea that "roller skating" needs a "smooth" surface and that "blacktop" is a smooth surface, then the system needs to reason about the knowledge, and finally provide the answer as an English sentence (realization).

Case #3

The second example shows how brittle the ROVER (Clark et al., 2020) can be, even with simple cases. Given background text (underlined), the system makes wrong predictions (red ones) with high confidence scores. This occurs because such systems are yet not fully capable of handling the logical formalism of text with enough generalization.

Open-ended question-answering conversation: Alice: "What surface would be the best for roller skating?" Bob: "Blacktop, I guess." Carol: "Sand is good to roller skate."
Soft reasoning output with confidence scores predicted by ROVER: "Mary regrets that John does not know he is sick." ROVER predictions: "John is sick." True (conf. = 0.99) "John is aware of it." False (conf. = 0.99) "Mary knows John is sick." False (conf. = 0.99) "100 is greater than 10. 10 is greater than 1." ROVER prediction: "100 is less than 1." True (conf. = 0.56) "A is B. B is C." ROVER predictions: "A is C." False (conf. = 0.93)

Table 1.3: Top: an example of an open-ended question answering conversation. Bottom: soft reasoning output with confidence scores predicted by ROVER (Clark et al., 2020).

Developing more human-like language understanding or generation systems is a challenging but promising task. Imagine an NLG system that can debate a human about a certain topic, compose a narrative story or a novel, or write a critique of your manuscript. Such systems suggest the next steps toward developing artificial general intelligence (AGI) systems. However, our three case studies show the clear gap that exists between current machine learning and humans in understanding and generating language. A question arises: what elements are needed to bridge that gap?

1.1.2 Toward Human-like Language Generation: More Facets Needed

One might think that the only information an NLG system needs to generate language is the information contained explicitly in an utterance. However, as noted above, a multitude of implicit information is *not* explicitly obvious on the surface. For instance, many different surface sentences can have the same meaning (i.e., denotation) but still have slightly different surface outputs (i.e., connotations). To develop a system that can generate language as humans do, we need more information. How can the system know which connotations(s) are the right ones? How can the NLG system decide what *exactly* to say? What parameters are needed besides what is reflected explicitly on the surface?

Several kinds of parameters seem to be reflected in variations of an utterance: external knowl-

edge, goals, interpersonal information, speaker-internal information, and more. In this work, we call the individual input parameter a *facet*. To generate more appropriate and natural utterances as humans do, appropriate modeling of these facets is necessary, and the system needs to be effectively guided by the facets.

In Table 1.4, we first take a deeper look at what types of facets are necessary in human language, specifically a daily conversation between two people. The surface conversation between Alice and Bob is far more complex than it appears. For example, are they using formal language? I'm guessing they are not, because of "honey, I'm here!", indicating they are a couple. In order to build a NLG system that can speak like Bob, the system needs to know a great deal of other external information beyond the utterances made by Alice, as shown in Table 1.5.

Alice:	"Are you there?"
Bob:	"Honey! I'm here!"
Alice:	"I skipped lunch today"
Bob:	"Choose one: Korean, Japanese, or American."
Alice:	"Japanese!"
Bob:	"How about Chaya tonight?"
Alice:	"Sounds perfect!"
Bob:	"I will book it at 6, so see you there by then. Don't forget to bring your umb. It's raining outside now."

Table 1.4: An example of human-human conversation about dinner.

- relationship with the listener: couple
- closeness to the listener: very close
- mood: <be romantic>, <be friendly>
- topic: dinner
- goals: taking an initiative, deciding a restaurant to go
- intents: <GREETING>, <ASK>, <PROVIDE> <SUGGESTION>, <make, reservation, at Chaya>, <...>
- past experience with the speaker: <Alice, like, Sushi>
- commonsense knowledge: <X1, is, Alice>, <X1, skip, lunch> < ¬lunch, cause, X1, hungry>, <X2, is, Chaya>, <X2, is, Japanese Restaurant> <X2, serve, Sushi>
- action sequence: (hungry, book restaurant, go to restaurant, eat food, ¬hungry),
- weather: <raining>, <raining, need, umbrella>
- medium, device: written, mobile
- and more

Table 1.5: Various facets used in a conversation about dinner.

For instance, Table 1.5 lists some facets that need to be considered to generate Bob's responses. This list reveals that many facets are hidden behind utterances and how humans can appropriately use these facets when generating language. In fact, there are many other facets

used unconsciously or consciously in our daily conversations that are not listed here, such as location and time of the conversation.

Supervisor:	"Are you there?"
Bob:	"Yes, sir! I am here. How can I help you?"
Supervisor:	"I skipped lunch today"
Bob:	"Oh, I see. You must be starving now then :-) Do you have any specific restaurants to go in your mind? Otherwise, please let me know which type of restaurant you prefer to go like Korean, Japanese, Chinese, or American."
Supervisor:	"Japanese!"
Bob:	"Okay. How about a Japanese restaurant called Chaya? It serves descent sushi plates, if you are okay with. It is also in a walking distance."
Supervisor:	"Sounds perfect!"
Bob:	"Okay, then I will book the restaurant at 6 under my name. I will be there a few minutes early so please let me know if you want me to order some appetizers first. By the way, it is raining outside, so don't forget to bring your umbrella. Also, let me know if you need a ride. Then I will pick you up by 5:30 in your office."

Table 1.6: An example of human-human conversation about dinner when the relationship with the listener changes to the relationship between a subordinate and a **senior guest**.

If the value of any of the facets changes, the output text will be entirely different. For example, given the same facet values, except that the relationship with the listener is changed to a "senior guest" (Table 1.6), the appropriate form of output would be more polite, formal, and descriptive. But, how many types of facets are there? Is there any linguistic theory developed to support facet-based language generation?

1.1.3 Systemic Functional Linguistics (SFL) Theory

This work is based on the linguistic theories developed by *Michael Halliday (1978)*, rather than on rationalist theory by Noam Chomsky or behaviorist theory by Leonard Bloomfield. Halliday's sociolinguistic view of language focuses more on communicative and pragmatic uses of language, which provide the comprehensive categorization of the additional facets mentioned in our previous examples.

Halliday believed that language is a not a system of signs, but rather a network of systems for chosen meanings. Each choice point expresses small pieces of communication including social, interpersonal, topical, ordering, and other aspects. Each choice point is then controlled by one or more contextual/pragmatic parameters. Of course, some choices depend on others. Halliday calls these choice points "systems", and his theory connects them into a large decision network that reflects this underlying interdependence. Therefore, generating language is a systemic process of combining various meaning choices.

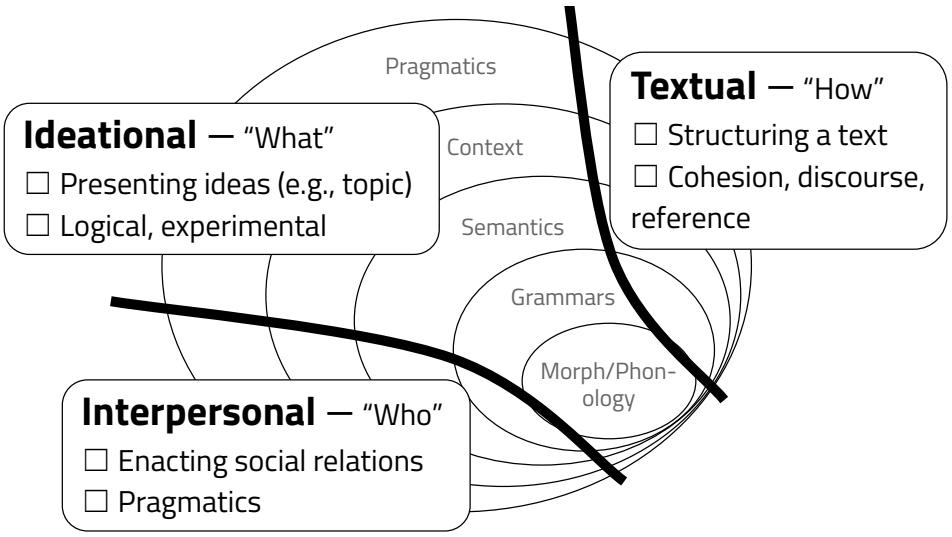


Figure 1.1: SFL meta-functions over the meaning spectrum of language. All languages involve three simultaneously generated metafunctions (Halliday, 2003a).

One of Halliday's theories, **systemic functional linguistics** (SFL) (Halliday, 1976, 2003a), states that language develops in response to three metafunctional needs: *ideational*, *interpersonal*, and *textual*. Figure 1.1 shows the three SFL functions over a spectrum of language meaning. Meaning can be multi-layered, including everything from morphology to, grammar to, semantics to, and context to pragmatics. The three SFL meta-functions exist over a spectrum of different layers of meaning. For example, the *textual* meta-function could play a functional role at the levels of grammar and pragmatics. Each meta-function is described as follows:

- The **ideational meta-function** allows us to construe experience in terms of what is going on around us and inside us. For example, it is to talk about experience, people and things, their actions, place, times, or circumstance in which events occur.
- The **textual meta-function** involves creating messages with which we can package our meanings in terms of what is new or given, and in terms of what the starting point for our message is. It embraces all the grammatical systems responsible for managing the flow of discourse. For example, the textual meta-function links complex ideas into coherent waves of information.
- The **interpersonal meta-function** involves interacting with the social world by negotiating social roles and attitudes. It represents the speaker's meaning potential as an intruder. This meta-function is used to enact social relationships, cooperate, form bonds, negotiate, ask for things, and more.

Halliday (2003b) believed that language displays functional complementarity over the three meta-functions. In other words, language has evolved under the human need to make meanings about the world around and inside us while at the same time allowing us to create and maintain our interpersonal relations. This thesis is found on top of his socio-linguistic perspective of language, particularly the functional interdependence of the three meta-functions.

One might be curious then whether SFL theory is just one of traditional theories or practi-

- **Ideational** meta-function
  - weather: <raining>, <raining, need, umbrella>
  - topic: dinner
  - experience with the speaker: <Alice, like, Sushi>
  - commonsense knowledge: <X1, is, Alice>, <X1, skip, lunch> < ¬lunch, cause, X1, hungry>
  - external knowledge: <X2, is, Chaya>, <X2, is, Japanese Restaurant> <X2, serve, Sushi>
  - medium: written
  - device: mobile
  - time: night
- **Textual** meta-function
  - goals: taking an initiative, deciding a restaurant to go for dinner
  - intents + ordering: <GREETING> - <ASK> - <PROVIDE_INFO> - <SUGGESTION> - <RESERVATION>
- **Interpersonal** meta-function
  - the listener's age: 31
  - the listener's gender: female
  - the relationship with the listener: `couple`
  - closeness with the listener: very close
  - mood: be romantic, be friendly,

Table 1.7: Various facets used in the human-human conversation under SFL meta-functions.

cally applicable to the modern NLG systems. Revisiting the previous dinner example and the relevant facets, as shown in Table 1.5, one can categorize the facets into the SFL meta-functions (Table 1.7). We observe that all facets related to "what" to say about such as topics, experience, knowledge people, or place fall into the *idealtional* meta-function, while facets about "how" to say such as goal setting, intent detection, and ordering fall into the *textual* meta-function. Finally, all other facets related to "who" is saying what and "whom" to say it to fall into the *interpersonal* meta-function. Here we describe some facets, but in fact there are many other facets that describe our daily communications. However, each meta-function is general enough by definition, to include various types of individual facet parameters, showing the generality and extent of SFL theory.

1.2 A Multifaceted Approach

With a theoretical basis of the SFL linguistic theory, this thesis proposes a conceptual framework called a *multifaceted* system, in order to build a human-like generation system. Due to the complexity of language, it is impossible to study every linguistic facet in one thesis. Instead, we choose an individual facet for each SFL meta-function: *knowledge* for the ideational meta-function, *structures* for the textual meta-function, and *styles* for the interpersonal meta-function. For each facet, we propose a cognitive architecture and develop prototypical systems in order to show how effectively the linguistic-facet-informed system produces more human-like output.

At the end, we suggest a conceptual framework for cascading multifaceted system in which the three cognitive modules are combined for future directions in NLG research.

Throughout the development of the multifaceted NLG system, we observe that *generating language is intended to achieve certain communicative goals*, such as the factual goal achieved by the knowledge facet, the coherence goal achieved by the structure facet, and the social goal achieved by the style facet. The following sub-sections will provide more detailed descriptions of each component in our proposal.

1.2.1 Facets

As shown in Table 1.7, humans use various parameters, either intentionally or unintentionally, when they communicate. We call an individual parameter a *facet*. Each specific facet can be grouped within higher-level facet groupings: circumstances (e.g., device, weather, age), knowledge types (e.g., experience, memory, database, ontology), cognitive functions (e.g., ordering, goal setting, reasoning, abstraction), or linguistic phenomena (e.g., metaphor, formality, exaggeration).

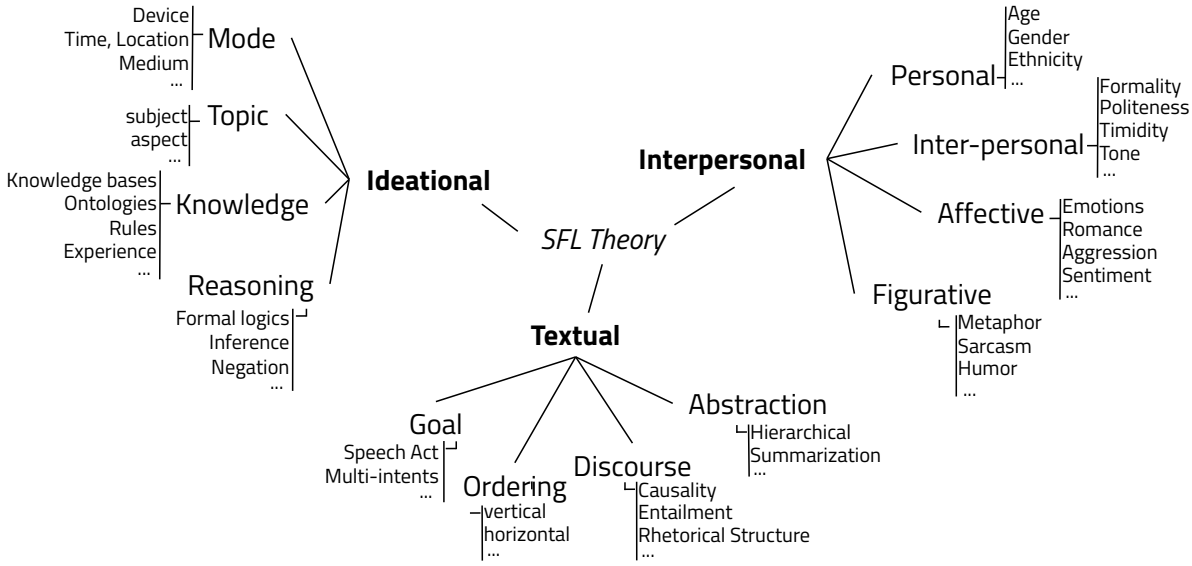


Figure 1.2: Facet ontology.

To the best of our knowledge, no formal classification system for facets exists, although some attempts have been made by Hovy (1987); Biber (1991), although these mostly focus on rhetorical goals (e.g., formality, tone) in the interpersonal meta-function. We first list the facets used in the thesis, then group them into sub-categories under each SFL meta-function, before building an ontology of facets (Figure 1.2).

In the ideational stage, NLG systems need to perform content selection and content retrieval by aggregating all topics, knowledge, and the inferred connections via reasoning, in order to make final selections of what content to talk about. Thus, one decides "what" to say: specific topics to discuss (e.g., the subject, the specific aspects of that subject), modes of the circumstance (e.g., device, time, location), knowledge about the topic required to bridge the gap between the

speaker's and listener's connotations (e.g., knowledge bases, commonsense, ontology, rules, experience, memory), and reasoning (e.g., formal logic, negation, temporal reasoning, numerical reasoning).

In the textual meta-function, "how" to link the content needs to be considered in different ways: goal setting (e.g., speech act, intents), content ordering, discourse modeling between parts of text (e.g., causal relations, entailment, rhetorical structures), and meaning abstraction (e.g., hierarchical modeling of the text). After that, the content is formed into a coherent text as a formal form of (English) language.

Finally and most rigorously, we categorize the interpersonal meta-function into four subgroups: personal facets (e.g., the speaker's age, gender, education level), inter-personal[1] facets (e.g., formality, politeness), affective facets (e.g., emotions, romance, aggression), and figurative facets (e.g., metaphor, sarcasm, humor). Unlike the previous categories on rhetorical goals (Hovy, 1987) or social constraints (Biber, 1991), our categorization of interpersoanl facets is much broader and comprehensive, ranging from personal and inter-personal facets to affective and figurative facets of language.

1.2.2 Facets-of-Interest: Knowledge, Structures, and Styles

It is intractable to propose a perfect ontology of every facet in language or to develop a single NLG system that effectively handles every facet. Moreover, some facets are very difficult to study due to the lack of annotated datasets or the complexity of building an appropriate system. For example, someone's moods (e.g., happiness, sadness) in the affective facets play an important role in human conversations. However, it requires a huge and sophisticated effort to collect the mood-annotated text and develop a generation system that produces mood-aware output in a conversation. Therefore, this thesis primarily focuses on some facets that are computationally implementational using underlying resources and algorithms. At the same time, we choose an individual facet of our interest for each SFL meta-function.

We choose three facets of interest, one for each SFL meta-function, and repackage into three facet groups: the *knowledge* facet (for the ideational meta-function), the *structures* facet (for the textual meta-function), and the *styles* facet (for the interpersonal meta-function). Figure 1.3 shows coverage of our repackaged facets in the facet ontology. We describe the rationale behind our selection of these three facets:

Knowledge

One has to represent the basic semantics of the topic that must be communicated; this we call the *knowledge* facet. It is often to lack of information between the context of communication and the knowledge that the speaker is aware of. To fill the gap between them, retrieval of necessary knowledge and reasoning over the knowledge are required to concretize the semantics of the topic.

[1]Note that inter-personal facet category is a subset of interpersonal metafunction.

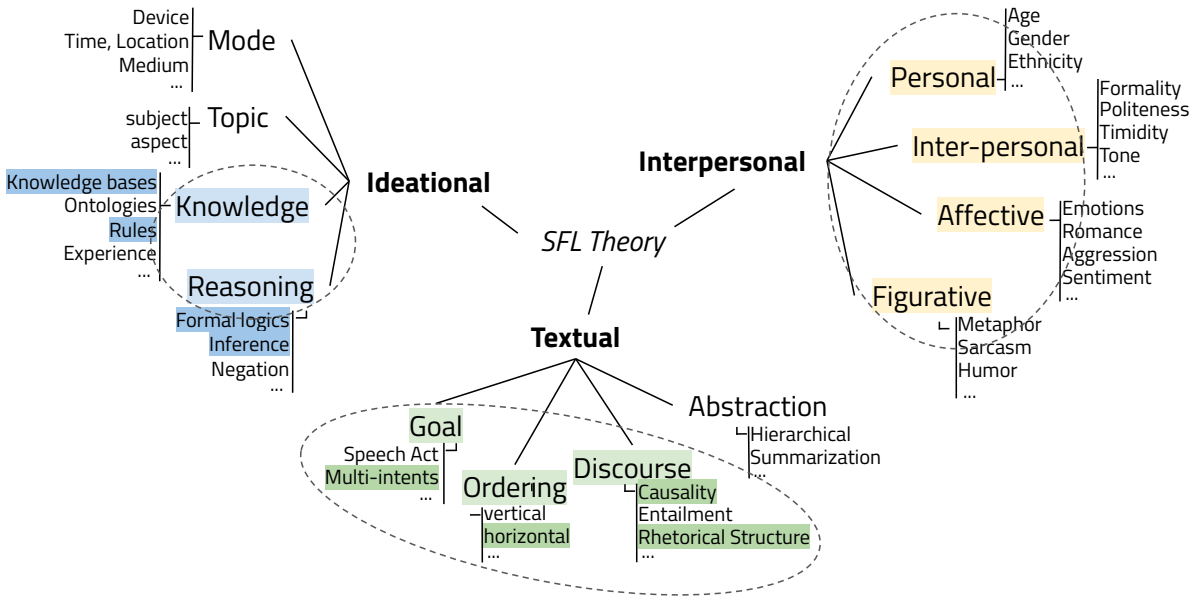


Figure 1.3: Facets of interests in this work: Knowledge facet, structure facet, and style facet.

Structures

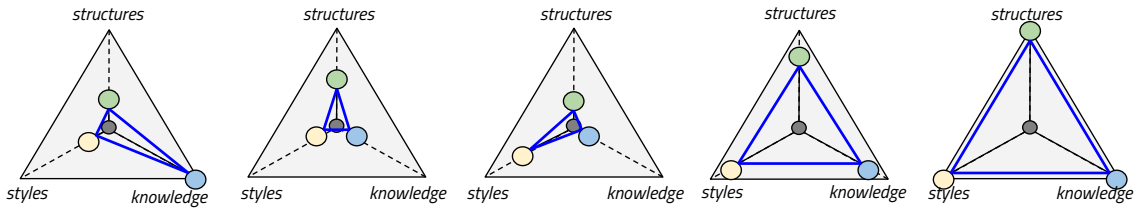
When the subject matter is decided, one has to represent the information guiding the structure of the (multi-sentence) communication in a coherent way, which is essentially a planning process; this we call *structures* facet. The structure here could be either setting a specific goal of generation, ordering the content, or connecting multiple sentences in a coherent way.

Styles

Lastly, one must represent all other, additional kinds of information guiding the interpersonal formulation of the communication; this we group into one large heterogeneous group we call *styles*. Instead of selecting a specific facet type (e.g., formality) and focusing closely on its textual variation, this thesis aims to provide a more comprehensive study of how different styles are dependent on each other and how they are combined.

1.2.3 NLG Applications with the Three Facets

To better understand how practical our choice of the three facets is, we analyze some NLG tasks using the three facet dimensions and see how much of each facet is required for each task (Figure 1.4). Some tasks such as question answering (QA), abstractive summarization, document-level machine translation, and style transfer, require a single facet, whereas others such as story generation and goal-oriented dialogue require multiple facets at the same time, making the tasks more challenging. As the number of facets needed increase, the problem of task becomes difficult to define so appropriate collection of the multi-faceted dataset becomes the real challenge and human evaluation is indispensable.



	QA	Summ. / Data2Text	Doc-level MT	StyleTransfer	StoryGen	Goal dialogue
Facets needed	Knowledge	Structures	Structures	Styles	Structures, Knowledge, Styles	Structures, Knowledge, Styles
Datasets	SQUAD (Rajpurkar et al., 2016)	CNNDM (See et al., 2017)	WMT (ACL, 2019)	SST (Socher et al., 2013)	Paragraphs	NegoDial (Lewis et al., 2017)
Evaluation	Auto (F1)	Auto (RBM)	Auto (RBM)	Auto (RBM), Human	Auto (RBM), Human	Auto (RBM), Goal, Human
Content given (Context ? Target)	Full or Partial (C $\supset$ T)	Full or Partial (C $\supset$ T / C $\subset$ T)	Full (C' $\sim$ T)	Full (C $\sim$ T)	None (C $\perp$ T)	None (C $\perp$ T)
Multi-sentence	N	Y	Y	N	Y	Y
NLG Objectives	Retrieval + Reasoning	Abstraction + Realization / Realization	Translation + Realization	Translation + Realization	Content Planning+ Realization	Content Planning+ Goal+Realization

Figure 1.4: NLG applications over the three facet dimensions (top) and their properties (bottom). Doc-level MT and StoryGen mean document-level machine translation and story generation, respectively. RBM in the evaluation row refers to automatic evaluation metrics such as ROUGE, BLEU, or METEOR. Multi-sentence means whether the target text to be generated is multiple sentences or not. Some tasks (e.g., summarization) have full or partial context provided, while others (e.g., StoryGen) have no content given, requiring context creation or planning process. (Context $\{\supset,\subset,\sim,\perp\}$T Target) shows the relationship between Context and Target text: context is a super/sub/equal/independent set of target text. In Doc-level MT, the context is given but in a different language (C').

Every task requires a comprehensive reader to understand the context, which is natural language understanding (NLU) part. In addition to that, each task has its own objectives to learn in addition to basic surface-level realization. For example, QA needs reasoning (and knowledge retrieval for open-ended QA (Khot et al., 2018)), abstractive summarization needs meaning abstraction or compression, and machine translation needs translation objective.

NLG encodes the context first and then decodes its semantics to the target text. Some tasks such as summarization and data2text generation provide full or partial context so target text becomes a subset (C $\supset$ T) or superset (C $\subset$ T) of the context. When there is no overlap between the context and the target text (C $\perp$ T), the tasks such as document generation, story generation, and dialogue generation require additional content planning and creation to decide what content to say and how to place them. In particular, the goal-oriented dialogues require specific goals to achieve (e.g., whether to negotiate or not, whether to change someone's view or not, whether to

recommend an appropriate item or not) throughout the generation.

Note that we do not list many other recently proposed multi-faceted datasets such as conversational question answering (Reddy et al., 2019), style-controlled machine translation (Niu et al., 2018), and more. Such efforts to combine various facets into the single task are good steps toward building multi-faceted generation systems, but are beyond the scope of this work.

1.2.4 NLG as a Multifaceted System

We now introduce our linguistic view of NLG as a system within which multiple facets interact with each other. Can one develop a cognitive NLG system that effectively reflects the three facets of communication and makes them interact simultaneously, in the same way that these facets function within human speech?

Figure 1.5 (a) shows the traditional pipeline of NLG (Reiter and Dale, 1997) that mostly focused on determining what content to say (content determination), deciding how chunks of content should be grouped in a document and how to relate the groups to each other and in what order they should appear (document structuring), deciding the specific lexical choices and referring expressions and aggregating them (microplanner), and finally converting them into actual text (surface realization). However, the pipeline fails to consider other facets of communication, including certain stylistic or pragmatic aspects.

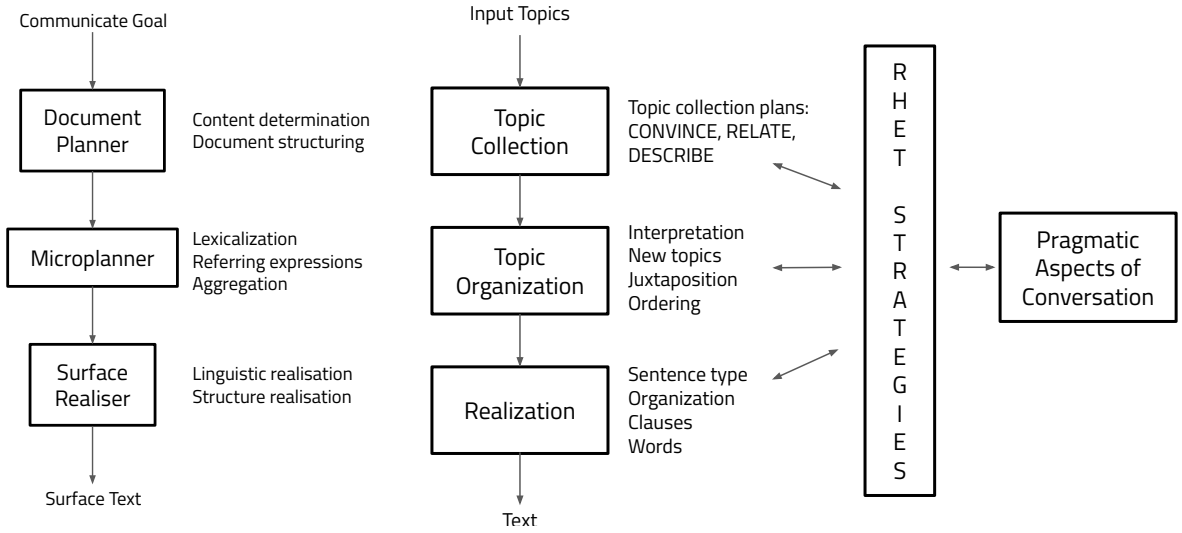
For the very first time, Hovy (1987) developed a system called PAULINE (Figure 1.5 (b)), which can produce stylistically appropriate text by explicitly setting rhetorical goals with pragmatic constraints such as formality, force, and timidity on top of the traditional pipeline. The rhetorical planner interacts with other modules; topic collection, topic organization, and realization, by moving back and forth and making pragmatically biased choices of text. From a cognitive perspective, it helps better understand speakers' goals and personal relationships in conversations. From a linguistic perspective, it shows how the information can be conveyed in different ways to different people, or even to the same person in different circumstances.

Motivated by the pragmatic aspects of the PAULINE system, we propose a more comprehensive NLG framework (Figure 1.5 (c)) that combines the three layered, cascading system where each layer corresponds to each facet – knowledge, structures, and styles – respectively and the goal setting module embraces all three layers simultaneously. Compared to PAULINE, whose selection of rhetorical goals is only triggered by the pragmatic constraints, we believe each facet layer has its own communication goal, which will be further discussed in the next sub-section. Each facet layer has different sub-functions as follows:

The first layer, *knowledge augmentation*, completes the basic semantics of the topic that must be communicated, by retrieving necessary knowledge and reasoning over the knowledge and the topic to decide "what" to say about the topic and to produce more trustworthy text.

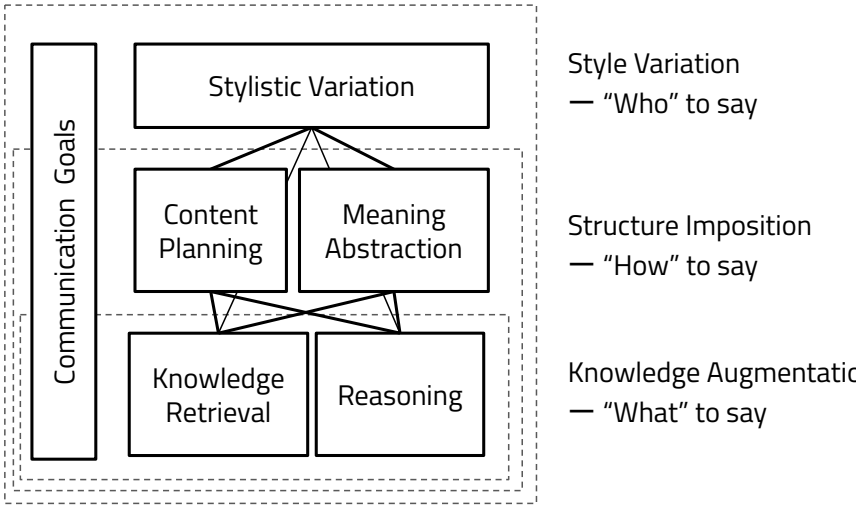
Once the content is decided, the second layer, *structure imposition*, organizes the structure of the content, in order to produce a coherent text. Our structure imposition layer combines the macro-planning and micro-planning found in the traditional pipeline: content planning for macro-level document structuring and meaning abstraction for micro-level semantic aggregation.

After deciding the content and structuring it, the third layer, *style control*, controls stylistic variation of the structured text with respect to interpersonal context and to achieve some social or personal goal (e.g., producing romantic text to make the listener happier).



(a) Traditional pipeline (Reiter and Dale, 1997)

(b) PAULINE (Hovy, 1987)



(c) A cascading multifaceted system

Figure 1.5: Comparison of three NLG systems.

Another major difference with the two prior systems is that all modules in each layer are connected to each other like a network, whereas prior systems restrict the connections to be either sequential or hierarchical. By giving more flexible connectivity across the layers, one can dynamically reverse the decisions made by previous facets, helping the interdependent facets become mutually reinforced. For example, one might choose a topic of conversation, then decide how to form the text, and then set the goal of RESPECTFUL in the style control layer to produce politer text that is more appropriate to senior listener. But, then the speaker might realize that the output text is too long to be appropriately read by the listener, and so decides to reduce the length of text either by reducing the amount of knowledge to convey (knowledge retrieval and reasoning)

or by structuring the text more concisely (planning and abstraction). Such dynamic interaction across the facets is the key element of modeling how multiple facets are simultaneously reflected in various linguistic variations.

1.2.5 Language generation is intended

Every natural language is intended, and each facet of language has its own communicative goal. Language is a tool for communication. Information is communicated between agents via a language (Parikh, 2001). The basic elements of the communication proposed by M. Halliday are *what* information to say, *how* to present the information, and *who* is the speaker and/or listener. One may be then curious about *why* humans communicate.

Every natural language is used for certain purposes. For example, even simple greetings like "It's nice to meet you" have their own goals, to make their presence known to each other, to show attention to, or to suggest a social status between individuals.

In particular, each facet of language has its own communication goal. We define communication goals that can be achieved from each facet below:

- The **knowledge** facet is used to achieve *factual* goals.
  - Content selection by retrieving factual knowledge or memories and reasoning about such knowledge is intended to convey trustfworthy pieces of information to the listener as a factual text.
- The **structure** facet is used to achieve *coherence* goals.
  - Content structuring by planning and abstraction is intended to make the content more cohesive and eventually make the communication more effective.
- The **style** facet is used to achieve *social* goals.
  - *Inter-personal styles* (e.g., formality, politeness) are used to intentionally or unintentionally form a better social relationship with the listener.
    - e.g., literal meaning ("It was a nice meeting") + social goal (RESPECTFUL) → polite text "It was my great pleasure to meet you."
  - *Personal styles* (e.g., gender, ethnicity, geography) are used to unconsciously (or sometimes consciously) express the speaker's persona by using dialect or identifying his or her group characteristics to the listener.
    - e.g., literal meaning "He failed out ages ago" + persona goal (AFRICAN AMERICAN) → "He **done** failed out" (**done** marks distance, past tense).
  - *Affective styles* (e.g., offense, romance, emotions) are used to synchronize emotional status across individuals and promote social interaction (Nummenmaa et al., 2012)
    - E.g., literal meaning ("You are a woman") + social goal (ROMANCE) → romantic text "You are the only woman I ever want in my life"
  - *Figurative styles* (e.g., metaphor, sarcasm, humor, hyperbole) are used to amplify the message of text by using various figures of speech and make communication to be more effective, persuasive, or impactful.
    - E.g., literal meaning ("Trump is not great") + social goal (SATIRICAL) → sarcastic text "It was 'great' year, Trump!"

The factual goal (for the knowledge facet) and coherence goal (for the structure facet) are straightforward to understand. However, the style facet is less so. Style is the most complex facet to define the role, understand its textual variation, and empirically validate its effect, due to its relation to social idiosyncrasies and linguistic variations. Social aspects of style in language have been studied for many decades in various fields such as linguistics, sociology, philosophy, and sociolinguistics, among other fields. For a better understanding of "why" styles are used in language, we study several linguistic theories related to style in §4.1.

This thesis does not provide any empirical verification of our proposal on the intended usage of language, particularly on the combined usage of the three facets. Instead, we show that the individual facet's goal can be quantified as a proxy measurement on a variety of downstream generation tasks. For example, for the knowledge facet, we measure *factuality* on open-ended question answering task by checking whether our system can predict the correct answer that requires appropriate external knowledge and reasoning over it. For the structure facet, we measure *coherence* on paragraph infilling/unmasking task by checking how our system generates the text with respect to context and evaluate how similar it is to the human-written reference text. For the style facet, instead of showing what *social* goals each individual style achieved, we show a comprehensive analysis of how various styles are combined together.

1.3 Research Questions and Technical Contributions

My research aims to build human-like NLG systems by repackaging the SFL linguistic theory, focusing on three facets such as knowledge, structures, and styles, and presenting effective computing methods for handling each facet in a wide range of generation tasks. The research questions and key contributions of my work are as follows.

1.3.1 Neural-symbolic integration for knowledge-augmented generation

This material is developed fully in Section 2. "What" knowledge and its processing are needed to produce more factual text? Most human-human conversations are open-ended and thus some partial information is missing between the utterances. It requires the ability to use external knowledge and reasoning processing over this knowledge to derive factual responses.

We addressed this challenge by integrating two opposing (i.e., neural and symbolic) systems for knowledge representation and to take the advantages of each: *lexical generalization* enabled by the neural system and *reasoning capabilities* provided from the symbolic system. We developed three different ways of integrating the two systems through embedding representations (Kang et al., 2020), data (Kang et al., 2018d), and modules (Kang et al., 2018c) (Figure 1.6). The guidance from the symbolic knowledge and module not only helps fill the knowledge gaps left by the neural-only system but provides better interpretability of how the final answer is derived through an explicit reasoning process over the

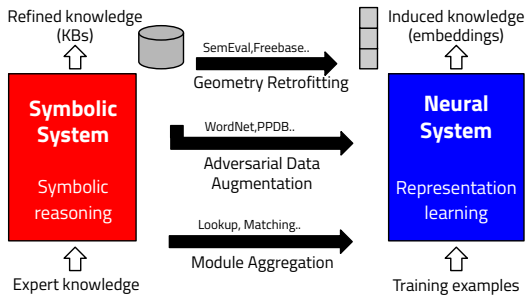


Figure 1.6: Neural-symbolic integration.

knowledge. We believe the neural-symbolic integration systems resemble the human cognitive process and are effective to deal with both symbolic and numeric representations, generating responses that are more logical, knowledge-aware, and explainable.

1.3.2 Text-planning for coherently-structured generation

This material is developed fully in Section 3. "How" can the model compose multiple texts (e.g., a story) coherently? Every part of a text plays a role in the whole, and the text is coherent ONLY if the reader can assemble the complete complex picture correctly by putting each piece in its correct place. A system that just randomly throws together facts (even if they are each true) does not produce coherent text. To ensure this, two functionalities are re-needed: (1) the delimitation of sentence-sized chunks from the complete total, and (2) their organization (in linear order, and/or subordination). This is called *planning* in the NLG literature.

Humans often make structural decisions before making utterances (e.g., topic introduction, ordering, conversation strategies) to ensure the coherence between the utterances or align their utterances with a certain goal of the conversation. We call such structural decisions as *plans*. Our proposed mechanism of **text-planning** is a hierarchical generation process of multiple texts by guiding the surface-level realization (i.e., LM) to match high-level plans. Such plans can be discourse relations between texts, framing, strategies in goal-oriented dialogues, speech acts, topics, and more. We suggest three-dimensional text planning (Figure 1.7): The *horizontal* planning focuses on producing a coherent long text such as a paragraph, whereas *vertical* planning focuses on producing semantically diverse texts (e.g., generating diverse reviews about different aspects of the same product). The *hierarchical* planning focuses on abstracting the meaning of multiple sentences into a short summary text.

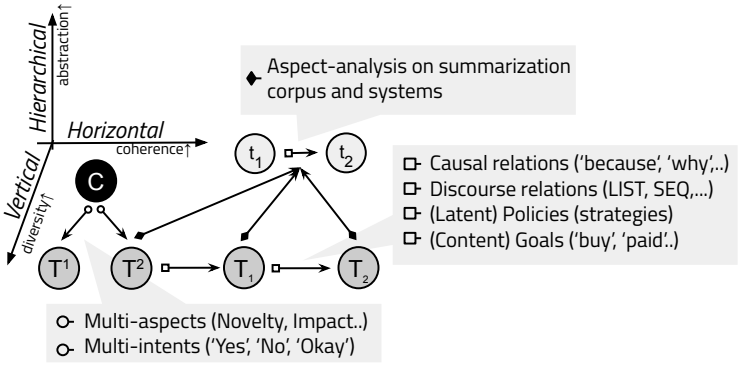


Figure 1.7: Text planning.

For horizontal text planning, we proposed three forms of horizontal plans: **discourse relations**, **script-like content**, and **strategic policies**. One of our studies (Kang et al., 2017b) produced a chain of causally-linked texts to explain the temporal causality between two time-series events. Another study (Kang et al., 2019c) extended the causal relation to more general discourse relations based on rhetorical structure theory. Motivated by the script theory (Schank, 1975), we proposed a text planner that first predicts keywords to say (i.e. plan words) based on the context and then guides a text generator (e.g., GPT2) to generate surface words using copy mechanism according to the predicted plan keywords (Kang and Hovy, 2020). Both the relation and content plans improve the coherency of output text on our newly-proposed generation tasks: *paragraph bridging* and *paragraph unmasking*. The plan can be also represented as a latent form using hierarchical modeling between adjacent sentences (Kang et al., 2019c) or policy learning with

bot-playing in machine-machine conversations (Kang et al., 2019a).

Vertical text planning is yet under-explored except for some industrial applications (e.g., SmartReply (Kannan et al., 2016)). We collected a new dataset PeerRead (Kang et al., 2018a) that includes academic papers and their corresponding peer-reviews from various venues. The dataset suggests a new generation task: **aspect**-specific automatic reviewing, that writes multiple reviews about a paper with respect to different aspects (e.g., novelty, impact).

Text planning helps the model generate coherent utterances that are faithful to the goal of the conversation. Ideally, both horizontal and vertical planning should take place at the same time in parallel and with other cognitive mechanisms, such as hierarchical text planning (Kang et al., 2019d). This would be an important direction for future research.

1.3.3 Cross-stylization for stylistically-appropriate generation

This material is developed fully in Section 4. How can we make the model produce stylistically appropriate output depending on "who" you are and "whom" you talk to? Every natural text is written or spoken in some style. The style is constituted by a complex combination of different stylistic factors, including formality markers, emotions, metaphors, etc. Some factors implicitly reflect the author's personality, while others are explicitly controlled by the author's choices to achieve some personal or social goal. The factors combine and co-vary in complex ways to form styles. One cannot form a complete understanding of a text and its author without considering these factors. Studying the nature of the co-varying combinations of the factors sheds light on stylistic language in general, sometimes called **cross-style language understanding**.

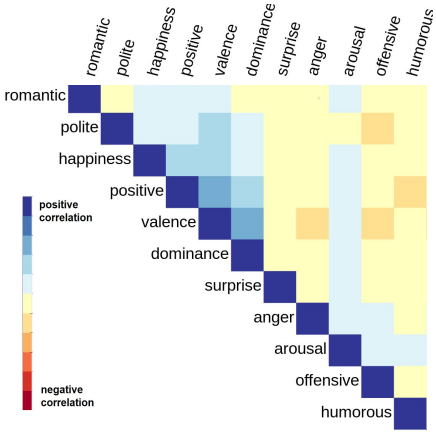


Figure 1.8: Cross-stylization.

One of the challenges in cross-style study is the lack of an appropriate dataset, leading to invalid model development and uncontrolled experiments. To address that, we collect two datasets: PASTEL (Kang et al., 2019b) and xSLUE (Kang and Hovy, 2019): The speaker's personal traits (e.g., age, gender, political view) may be reflected in his or her text. PASTEL is a stylistic language dataset that consists of descriptions about a common set of situations written by people with different personas. The main goal of the dataset is to preserve meaning between texts while promoting stylistic diversity. PASTEL contributed to appropriately designing models and experiments in style classification and style transfer tasks.

xSLUE, on the other hand, provides a benchmark corpus for cross-style language understanding and evaluation. It contains text in 15 different styles and 23 classification tasks. Our analysis shows that some styles are highly dependent on each other (e.g., impoliteness and offense) (Figure 1.8), showing the importance of modeling the inter-dependencies of different styles in stylistic variation.

Stylistically appropriate NLG systems should act like an orchestra conductor. An orchestra is a large ensemble of instruments that jointly produce a single integrate *message*. What we only hear at the end is the harmonized sound of complex interacting combinations of individual instruments, where the conductor controls their combinatory choices (e.g., score, tempo). Some

instruments are in the same category such as bowed strings for the violin and cello. Similarly, text is an output that reflects a complex combination of different style factors where each has its own lexical choices but some factors are dependent on each other. We believe modeling the complex combination and finding the dependency between two styles or between content and style are an important step toward being a maestro of cross-style language generation.

1.4 Goal and Scope of Thesis

NLG is a very rich and understudied topic in NLP. It exposes many as-yet unaddressed phenomena and aspects, especially relating to styles, and also requires complex multi-faceted planning across them all. Better understanding of these issues will not only produce better NLG systems but also enrich and inform other areas of NLP, such as machine translation, summarization, question-answering, dialogues, etc. However, some fundamental advances are needed.

This thesis has identified and addressed some of the facets at a certain degree. Addressing them, plus other facets, is absolutely required in order to build human-like NLG systems. For each facet, we developed prototypical yet effective cognitive-systems: neural-symbolic integration, text planning, and cross-stylization. Although this work proposes a multi-faceted architecture, we studied individual facets separately. However, our daily conversations often require a combination of multiple facets together. The development of such systems needs more cognitive architectures developed such as the cascading system proposed (Figure 1.5 (c)) where multiple facets can dynamically interact with each other.

Furthermore, we argue that language generation is intended to achieve certain communication goals, and each facet of language has its own goal. If one can develop an NLG system whose individual facets effectively operate and interact with each other, such a system can achieve factual, coherence, and social goals in communication as well as humans do.

Following what has been labeled the Bender, it is necessary to spell out that this work applies mostly to work in English[2]. That said, the general framework of multifaceted generation and its philosophy can be easily applied to any other language. However, the extensibility of the developed computing systems to other languages depends on the availability of datasets, annotations, and external databases we used in the experiments.

[2]Emily Bender, `https://twitter.com/emilymbender/status/1135907994678562817`, June 4, 2019. See also Emily Bender, "English Isn't Generic for Language, Despite What NLP Papers Might Lead You to Believe," Symposium on Data Science and Statistics, Bellevue, Washington, May 30, 2019, `http://faculty.washington.edu/ebender/papers/Bender-SDSS-2019.pdf`.

Chapter 2

Knowledge-Augmented Generation

We begin by incorporating our *knowledge* facet into our language generation. Note that we fixed other facets, such as styles and structures. Thus, this chapter will be focused on studying the effect of the knowledge facet alone.

By definition, the *knowledge* facet should decide the basic semantics of communication. In other words, "what" knowledge must be processed to produce more factual text? Most conversations between humans are open-ended, thus information is missing between the utterances. In this setting, external knowledge and reasoning is required to derive facets from the responses. The external knowledge can be commonsense knowledge, lexical ontologies, episodic memory, an external knowledge base, and more. The reasoning process can be formal logic, temporal or spatial reasoning, numerical reasoning, negation, and more.

The factualness of language generation is usually measured on downstream tasks that require external knowledge, such as open-ended question answering (QA) or knowledge graph completion tasks. Chen et al. (2017) showed the effectiveness of utilizing external knowledge bases, such as Wikipedia documents, in order to achieve better performance on open-domain machine comprehension tasks. Petroni et al. (2019) showed that pre-trained language models like BERT (Devlin et al., 2019) can be used as knowledge bases themselves, showing remarkable performance on open-ended QA tasks. they also contain more factual knowledge than standard language models.

This chapter introduces a task called *open-ended question answering* that requires knowledge and an associated reasoning process (§2.1) and proposes a general framework called *neural-symbolic integration* to tackle the issues in the task (§2.2). In §2.3, we summarize previous work on our proposed approaches.

2.1 Task: Open-ended Question Answering

Question answering between two agents is common form of communication. It can be often categorized into *close-ended* or *open-ended*. Closed-ended QA is characterized by questions that can be answered by reading the input text while searching for the correct answer, such as machine comprehension (Hermann et al., 2015; Rajpurkar et al., 2016). On the other hand, open-ended QA is characterized by questions that can be answered by retrieving external background

knowledge and reasoning over the input text and knowledge to predict the answer. This thesis focuses on addressing the open-ended QA problem, by casting it as an entailment problem.

Alice: "What surface would be the best for roller skating?"
Bob: "Blacktop, I guess." Carol: "Sand is good for roller skating."

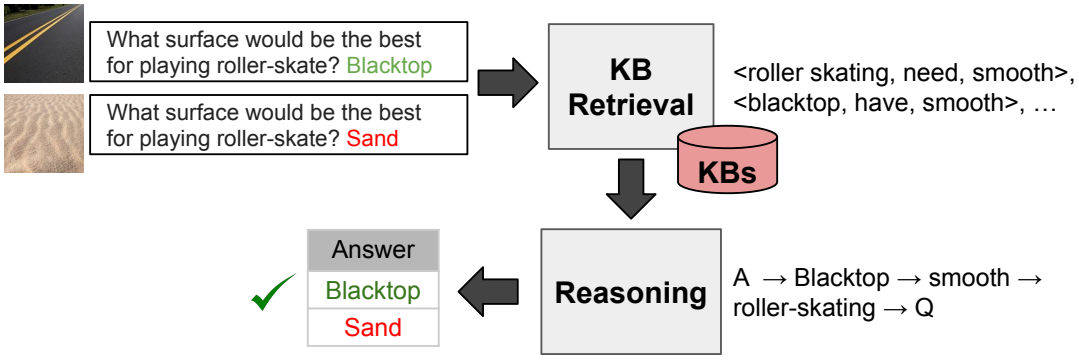


Figure 2.1: Open-ended question answering.

Figure 2.1 shows an example conversation of open-ended QA. Alice asks a question and there are two possible answers, one each from by Bob and Carol. In order to correctly answer Alice's question, Bob needs to have relevant background knowledge from external sources to know that roller skating needs a smooth face, and that blacktop is smooth. This is represented by the keywords in the figures: <`roller skating`, `need`, `smooth`>, <`blacktop`, `have`, `smooth`>. The next process is to reason out the correct answer based on the knowledge and question: *Question* $\xrightarrow{surface}$ *Roller Skating* $\xrightarrow{need}$ *Smooth* $\xrightarrow{have}$ *Blacktop* → *Answer*. Traditionally, the problem can be addressed by developing an explicit symbolic reasoning engine written in a formal logic, like LISP, which examines the retrieved knowledge. However, implementing a symbolic reasoning program capable of producing natural language is particularly challenging due to the generalization of lexical meanings and structures of languages.

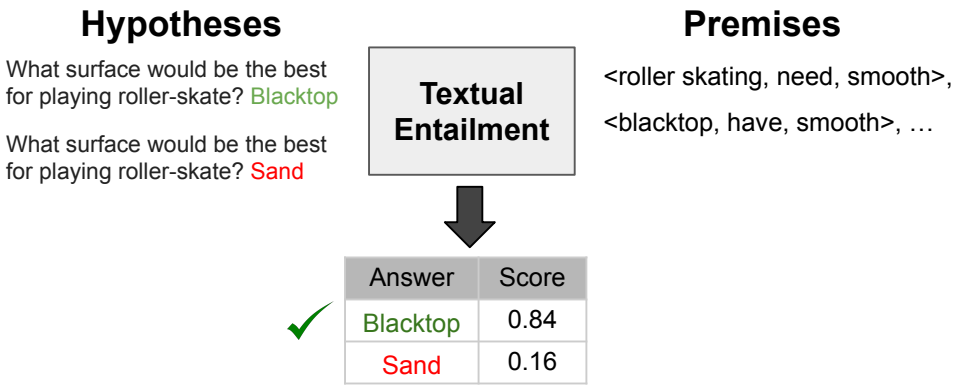


Figure 2.2: Open-ended QA can be cast as a textual entailment problem, predicting whether the hypothesis (question+answer) is entailed by the premise (retrieved knowledge tuples).

As an alternative to a symbolic language reasoner, we cast the problem as a textual entailment classification (Figure 2.2). For each possible answer, we concatenate it with the question, then retrieve the necessary knowledge from external sources. Finally, we generate a score based on whether the combined text of the question and an answer choice (*hypothesis*) entails the retrieved knowledge text (*premise*) or not. We answer the question with the highest-scored answer choice. We propose various approaches to implement the textual entailment engine in the following section.

2.2 Proposed Approach: *Neural-Symbolic Learning* (NSL)

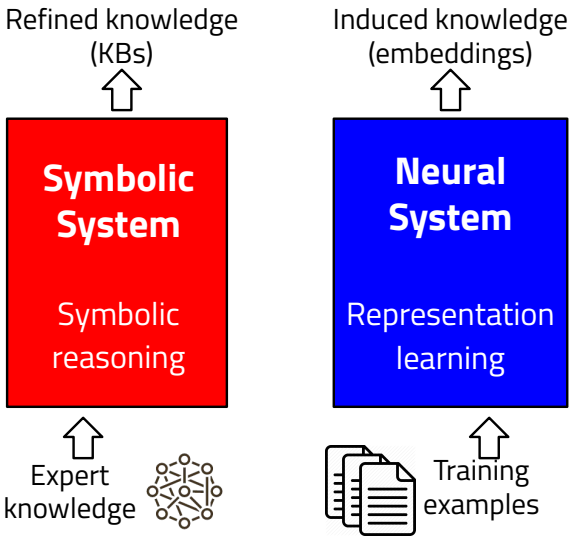


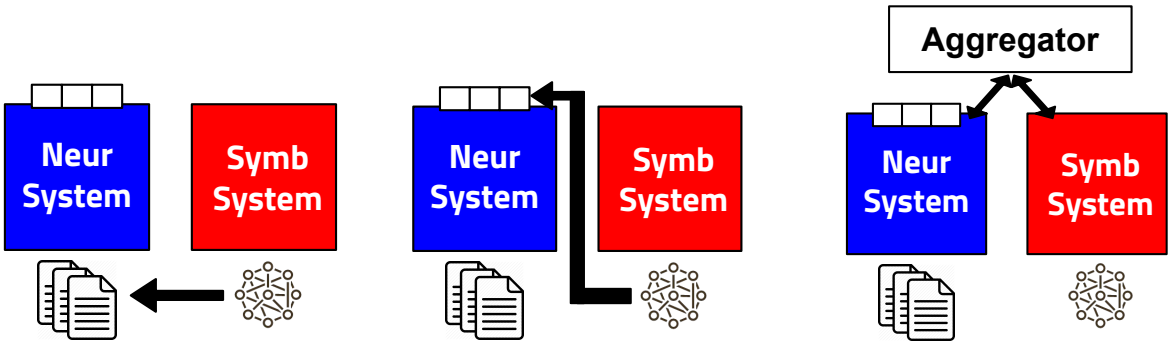
Figure 2.3: Symbolic system (left) and neural system (right).

This section begins by comparing learning methods accomplished via a neural system and a symbolic system. Following this, we propose our framework, called *neural-symbolic learning*, which combines the advantages of the two systems. Each system has its pros and cons (Figure 2.3). Symbolic systems take expert knowledge, like knowledge graphs and databases, then reason over it, finally inferring the refined knowledge as an answer. The explicit reasoning over the symbolic knowledge helps interpret how the answer is derived from a particular set of knowledge and by what what procedure.

On the other hand, neural systems learn internal representations or patterns from unstructured data, and output induced knowledge, often called embeddings. The representation is learned through a fully or partially connected neural network with or without annotated output labels. The connectivity from the neural net creates embeddings which contain complicated patterns of lexically or structurally similar text, giving neural systems a strong generalizing power.

We solved the problem of open-ended QA, by combining the two opposing systems to take the advantages of each: the lexical and structural generalization from the neural system and the explicit knowledge and reasoning power from the symbolic system. The generalization helps

understand whether two lexical terms or sentence structures are similar or not. In the previous example, "roller-skate" in the question and "roller skating" in the knowledge base are lexically similar. On the other hand, external knowledge such as <roller skating, need, smooth> is retrieved from existing symbolic knowledge bases. Moreover, explicit reasoning is a process of drawing inferences between the question and the knowledge items to conclude the correct answer, such as $Q \xrightarrow{surface} Roller\ Skating \xrightarrow{need} ... \rightarrow A$.



(a) Data-driven NSL **(b)** Embed-driven NSL **(c)** Model-driven NSL

Figure 2.4: Neural-Symbolic Learning: *data-driven* (Kang et al., 2018d), *embedding-driven* (Kang et al., 2020), and *model-driven* (Kang et al., 2018c).

Here we propose three distinct types of neural-symbolic integration: *data-driven*, *embedding-driven*, and *model-driven*. Figure 2.4 shows the conceptual difference of the three approaches when combining the two systems.

The *data-driven* approach incorporates symbolic knowledge directly into a neural model's training data to make it more robust against adversarial attacks. For instance, Kang et al. (2018d) augmented a neural system with symbolic knowledge, like WordNet and PPDB relationships, into the neural system using adversarial training. This improves the robustness of long-tail linguistic phenomena, such as negation, which neural-only systems cannot handle. Moreover, symbolic knowledge helps the most when the neural model is trained on less data. Further details are described in §2.4.

The *embedding-driven* approach incorporates symbolic knowledge into the word embeddings from the neural model. In particular, Kang et al. (2020) constrained the geometric properties of word-pair relations, like slope(queen,king) ~ slope(woman,man), to the word embedding vectors. The geometric constraints for the vectors were created by relational knowledge sources, such as <Part-Of(Tire, Car)> relation in SemEval. These geometrically-constrained vectors showed better relationship inference performance on various tasks such as textual entailment, semantic similarity, and more. Further details are described in §2.5.

Separately, the *model-driven* approach indirectly integrates the neural and symbolic modules. Kang et al. (2018c) combined symbolic modules, like lookup and matching, with neural modules, like decomposable attention, by merging them with an aggregation module. They achieved this by sharing every internal parameter in an end-to-end fashion. Further details are described in §2.6.

2.3 Related Work

We survey prior work on the three different ways of combining neural and symbolic systems (or knowledge) as follows:

Data-driven Approaches Incorporating external rules or linguistic resources in a deep learning model generally requires substantially adapting the model architecture (Sha et al., 2016; Liang et al., 2016). This is a model-dependent approach, which can be cumbersome and constraining. Similarly, non-neural textual entailment models have been developed that incorporate knowledge bases. However, these also require model-specific engineering (Raina et al., 2005; Haghighi et al., 2005; Silva et al., 2018).

An alternative is a model- and task-independent route of incorporating linguistic resources via word embeddings that are *retro-fitted* (Faruqui et al., 2015) or *counter-fitted* (Mrksic et al., 2016) to such resources. We demonstrate, however, that this has a little positive impact in our setting and can even be detrimental. Further, it is unclear how to incorporate knowledge sources into advanced representations such as contextual embeddings (McCann et al., 2017; Peters et al., 2018a). Logical rules have also been defined to label existing examples based on external resources (Hu et al., 2016). Our focus here is on generating *new* training examples.

Our use of the GAN framework to create a better discriminator is related to CatGANs (Wang and Zhang, 2017) and TripleGANs (Chongxuan et al., 2017) where the discriminator is trained to classify the original training image classes as well as a new 'fake' image class. We, on the other hand, generate examples belonging to the same classes as the training examples. Further, unlike the earlier focus on the vision domain, this is the first approach to train a discriminator using GANs for a natural language task with discrete outputs.

Embedding-driven Approaches Our work is motivated by the previous analysis of word embeddings: Levy and Goldberg (2014); Levy et al. (2015) claim that relational similarities could be viewed as a mixture of attributional similarities, each one reflecting a different aspect. Without intrinsic modeling of the attributes or properties, linearity behind the offset model usually does not hold (See Figure 2.13 for empirical evidence for this). Linzen (2016) points out the limitation of the linear offset model when the offset of relation is too small. We find the cases (e.g., Similar) and augment their unique regularities using geofitting. Compared to prior analyses, we focus on exploring the inherent properties of each relation and its dependencies.

In a broader context, geofitting technique can be viewed as a knowledge injection: Different types of knowledge have been studied to incorporate them into word embeddings: lexical knowledge (e.g., PPDB) (Faruqui et al., 2015; Kang et al., 2018d), multiple word senses in an ontology (Jauhar et al., 2015), and antonymy and synonymy (Mrksic et al., 2016). Recently, Lengerich et al. (2018) developed a Functional retrofitting technique that incorporates the explicit relation representation as retrofitting penalty terms for better graph completion, while our work extracts geometric properties of relations and injects them into original word embeddings to be applied to more general tasks.

Model-driven Approaches Compared to neural only (Bowman et al., 2015; Parikh et al., 2016) or symbolic only (Khot et al., 2017; Khashabi et al., 2016) systems, our approach takes advantage of both systems, often called neural-symbolic learning (Garcez et al., 2015). Various neural-symbolic models have been proposed for question answering (Liang et al., 2016) and causal explanations (Kang et al., 2017b). My work focuses on end-to-end training of these models specifically for textual entailment. Contemporaneous to this work, Chen et al. (2018) have incorporated knowledge-bases within the attention and composition functions of a neural entailment model, while Kang et al. (2018d) generate adversarial examples using symbolic knowledge (e.g., WordNet) to train a robust entailment model. We focused on integrating knowledge-bases via a separate symbolic model to fill the knowledge gaps.

In the following three sub-chapters §2.4, §2.5, and §2.6, we describe, in detail, the problem formulation, approach, and experimental results for all three ways to create NSL.

2.4 Data-driven NSL: Adversarial Knowledge Augmentation

2.4.1 Introduction

The impressive success of machine learning models on large natural language datasets often does not carry over to moderate training data regimes, where models often struggle with infrequently observed patterns and simple adversarial variations. A prominent example of this phenomenon is *textual entailment*, the fundamental task of deciding whether a *premise* text entails ($\models$) a *hypothesis* text. On certain datasets, recent deep learning entailment systems (Parikh et al., 2016; Wang et al., 2017; Gong et al., 2017) have achieved close to human level performance. Nevertheless, the problem is far from solved, as evidenced by how easy it is to generate minor adversarial examples that break even the best systems. As Table 2.1 illustrates, a state-of-the-art neural system for this task, namely the Decomposable Attention Model (Parikh et al., 2016), fails when faced with simple linguistic phenomena such as negation, or a re-ordering of words. This is not unique to a particular model or task. Minor adversarial examples have also been found to easily break neural systems on other linguistic tasks such as reading comprehension (Jia and Liang, 2017).

Table 2.1: Failure examples from the SNLI dataset: negation (Top) and re-ordering (Bottom). **P** is premise, **H** is hypothesis, and **S** is prediction made by an entailment system (Parikh et al., 2016).

P: The dog did not eat all of the chickens.
H: The dog ate all of the chickens.
S: entails (score 56.5%)
P: The red box is in the blue box.
H: The blue box is in the red box.
S: entails (score 92.1%)

A key contributor to this brittleness is the use of specific datasets such as SNLI (Bowman et al., 2015) and SQuAD (Rajpurkar et al., 2016) to drive model development. While large and

challenging, *these datasets also tend to be homogeneous*. E.g., SNLI was created by asking crowd-source workers to generate entailing sentences, which then tend to have limited linguistic variations and annotation artifacts (Gururangan et al., 2018b). Consequently, models overfit to sufficiently repetitive patterns—and sometimes idiosyncrasies—in the datasets they are trained on. They fail to cover long-tail and rare patterns in the training distribution, or linguistic phenomena such as negation that would be obvious to a layperson.

To address this challenge, we propose to *train textual entailment models more robustly using adversarial examples* generated in two ways: (a) by incorporating knowledge from large linguistic resources, and (b) using a sequence-to-sequence neural model in a GAN-style framework.

The motivation stems from the following observation. While deep-learning based textual entailment models lead the pack, they generally do not incorporate intuitive rules such as negation, and ignore large-scale linguistic resources such as PPDB (Ganitkevitch et al., 2013) and WordNet (Miller, 1995). These resources could help them generalize beyond specific words observed during training. For instance, while the SNLI dataset contains the pattern *two men* $\models$ *people*, it does not contain the analogous pattern *two dogs* $\models$ *animals* found easily in WordNet.

Effectively integrating simple rules or linguistic resources in a deep learning model, however, is challenging. Doing so directly by substantially adapting the model architecture (Sha et al., 2016; Chen et al., 2018) can be cumbersome and limiting. Incorporating such knowledge indirectly via modified word embeddings (Faruqui et al., 2015; Mrksic et al., 2016), as we show, can have little positive impact and can even be detrimental.

Our proposed method, which is task-specific but model-independent, is inspired by data-augmentation techniques. We generate new training examples by applying knowledge-guided rules, via only a handful of rule templates, to the original training examples. Simultaneously, we also use a sequence-to-sequence or seq2seq model for each entailment class to generate new hypotheses from a given premise, adaptively creating new adversarial examples. These can be used with any entailment model without constraining model architecture.

We also introduce the first approach to train a robust entailment model using a Generative Adversarial Network or GAN (Goodfellow et al., 2014) style framework. We iteratively improve both the entailment system (the *discriminator*) and the differentiable part of the data-augmenter (specifically the neural *generator*), by training the generator based on the discriminator's performance on the generated examples. Importantly, unlike the typical use of GANs to create a strong generator, we use it as a mechanism to create a strong and robust discriminator.

Our new entailment system, called AdvEntuRe, demonstrates that in the moderate data regime, adversarial iterative data-augmentation via only a handful of linguistic rule templates can be surprisingly powerful. Specifically, we observe 4.7% accuracy improvement on the challenging SciTail dataset (Khot et al., 2018) and a 2.8% improvement on 10K-50K training subsets of SNLI. An evaluation of our algorithm on the negation examples in the test set of SNLI reveals a 6.1% improvement from just a single rule.

2.4.2 Adversarial Example Generation

We present three different techniques to create adversarial examples for textual entailment. Specifically, we show how external knowledge resources, hand-authored rules, and neural language

generation models can be used to generate such examples. Before describing these generators in detail, we introduce the notation used henceforth.

We use lower-case letters for single instances (e.g., x, p, h), upper-case letters for sets of instances (e.g., X, P, H), blackboard bold for models (e.g., $\mathbb{D}$), and calligraphic symbols for discrete spaces of possible values (e.g., class labels $\mathcal{C}$). For the textual entailment task, we assume each example is represented as a triple (p, h, c), where p is a premise (a natural language sentence), h is a hypothesis, and c is an entailment label: (a) *entails* ($\sqsubseteq$) if h is true whenever p is true; (b) *contradicts* ($\curlywedge$) if h is false whenever p is true; or (c) *neutral* (#) if the truth value of h cannot be concluded from p being true.[1]

We will introduce various example generators in the rest of this section. Each such generator, $\mathbb{G}_\rho$, is defined by a partial function f_ρ and a label g_ρ. If a sentence s has a certain property required by f_ρ (e.g., contains a particular string), f_ρ transforms it into another sentence s' and g_ρ provides an entailment label from s to s'. Applied to a sentence s, $\mathbb{G}_\rho$ thus either "fails" (if the pre-requisite isn't met) or generates a new entailment example triple, $\left(s, f_\rho(s), g_\rho\right)$. For instance, consider the generator for ρ:=hypernym(car, vehicle) with the (partial) transformation function f_ρ:="Replace *car* with *vehicle*" and the label g_ρ:=*entails*. f_ρ would fail when applied to a sentence not containing the word "car". Applying f_ρ to the sentence s="*A man is driving the car*" would generate s'="*A man is driving the vehicle*", creating the example ($s, s', entails$).

The seven generators we use for experimentation are summarized in Table 2.2 and discussed in more detail subsequently. While these particular generators are simplistic and one can easily imagine more advanced ones, we show that training using adversarial examples created using even these simple generators leads to substantial accuracy improvement on two datasets.

Source	$\boldsymbol{\rho}$	$\boldsymbol{f_\rho}$(s)	$\boldsymbol{g_\rho}$
Knowledge Base, $\mathbb{G}^{\text{KB}}$			
WordNet	hyper(x, y)	Replace x with y in s	$\sqsubseteq$
	anto(x, y)		$\curlywedge$
	syno(x, y)		$\sqsubseteq$
PPDB	$x \equiv y$		$\sqsubseteq$
SICK	$c(x, y)$		c
Hand-authored, $\mathbb{G}^{\text{H}}$			
Domain knowledge	NEG	NEGATE(s)	$\curlywedge$
Neural Model, $\mathbb{G}^{\text{s2s}}$			
Training data	(s2s, c)	$\mathbb{G}^{\text{s2s}}_c(s)$	c

Table 2.2: Various generators $\mathbb{G}_\rho$ characterized by their source, (partial) transformation function f_ρ as applied to a sentence s, and entailment label g_ρ

[1]The symbols are based on Natural Logic (Lakoff, 1970) and use the notation of (MacCartney and Manning, 2014).

Knowledge-Guided Generators

Large knowledge-bases such as WordNet and PPDB contain lexical equivalences and other relationships highly relevant for entailment models. However, even large datasets such as SNLI generally do not contain most of these relationships in the training data. E.g., that *two dogs* entails *animals* isn't captured in the SNLI data. We define simple generators based on lexical resources to create adversarial examples that capture the underlying knowledge. This allows models trained on these examples to learn these relationships.

As discussed earlier, there are different ways of incorporating such symbolic knowledge into neural models. Unlike task-agnostic ways of approaching this goal from a word embedding perspective (Faruqui et al., 2015; Mrksic et al., 2016) or the model-specific approach (Sha et al., 2016; Chen et al., 2018), we use this knowledge to generate task-specific examples. This allows any entailment model to learn how to use these relationships *in the context of the entailment task*, helping them outperform the above task-agnostic alternative.

Our knowledge-guided example generators, $\mathbb{G}_\rho^{\text{KB}}$, use lexical relations available in a knowledge-base: $\rho := r(x, y)$ where the relation r (such as synonym, hypernym, etc.) may differ across knowledge bases. We use a simple (partial) transformation function, $f_\rho(s)$:="Replace x in s with y", as described in an earlier example. In some cases, when part-of-speech (POS) tags are available, the partial function requires the tags for x in s and in $r(x, y)$ to match. The entailment label g_ρ for the resulting examples is also defined based on the relation r, as summarized in Table 2.2.

This idea is similar to Natural Logic Inference or NLI (Lakoff, 1970; Byrd, 1986; Angeli and Manning, 2014) where words in a sentence can be replaced by their hypernym/hyponym to produce entailing/neutral sentences, depending on their context. We propose a context-agnostic use of lexical resources that, despite its simplicity, already results in significant gains. We use three sources for generators:

WordNet (Miller, 1995) is a large, hand-curated, semantic lexicon with synonymous words grouped into *synsets*. Synsets are connected by many semantic relations, from which we use *hyponym* and *synonym* relations to generate entailing sentences, and *antonym* relations to generate contradicting sentences[2]. Given a relation $r(x, y)$, the (partial) transformation function f_ρ is the POS-tag matched replacement of x in s with y, and requires the POS tag to be noun or verb. NLI provides a more robust way of using these relations based on context, which we leave for future work.

PPDB (Ganitkevitch et al., 2013) is a large resource of lexical, phrasal, and syntactic paraphrases. We use 24,273 lexical paraphrases in their smallest set, PPDB-S (Pavlick et al., 2015), as equivalence relations, $x \equiv y$. The (partial) transformation function f_ρ for this generator is POS-tagged matched replacement of x in s with y, and the label g_ρ is *entails*.

SICK (Marelli et al., 2014b) is dataset with entailment examples of the form (p, h, c), created to evaluate an entailment model's ability to capture compositional knowledge via hand-authored rules. We use the 12,508 patterns of the form $c(x, y)$ extracted by Beltagy et al. (2016) by comparing sentences in this dataset, with the property that for each SICK example (p, h, c), replacing (when applicable) x with y in p produces h. For simplicity, we ignore positional information in these patterns. The (partial) transformation function f_ρ is replacement of x in s with y, and the

[2]A similar approach was used in a parallel work to generate an adversarial dataset from SNLI (Glockner et al., 2018).

label g_ρ is c.

Hand-Defined Generators

Even very large entailment datasets have no or very few examples of certain otherwise common linguistic constructs such as negation,[3] causing models trained on them to struggle with these constructs. A simple model-agnostic way to alleviate this issue is via a negation example generator whose transformation function $f_\rho(s)$ is NEGATE(s), described below, and the label g_ρ is *contradicts*.

NEGATE(S): If s contains a 'be' verb (e.g., is, was), add a "not" after the verb. If not, also add a "did" or "do" in front based on its tense. E.g., change "A person is crossing" to "A person is not crossing" and "A person crossed" to "A person did not cross." While many other rules could be added, we found that this single rule covered a majority of the cases. Verb tenses are also considered[4] and changed accordingly. Other functions such as dropping adverbial clauses or changing tenses could be defined in a similar manner.

Both the knowledge-guided and hand-defined generators make local changes to the sentences based on simple rules. It should be possible to extend the hand-defined rules to cover the long tail (as long as they are procedurally definable). However, a more scalable approach would be to extend our generators to trainable models that can cover a wider range of phenomena than hand-defined rules. Moreover, the applicability of these rules generally depends on the context which can also be incorporated in such trainable generators.

Neural Generators

For each entailment class c, we use a trainable sequence-to-sequence neural model (Sutskever et al., 2014; Luong et al., 2015) to generate an entailment example (s, s', c) from an input sentence s. The seq2seq model, trained on examples labeled c, itself acts as the transformation function f_ρ of the corresponding generator $\mathbb{G}_c^{\text{s2s}}$. The label g_ρ is set to c. The joint probability of seq2seq model is:

$$\mathbb{G}_c^{\text{s2s}}(X_c; \phi_c) = \mathbb{G}_c^{\text{s2s}}(H_c, P_c; \phi_c) \tag{2.1}$$

$$= \Pi_i P(h_{i,c} | p_{i,c}; \phi_c) P(h_i) \tag{2.2}$$

The loss function for training the seq2seq is:

$$\hat{\phi}_c = \arg\min_{\phi_c} L(H_c, \mathbb{G}_c^{\text{s2s}}(X_c; \phi_c)) \tag{2.3}$$

where L is the cross-entropy loss between the original hypothesis H_c and the predicted hypothesis. Cross-entropy is computed for each predicted word w_i against the same in H_c given the sequence of previous words in H_c . $\hat{\phi}_c$ are the optimal parameters in $\mathbb{G}_c^{\text{s2s}}$ that minimize the loss for class c. We use the single most likely output to generate sentences in order to reduce decoding time.

[3]Only 211 examples (2.11%) in the SNLI training set contain negation triggers such as not, 'nt, etc.

[4]https://www.nodebox.net/code/index.php/Linguistics

Example Generation

The generators described above are used to create new entailment examples from the training data. For each example (p, h, c) in the data, we can create two new examples: $\big(p, f_\rho(p), g_\rho\big)$ and $\big(h, f_\rho(h), g_\rho\big)$.

The examples generated this way using $\mathbb{G}^{\text{KB}}$ and $\mathbb{G}^{\text{H}}$ can, however, be relatively easy, as the premise and hypothesis would differ by only a word or so. We therefore compose such simple ("first-order") generated examples with the original input example to create more challenging "second-order" examples. We can create second-order examples by composing the original example (p, h, c) with a generated sentence from hypothesis, $f_\rho(h)$ and premise, $f_\rho(p)$. Figure 2.5 depicts how these two kinds of examples are generated from an input example (p, h, c).

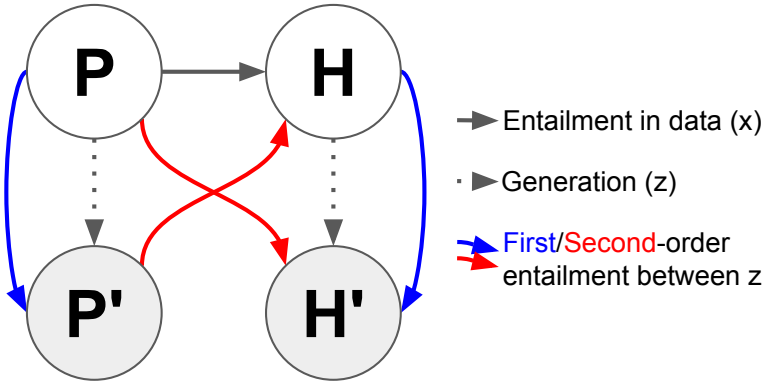


Figure 2.5: Generating first-order (blue) and second-order (red) examples.

First, we consider the second-order example between the original premise and the transformed hypothesis: $(p, f_\rho(h), \bigoplus(c, g_\rho))$, where $\bigoplus$, defined in the left half of Table 2.3, composes the input example label c (connecting p and h) and the generated example label g_ρ to produce a new label. For instance, if p entails h and h entails $f_\rho(h)$, p would entail f_ρ. In other words, $\bigoplus(\sqsubseteq, \sqsubseteq)$ is $\sqsubseteq$. For example, composing ("A man is playing soccer", "A man is playing a game", $\sqsubseteq$) with a generated hypothesis $f_\rho(h)$: "A person is playing a game." will give a new second-order entailment example: ("A man is playing soccer", "A person is playing a game", $\sqsubseteq$).

Second, we create an example from the generated premise to the original hypothesis: $(f_\rho(p), h, \bigotimes(g_\rho, c))$. The composition function here, denoted $\bigotimes$ and defined in the right half of Table 2.3, is often undetermined. For example, if p entails $f_\rho(p)$ and p entails h, the relation between $f_\rho(p)$ and h is undetermined i.e. $\bigotimes(\sqsubseteq, \sqsubseteq)$ =?. While this particular composition $\bigotimes$ often leads to undetermined or neutral relations, we use it here for completeness. For example, composing the previous example with a generated *neutral* premise, $f_\rho(p)$: "A person is wearing a cap" would generate an example ("A person is wearing a cap", "A man is playing a game", #)

The composition function $\bigoplus$ is the same as the "join" operation in natural logic reasoning (Icard III and Moss, 2014), except for two differences: (a) relations that do not belong to our three entailment classes are mapped to '?', and (b) the exclusivity/alternation relation is mapped to *contradicts*. The composition function $\bigotimes$, on the other hand, does not map to the join operation.

$p \Rightarrow h$	$h \Rightarrow h'$	$p \Rightarrow h'$	$p \Rightarrow h$	$p \Rightarrow p'$	$p' \Rightarrow h$
c	g_ρ	$\bigoplus$	c	g_ρ	$\bigotimes$
$\sqsubseteq$	$\sqsubseteq$	$\sqsubseteq$	$\sqsubseteq$	$\sqsubseteq$	?
$\sqsubseteq$	人	人	$\sqsubseteq$	人	?
$\sqsubseteq$	#	#	$\sqsubseteq$	#	#
人	$\sqsubseteq$	?	人	$\sqsubseteq$	?
人	人	?	人	人	?
人	#	#	人	#	#
#	$\sqsubseteq$	#	#	$\sqsubseteq$	#
#	人	#	#	人	#
#	#	#	#	#	#

Table 2.3: Entailment label composition functions $\bigoplus$ (left) and $\bigotimes$ (right) for creating second-order examples. c and g_ρ are the original and generated labels, resp. $\sqsubseteq$: *entails*, 人: *contradicts*, #: *neutral*, ?: undefined

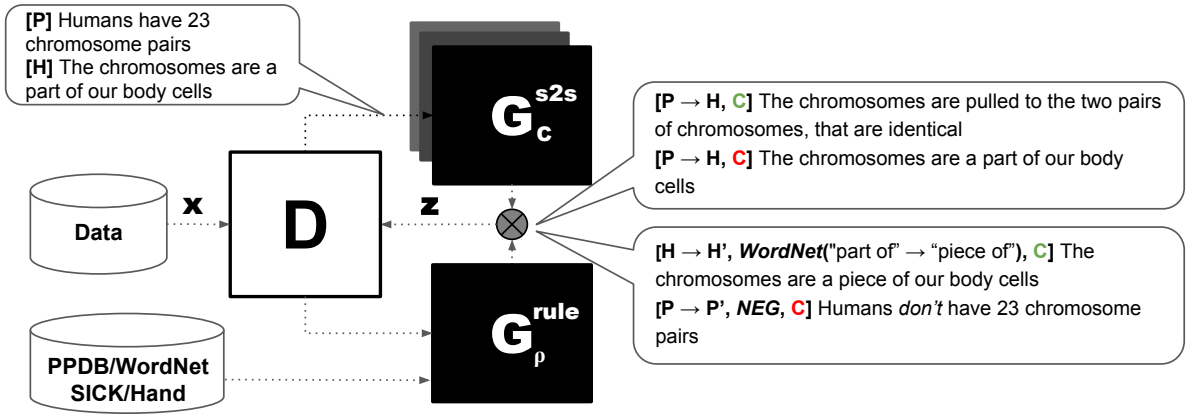


Figure 2.6: Overview of AdvEntuRe, our model for knowledge-guided textual entailment.

Implementation Details

Given the original training examples X, we generate the examples from each premise and hypothesis in a batch using $\mathbb{G}^{\text{KB}}$ and $\mathbb{G}^{\text{H}}$. We also generate new hypothesis per class for each premise using $\mathbb{G}_c^{\text{s2s}}$. Using all the generated examples to train the model would, however, overwhelm the original training set. For examples, our knowledge-guided generators $\mathbb{G}^{\text{KB}}$ can be applied in 17,258,314 different ways.

To avoid this, we sub-sample our synthetic examples to ensure that they are proportional to the input examples X, specifically they are bounded to $\alpha|X|$ where α is tuned for each dataset. Also, as seen in Table 2.3, our knowledge-guided generators are more likely to generate *neutral* examples than any other class. To make sure that the labels are not skewed, we also sub-sample the examples to ensure that our generated examples have the same class distribution as the input batch. The SciTail dataset only contains two classes: *entails* mapped to $\sqsubseteq$ and *neutral* mapped to 人. As a result, generated examples that do not belong to these two classes are ignored.

The sub-sampling, however, has a negative side-effect where our generated examples end up using a small number of lexical relations from the large knowledge bases. On moderate datasets, this would cause the entailment model to potentially just memorize these few lexical relations. Hence, we generate new entailment examples for each mini-batch and update the model parameters based on the training+generated examples in this batch.

The overall example generation procedure goes as follows: For each mini-batch X (1) randomly choose 3 applicable rules per source and sentence (e.g., replacing "men" with "people" based on PPDB in premise is one rule), (2) produce examples Z_{all} using $\mathbb{G}^{\text{KB}}$, $\mathbb{G}^{\text{H}}$ and $\mathbb{G}^{\text{s2s}}$, (3) randomly sub-select examples Z from Z_{all} to ensure the balance between classes and $|Z| = \alpha|X|$.

2.4.3 Model Training

Figure 2.6 shows the complete architecture of our model, AdvEntuRe (ADVersarial training for textual ENTailment Using Rule-based Examples.). The entailment model $\mathbb{D}$ is shown with the white box and two proposed generators are shown using black boxes. We combine the two symbolic untrained generators, $\mathbb{G}^{\text{KB}}$ and $\mathbb{G}^{\text{H}}$ into a single $\mathbb{G}^{\text{rule}}$ model. We combine the generated adversarial examples Z with the original training examples X to train the discriminator. Next, we describe how the individual models are trained and finally present our new approach to train the generator based on the discriminator's performance.

Discriminator Training

We use one of the state-of-the-art entailment models (at the time of its publication) on SNLI, decomposable attention model (Parikh et al., 2016) with intra-sentence attention as our discriminator $\mathbb{D}$. The model attends each word in hypothesis with each word in the premise, compares each pair of the attentions, and then aggregates them as a final representation. This discriminator model can be easily replaced with any other entailment model without any other change to the AdvEntuRe architecture. We pre-train our discriminator $\mathbb{D}$ on the original dataset, X=(P, H, C) using:

$$\mathbb{D}(X;\theta) = \arg\max_{\hat{C}} \mathbb{D}(\hat{C}|P,H;\theta) \tag{2.4}$$

$$\hat{\theta} = \arg\min_{\theta} L(C, \mathbb{D}(X;\theta)) \tag{2.5}$$

where L is cross-entropy loss function between the true labels, Y and the predicted classes, and $\hat{\theta}$ are the learned parameters.

Generator Training

Our knowledge-guided and hand-defined generators are symbolic parameter-less methods which are not currently trained. For simplicity, we will refer to the set of symbolic rule-based generators as $\mathbb{G}^{\text{rule}} := \mathbb{G}^{\text{KB}} \cup \mathbb{G}^{\text{H}}$. The neural generator $\mathbb{G}^{\text{s2s}}$, on the other hand, can be trained as described earlier. We leave the training of the symbolic models for future work.

Adversarial Training

We now present our approach to iteratively train the discriminator and generator in a GAN-style framework. Unlike traditional GAN (Goodfellow et al., 2014) on image/text generation that aims to obtain better generators, our goal is to build a robust discriminator regularized by the generators ($\mathbb{G}^{\text{s2s}}$ and $\mathbb{G}^{\text{rule}}$). The discriminator and generator are iteratively trained against each other to achieve better discrimination on the augmented data from the generator and better example generation against the learned discriminator. Algorithm 1 shows our training procedure.

Algorithm 1 Training procedure for AdvEntuRe.

```
pretrain discriminator 𝔻(θ̂) on X;
pretrain generators 𝔾_c^s2s(φ̂) on X;
for number of training iterations do
  for mini-batch B ← X do
    generate examples from 𝔾
      Z_G ⇐ 𝔾(B; φ),
    balance X and Z_G s.t. |Z_G| ≤ α|X|
    optimize discriminator:
      θ̂ = arg min_θ L_𝔻(X + Z_G; θ)
    optimize generator:
      φ̂ = arg min_φ L_𝔾s2s(Z_G; L_𝔻; φ)
    Update θ ← θ̂; φ ← φ̂
  end for
end for
```

First, we pre-train the discriminator $\mathbb{D}$ and the seq2seq generators $\mathbb{G}^{\text{s2s}}$ on the original data X. We alternate the training of the discriminator and generators over K iterations (set to 30 in our experiments).

For each iteration, we take a mini-batch B from our original data X. For each mini-batch, we generate new entailment examples, Z_G using our adversarial examples generator. Once we collect all the generated examples, we balance the examples based on their source and label (as described in Section 2.4.2). In each training iteration, we optimize the discriminator against the augmented training data, $X + Z_G$ and use the discriminator loss to guide the generator to pick challenging examples. For every mini-batch of examples $X + Z_G$, we compute the discriminator loss $L(C; \mathbb{D}(X + Z_G; \theta))$ and apply the negative of this loss to each word of the generated sentence in $\mathbb{G}^{\text{s2s}}$. In other words, the discriminator loss value replaces the cross-entropy loss used to train the seq2seq model (similar to a REINFORCE (Williams, 1992a) reward).

2.4.4 Results

Our empirical assessment focuses on two key questions: (a) Can a handful of rule templates improve a state-of-the-art entailment system, especially with moderate amounts of training data? (b) Can iterative GAN-style training lead to an improved discriminator?

To this end, we assess various models on the two entailment **datasets** mentioned earlier: SNLI (570K examples) and SciTail (27K examples).[5] To test our hypothesis that adversarial example based training prevents overfitting in small to moderate training data regimes, we compare model accuracies on the test sets when using 1%, 10%, 50%, and 100% subsamples of the train and dev sets.

We consider two baseline **models**: $\mathbb{D}$, the Decomposable Attention model (Parikh et al., 2016) with intra-sentence attention using pre-trained word embeddings (Pennington et al., 2014); and $\mathbb{D}_{\text{retro}}$ which extends $\mathbb{D}$ with word embeddings initialized by retrofitted vectors (Faruqui et al., 2015). The vectors are retrofitted on PPDB, WordNet, FrameNet, and all of these, with the best results for each dataset reported here.

Our proposed model, AdvEntuRe, is evaluated in three flavors: $\mathbb{D}$ augmented with examples generated by $\mathbb{G}^{\text{rule}}$, $\mathbb{G}^{\text{s2s}}$, or both, where $\mathbb{G}^{\text{rule}} = \mathbb{G}^{\text{KB}} \cup \mathbb{G}^{\text{H}}$. In the first two cases, we create new examples for each batch in every epoch using a fixed generator (cf. Section 2.4.2). In the third case ($\mathbb{D} + \mathbb{G}^{\text{rule}} + \mathbb{G}^{\text{s2s}}$), we use the GAN-style training.

We uses grid search to find the best hyper-parameters for $\mathbb{D}$ based on the validation set: hidden size 200 for LSTM layer, embedding size 300, dropout ratio 0.2, and fine-tuned embeddings.

The ratio between the number of generated vs. original examples, α is empirically chosen to be 1.0 for SNLI and 0.5 for SciTail, based on validation set performance. Generally, very few generated examples (small α) has little impact, while too many of them overwhelm the original dataset resulting in worse scores (cf. Appendix for more details).

Results

Table 2.4 summarizes the test set accuracies of the different models using various subsampling ratios for SNLI and SciTail training data.

We make a few observations. First, $\mathbb{D}_{\text{retro}}$ is ineffective or even detrimental in most cases, except on SciTail when 1% (235 examples) or 10% (2.3K examples) of the training data is used. The gain in these two cases is likely because retrofitted lexical rules are helpful with extremely less data training while not as data size increases.

On the other hand, our method always achieves the best result compared to the baselines ($\mathbb{D}$ and $\mathbb{D}_{\text{retro}}$). Especially, significant improvements are made in less data setting: +2.77% in SNLI (1%) and 9.18% in SciTail (1%). Moreover, $\mathbb{D} + \mathbb{G}^{\text{rule}}$'s accuracy on SciTail (100%) also outperforms the previous state-of-the-art model (DGEM (Khot et al., 2018), which achieves 77.3%) for that dataset by 1.7%.

Among the three different generators combined with $\mathbb{D}$, both $\mathbb{G}^{\text{rule}}$ and $\mathbb{G}^{\text{s2s}}$ are useful in SciTail, while $\mathbb{G}^{\text{rule}}$ is much more useful than $\mathbb{G}^{\text{s2s}}$ on SNLI. We hypothesize that seq2seq model trained on large training sets such as SNLI will be able to reproduce the input sentences. Adversarial examples from such a model are not useful since the entailment model uses the same training examples. However, on smaller sets, the seq2seq model would introduce noise that can improve the robustness of the model.

[5] SNLI has a 96.4%/1.7%/1.7% split and SciTail has a 87.3%/4.8%/7.8% split on train, valid, and test sets, resp.

Table 2.4: Test accuracies with different subsampling ratios on SNLI (top) and SciTail (bottom).

SNLI	1%	10%	50%	100%
$\mathbb{D}$	57.68	75.03	82.77	84.52
$\mathbb{D}_{\text{retro}}$	57.04	73.45	81.18	84.14
AdvEntuRe				
⌞ $\mathbb{D}$ + $\mathbb{G}^{\text{s2s}}$	58.35	75.66	82.91	**84.68**
⌞ $\mathbb{D}$ + $\mathbb{G}^{\text{rule}}$	**60.45**	**77.11**	**83.51**	84.40
⌞ $\mathbb{D}$ + $\mathbb{G}^{\text{rule}}$ + $\mathbb{G}^{\text{s2s}}$	59.33	76.03	83.02	83.25

SciTail	1%	10%	50%	100%
$\mathbb{D}$	56.60	60.84	73.24	74.29
$\mathbb{D}_{\text{retro}}$	59.75	67.99	69.05	72.63
AdvEntuRe				
⌞ $\mathbb{D}$ + $\mathbb{G}^{\text{s2s}}$	**65.78**	**70.77**	74.68	76.92
⌞ $\mathbb{D}$ + $\mathbb{G}^{\text{rule}}$	61.74	66.53	73.99	**79.03**
⌞ $\mathbb{D}$ + $\mathbb{G}^{\text{rule}}$ + $\mathbb{G}^{\text{s2s}}$	63.28	66.78	**74.77**	78.60

Ablation Study

To evaluate the impact of each generator, we perform ablation tests against each symbolic generator in $\mathbb{D}$ + $\mathbb{G}^{\text{rule}}$ and the generator $\mathbb{G}_c^{\text{s2s}}$ for each entailment class c. We use a 5% sample of SNLI and a 10% sample of SciTail. The results are summarized in Table 2.5.

Interestingly, while PPDB (phrasal paraphrases) helps the most (**+3.6%**) on SNLI, simple negation rules help significantly (**+8.2%**) on SciTail dataset. Since most entailment examples in SNLI are minor rewrites by Turkers, PPDB often contains these simple paraphrases. For SciTail, the sentences are authored independently with limited gains from simple paraphrasing. However, a model trained on only 10% of the dataset (2.3K examples) would end up learning a model relying on purely word overlap. We believe that the simple negation examples introduce *neutral* examples with high lexical overlap, forcing the model to find a more informative signal.

On the other hand, using all classes for $\mathbb{G}^{\text{s2s}}$ results in the best performance, supporting the effectiveness of the GAN framework for penalizing or rewarding generated sentences based on $\mathbb{D}$'s loss. Preferential selection of rules within the GAN framework remains a promising direction.

Qualitative Results

Table 2.6 shows examples generated by various methods in AdvEntuRe. As shown, both seq2seq and rule based generators produce reasonable sentences according to classes and rules. As expected, seq2seq models trained on very few examples generate noisy sentences. The quality of our knowledge-guided generators, on the other hand, does not depend on the training set size and they still produce reliable sentences.

Table 2.5: Test accuracies across various rules $\mathcal{R}$ and classes C. Since SciTail has two classes, we only report results on two classes of $\mathbb{G}^{\text{s2s}}$

	$\mathcal{R}/C$	SNLI (5%)	SciTail (10%)
$\mathbb{D}+\mathbb{G}^{\text{rule}}$	$\mathbb{D}$	69.18	60.84
	+ PPDB	**72.81** (**+3.6%**)	65.52 (+4.6%)
	+ SICK	71.32 (+2.1%)	67.49 (+6.5%)
	+ WordNet	71.54 (+2.3%)	64.67 (+3.8%)
	+ HAND	71.15 (+1.9%)	**69.05** (**+8.2%**)
	+ all	71.31 (+2.1%)	64.16 (+3.3%)
$\mathbb{D}+\mathbb{G}^{\text{s2s}}$	$\mathbb{D}$	69.18	60.84
	+ positive	71.21 (+2.0%)	67.49 (+6.6%)
	+ negative	71.76 (+2.6%)	68.95 (+8.1%)
	+ neutral	71.72 (+2.5%)	-
	+ all	**72.28** (**+3.1%**)	**70.77** (**+9.9%**)

Case Study: Negation

For further analysis of the negation-based generator in Table 2.1, we collect only the negation examples in test set of SNLI, henceforth referred to as nega-SNLI. Specifically, we extract examples where either the premise or the hypothesis contains "not", "no", "never", or a word that ends with "n't'. These do not cover more subtle ways of expressing negation such as "seldom" and the use of antonyms. nega-SNLI contains 201 examples with the following label distribution: 51 (25.4%) neutral, 42 (20.9%) entails, 108 (53.7%) contradicts. Table 2.7 shows examples in each category.

While $\mathbb{D}$ achieves an accuracy of only 76.64%[6] on nega-SNLI, $\mathbb{D}$ + $\mathbb{G}^{\text{H}}$ with NEGATE is substantially more successful (+6.1%) at handling negation, achieving an accuracy of 82.74%.

2.4.5 Conclusion

We introduced an adversarial training architecture for textual entailment. Our seq2seq and knowledge-guided example generators, trained in an end-to-end fashion, can be used to make any base entailment model more robust. The effectiveness of this approach is demonstrated by the significant improvement it achieves on both SNLI and SciTail, especially in the low to medium data regimes.

We believe that the amount of improvements gained from the knowledge will increase if more knowledge bases are incorporated. For the future work, if the rule selection part could be trainable (e.g., giving reward signals if specific rules give more information to learn), the rule-based GAN architecture could take more advantages from the learning.

[6]This is much less than the full test accuracy of 84.52%.

Table 2.6: Given a premise **P** (underlined), examples of hypothesis sentences **H'** generated by seq2seq generators $\mathbb{G}^{\text{s2s}}$, and premise sentences **P'** generated by rule based generators $\mathbb{G}^{\text{rule}}$, on the full SNLI data. Replaced words or phrases are shown in **bold**.

P	a person on a horse jumps over a broken down airplane
H': $\mathbb{G}^{\text{s2s}}_{c=\sqsubseteq}$	a person is on a horse jumps over a rail, a person jumping over a plane
H': $\mathbb{G}^{\text{s2s}}_{c=人}$	a person is riding a horse in a field with a dog in a red coat
H': $\mathbb{G}^{\text{s2s}}_{c=\#}$	a person is in a blue dog is in a park
P (or **H**)	a dirt bike rider catches some air going off a large hill
P': $\mathbb{G}^{\text{KB(PPDB)}}_{\rho=\equiv,g_\rho=\sqsubseteq}$	a dirt **motorcycle** rider catches some air going off a large hill
P': $\mathbb{G}^{\text{KB(SICK)}}_{\rho=c,g_\rho=\#}$	a dirt bike **man on yellow bike** catches some air going off a large hill
P': $\mathbb{G}^{\text{KB(WordNet)}}_{\rho=syno,g_\rho=\sqsubseteq}$	a dirt bike rider catches some **atmosphere** going off a large hill
P': $\mathbb{G}^{\text{Hand}}_{\rho=\text{NEG},g_\rho=人}$	a dirt bike rider **do not catch** some air going off a large hill

Table 2.7: Negation examples in nega-SNLI

$\sqsubseteq$	P: several women are playing volleyball. H: this doesn't look like soccer.
#	P: a man with no shirt on is performing with a baton. H: a man is trying his best at the national championship of baton.
人	P: island native fishermen reeling in their nets after a long day's work. H: the men did not go to work today but instead played bridge.

2.5 Embedding-driven NSL: Geometry Retrofitting

2.5.1 Introduction

A distributed representation (or word embedding vector) maps a word into a fixed-dimensional vector. The vectors can be trained by supervised models (Turney, 2012, 2013), unsupervised models (Mikolov et al., 2013; Pennington et al., 2014), or recently pretrained language models (Peters et al., 2018b; Devlin et al., 2019). The context-awareness of the models makes the vectors include linguistic regularities such as morphology or syntax (Mikolov and Dean, 2013).

It is a surprising result that some word pairs in the same relation have exhibited a linear relationship in the continuous embedding space. This intuition is that the embedding space is somehow 'linear' or 'equally dense', so that geometric distance and orientation in one part of it have the same relative effects and transformational implications everywhere else. We call this the *geometric property*. However, that this should be true is neither obvious nor true in general.

In fact, Levy and Goldberg (2014) observe the importance of geometric attributes. For example, given two words pairs (a='Women', a^*='Men') and (b='Queen', b^*='King'), the analogy

test is to find the closest $\hat{b}$ with b^*. They found that the linear offset method with `similar`($\hat{b}$, a^*-a+b) outperforms `similar`($\hat{b}$-b, a^*-a) on the analogy test, where the latter model ignores geometry between the two pairs. This supports that geometric properties (i.e., directions and spatial distances) are important attributes in capturing relational semantics. Linzen (2016) also points out when the offset of relation (i.e., a-a^*) is too small, then the offset model finds the nearest neighbor of b instead. While the prior analyses are limited to pairwise analogies, we focus on exploring the inherent properties of many relations.

Spatial geometry of word vectors is a still under-explored area except for a few recent attempts: Mimno and Thompson (2017) analyze geometric properties between word vectors and context vectors during the training process and find that they are geometrically opposed to each other instead of being simply determined by semantic similarity. McGregor et al. (2017) find a correlation of geometric features to the identification of semantic type coercion task, which is limited to a specific task.

In this work, we conduct an in-depth analysis on the spatial geometric properties of word vectors as well as their application to various NLP tasks. Our work has following contributions:

- provides a quantitative and qualitative analysis of geometric regularities of existing word vectors with various relation datasets (e.g., SemEval) and the dependencies between similar/different relations. We observe that only a few types of relation (e.g., Cause:Purpose) have strong geometric tendencies in current word embeddings (e.g., GloVe).
- proposes an optimization technique called *geofitting* to make word pairs from the same relation type have similar geometric properties. Our new vectors outperform the original vectors and retrofitting vectors (Faruqui et al., 2015) on various NLP tasks such as movie review classification and textual entailment.

In §2.5.2 and §2.5.3, we define and analyze geometric properties in current word embedding vectors. We propose a new method *geofitting* to incorporate geometric properties into the vectors in §2.5.4, and describe its effectiveness in various NLP applications in §2.5.5.

2.5.2 Geometric Properties and Regularities

We first define the geometric properties of a relation using a set of word vectors from the relation and provide a statistical measurement to calculate their regularity. For example, if the geometric values of all word pairs in a relation type are different from word pair values of other relations, this implies that the relation type has its unique geometric properties. In this paper, we define the geometric property (or tendency) of a word pair as *slope* and *distance* of word vectors in the pair.

Figure 2.7 shows two pairs of words that have the "gender" relation : the red pair (a, a^*) is ('men', 'women') and the blue pair (b, b^*) is ('king', 'queen'). If their geometric tendency, `distance` and `slope`, is similar, the relation has strong regularity. The distance and slope of a word pair (a, a^*) are calculated as follows:

$$\texttt{dist}(a, a^*) = \sqrt{\sum_d (a^d - a^{*d})^2}$$

$$\texttt{slope}(a, a^*) = degree\Big(\arccos\langle \frac{a - a^* \cdot o}{|a - a^*||o|} \rangle\Big) \tag{2.6}$$

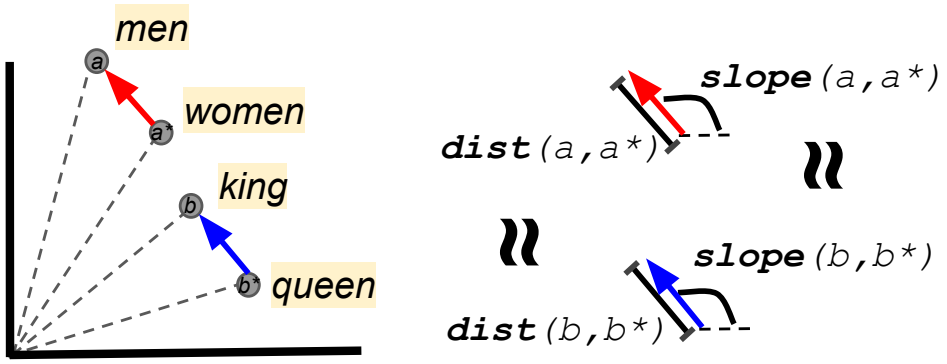


Figure 2.7: Two word pairs from a "gender" relation. We define a relation vector as an offset of two word vectors in a pair. Relation vectors are characterized as two geometric properties: `distance` and `slope`. If they are similar, the relation has a strong regularity.

where d is the dth dimension of vector a or a^*, o is a randomly assigned vector to measure the angle of $a - a^*$, and *degree* converts radians to degrees. Note that arccos returns radians in $[0, \pi)$, therefore *degree* is bounded by $[0, 180)$.

In a generalized form, given a relation type $r \in \boldsymbol{R}$, we have N pairs of two words: $((w_1, w_1^*)...(w_N, w_N^*)) \in \boldsymbol{w}_r$ where $\boldsymbol{w}$ is a set of word pairs and $\boldsymbol{R}$ is all relations in the dataset. Using Eq 2.6, distance and slope of a relation r are calculated using the standard deviation[7]:

$$\sigma_{\texttt{dist}}(r) = \sqrt{\frac{\sum_{i=1,w\in\boldsymbol{w}_r}^{N}(\texttt{dist}(w_i, w_i^*) - \mu_{\texttt{dist}})^2}{N-1}}$$
$$\sigma_{\texttt{slope}}(r) = \sqrt{\frac{\sum_{i=1,w\in\boldsymbol{w}_r}^{N}(\texttt{slope}(w_i, w_i^*) - \mu_{\texttt{slope}})^2}{N-1}} \tag{2.7}$$

where $\mu_{\texttt{dist}}(r) = \frac{\sum_{i=1,w\in\boldsymbol{w}_r}^{N}\texttt{dist}(w_i,w_i^*)}{N}$ and $\mu_{\texttt{slope}}(r) = \frac{\sum_{i=1,w\in\boldsymbol{w}_r}^{N}\texttt{slope}(w_i,w_i^*)}{N}$ are the means of distance and slope, respectively. Finally, we define the regularity `Reg` of the relation r as a multiplication of the two geometric variances.

$$\texttt{Reg}(r) = \sigma_{\texttt{dist}}(r) * \sigma_{\texttt{slope}}(r) \tag{2.8}$$

This indicates how much the geometric properties in the same relation type differ from each other: lower variance means higher consistency or regularity between the word pairs of the relation. The proposed measures are simple but effective to check consistency of word pairs in a relation.

2.5.3 Preliminary Analysis

With proposed measures in Eqs 2.6 and 2.8, we conduct an in-depth analysis of how different types of relations have unique properties and how they are related. We first describe relation

[7]We simply call σ as 'variance' in the following sections.

datasets we use in our analysis and then provide qualitative and quantitative observations on their regularities.

Relation Datasets

We use three relation datasets: `SemEval`, `Google`, and `Freebase`. Each dataset differs in the number of relation types, granularity of relations, and scales. Tables 2.8 and 2.9 are examples of relations and their statistics, respectively. Full lists can be found in Appendix. We first describe how we preprocess each dataset.

Table 2.8: Example relations in `SemEval`, `Freebase`, and `Google`. *L*1 relations are more abstract than *L*2.

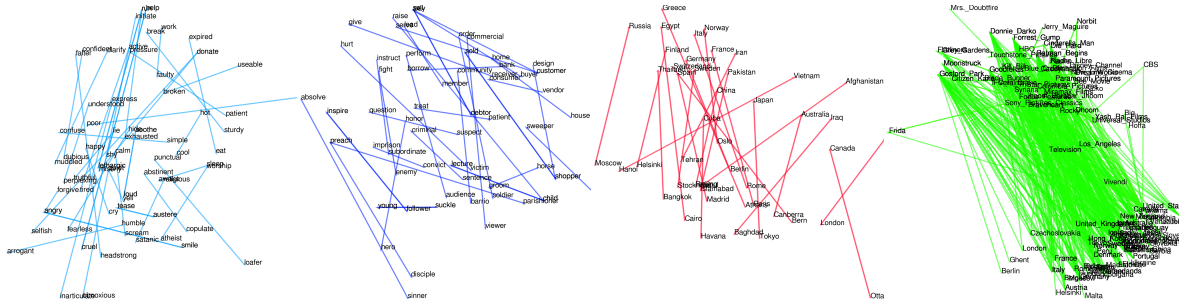
	Level-1 (L1) / Level-2 (L2)	**Examples**
SemEv.	PART-WHOLE / Mass:Portion	car:engine, face:nose
	CAUSE-PURPOSE / Cause:Effect	enigma:puzzlement
Goog.	- / capital-world	Algeria:Oran
	- / gram1-adj2adverb	amazing:amazingly
Freeb.	- / MUSIC.GENRE.parent_genre	punk_rocker:glam_rock
	- / FILM.FILM.produced_by	Die_Hard:Bruce_Willis

Table 2.9: Number of relations, pairs and types.

	`SemEval`	`google`	`Freebase`
number of relations (*L1* / *L2*)	10 / 79	14	28
number of word pairs	3,285	573	2,272
number of word types	2,983	905	1,287

`SemEval` (Jurgens et al., 2012) is a task, given two pairs of words, to determine the degree of semantic relatedness between the two pairs. `Google` is Mikolov et al. (2013)'s analogy dataset. We use a distinct set of word-pairs for each relation type. It has 14 types of relations; 5 knowledge relations and 9 morphological relations. `Freebase` contains a subset of the original Freebase dump (Bollacker et al., 2008) from Lin et al. (2015b), including about 40,000 triples. While relations in `SemEval` are linguistic types (e.g., syntactic or morphological), relations in `Freebase` are about knowledge (e.g., a film produced by). We only choose relations with three hops (e.g., location.location.contains) where each hop refers to a domain, type, and predicate. Relations whose frequency is less than ten are filtered out.

To obtain an embedding vector for each word, we map words in the pairs into a vocabulary of word embedding vectors such as W2V (Mikolov and Dean, 2013) or GloVe (Pennington et al., 2014). For bigram words, we concatenate two words with an underscore.



(a) NONATTR/A-gent:Obj **(b)** CASE / Action:Recip **(c)** capital-countries **(d)** FILM.FILM.distributed

Figure 2.8: PCA projection of word pairs : `SemEval` (a,b), `Google` (c) and `Freebase` (d). The red and green relations seem to have stronger geometric regularities than other relations. Best viewed in color.

Qualitative Observation

In this section, we provide qualitative evidence of whether geometric regularities of relations exist in the current word embedding vectors. If word pairs of the same relation have a similar geometric tendency, they should look parallel in direction and align with the same distance. Our first analysis is to show and check these tendencies visually.

We reduce the 300-dimensional GloVe word vectors into the first two principal dimensions using Principal Component Analysis (PCA) following Mikolov et al. (2013)[8]. Figure 2.8 shows PCA projections of some relations in the relation datasets. We scatter all words in each type of relation with a straight line between two words in a relation. Each relation type is mapped with a different color. This visualization shows that knowledge relations (e.g., (c) capital-countries in `Google`) have stronger geometric tendencies than linguistic relations (e.g., (a) Agent/Object in `SemEval`). More projection results can be found in the Appendix.

Quantitative Observation

Regularities: We measure the variance of two geometric properties using Eqs 2.7 and 2.8. Table 2.10 shows three types of the relations with the highest ($\triangle$) and lowest ($\triangledown$) `Reg`(r).

In `SemEval`, Cause-Purpose and Case Relation have the lowest variance, indicating that word pairs of causality or case patterns often co-occur and are easily captured from the context. Meanwhile, Contrast (e.g., before:after) or Similarity (e.g., trackpad:mouse) are difficult to identify because word embeddings themselves are tightly clustered together and lose the geometric patterns, as argued by Linzen (2016). As Levy and Goldberg (2014) point out, relations have multiple attributional sub-relations (e.g., king:queen pair has a gender attribute as well as a royalty). That is, without considering *all* the attributes of a word pair, it is hard to observe perfect regularities.

`Google` has relatively less variance than `SemEval` because the relation types in `Google` are

[8]Since linearity of data is vital to observe geometry, we do not use t-SNE (Maaten and Hinton, 2008) which is a non-linear transformation. All vectors are length-normalized.

Table 2.10: Relation types with highest (△) and lowest (▽) variance sorted by Reg in Eq 2.8 with GloVe. Variances from 1,000 randomly-assigned word pairs are on the first row for the comparison. The lower variance the stronger geometric regularities.

		σ_{slope}	σ_{dist}	Reg($\sigma_{s*}\sigma_{d}$)
	random	3.23	0.07	0.23
SemEval	△ SIMILAR,DimenSimilar	3.09	0.19	0.59
	△ SIMILAR,Coordinate	3.70	0.20	0.74
	△ CONTRAST,Direction	3.52	0.23	0.81
	▽CAUSE-PUR,Agent:Goal	2.35	0.10	0.24
	▽CASE REL,Obj:Instr	2.71	0.09	0.24
	▽CASE REL,Age:Instr	2.63	0.10	0.26
Google	△ gram6-nationality-adj	2.55	0.13	0.33
	△ capital-world	3.00	0.11	0.33
	△ gram1-adj2adv	3.64	0.10	0.36
	▽currency	2.86	0.06	0.17
	▽gram4-superlative	2.62	0.08	0.21
	▽gram8-plural	2.18	0.10	0.22
Freebase	△ business.company.child	3.78	0.17	0.64
	△ loca.loca.adjoin	3.03	0.22	0.67
	△ loca.adjoin.adjoin	3.03	0.22	0.67
	▽busin.board.member	0.97	0.06	0.06
	▽busin.company.founders	1.29	0.05	0.06
	▽busin.board.org	2.04	0.05	0.10

about knowledge which often occurs in text and relatively simpler than the types in Freebase and SemEval. Interestingly, many of knowledge type relations in Freebase have strong regularities except a few such as adjoining locations that share a border or child companies.

We assign 1,000 random word pairs which have no relation between two words in a pair as a comparison group. Compared to the random word pairs, Semeval relations always have higher regularity Reg, whereas Google and Freebase have lower regularity Reg. This indicates that current word embeddings do not capture geometric regularities except a few types.

Unique Properties and Dependencies: The next question is to find whether each type of relation has unique geometric properties and how they are related (i.e., similar relations are close to each other). Figure 2.9 shows a distribution of relations in $\boldsymbol{R}$ with respect to the averaged distance $\mu_{dist}(r)$ (y-axis) and the averaged slope $\mu_{slope}(r)$ (x-axis). We use a number+letter index for each $L2$ relation in SemEval due to space limitations, following the naming convention in the original dataset: $L1$ number and $L2$ letter. Please find the mapping table of the names in Appendix.

In SemEval, we observe a few number of small size clusters of $L2$ relations under the same $L1$ relation; For example, there is a small group <Taxonomic (1a), Singular Collective (1c), Plural Collective (1d)> from CLASS-INCLUSION (1), another small group <Attribute Noncondition (6b),

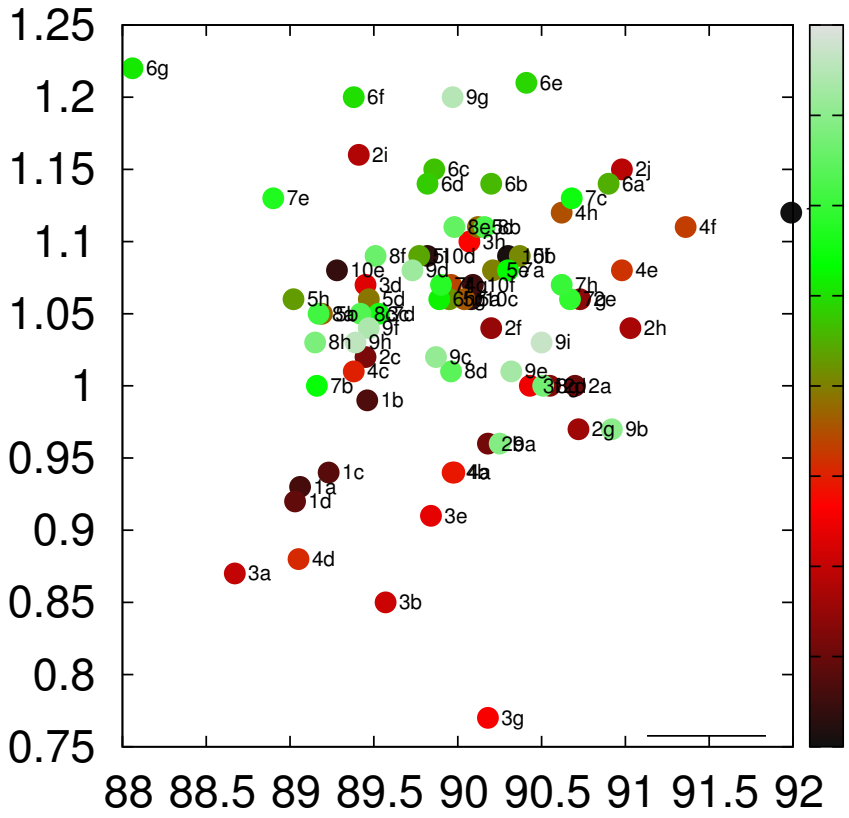


Figure 2.9: Distribution of relations with their averaged slope (x-axis) and distance (y-axis) on `SemEval` with GloVe. The gradient colors (from black, red to green) are mapped to relation types sorted by name, for better understanding their dependencies. Only few $L1$ relations (e.g., 1) have a coherent cluster of its child $L2$ relations (e.g., 1a, 1c, 1d). Best viewed in color.

Object Nonstate (6c), Attribute Nonstate (6d)> under NON-ATTRIBUTE (6), and others. We find that SIMILARITY: Synonymity (3a) and CONTRAST: Directional (4d) are very close to each other because synonyms or antonyms are designed to be close and mixed up together. In general, however, many other types of relations are not coherently clustered, making it difficult to capture their unique geometric features or dependencies between sub-relations from the same $L1$ relation.

Given the observation that only a few types of relations have unique geometric regularities and dependencies, we could say that current word embeddings themselves do not include relational semantics with respect to geometry of the vectors. The appendix includes more graphs with a different combination of relation levels, dataset, or embeddings.

2.5.4 Proposed Method

We conjecture that regular geometric properties are desirable. They would enable inference after less training and generally decrease the representational complexity of the space. Therefore we ask: can we somehow encourage greater geometric regularity? We to prove our conjecture, explore the applicability of geometric properties to various NLP tasks.

Many recent neural network models use pre-trained word embeddings for initializing word vectors for their tasks. If the word embeddings contain geometric regularities for each relation, some NLP tasks that require relational regularity in their inference may take advantage from the relational semantics. We describe our objective function and learning process called *geofitting*.

Objective Function

The objective function constrains word pairs in the same relation type to have similar geometric values. Our objective is to learn new word vectors $\boldsymbol{x}$ throughout the learning procedure. Let's suppose $\boldsymbol{x}_{r,i} = (x_i, x_i^*)$ is an ith word pair of two new word vectors from the relation type $r \in \boldsymbol{R}$. Each pair of new word vectors corresponds to the original word pair (w_i, w_i^*) in $\boldsymbol{w}_r$ where $\boldsymbol{w}_r = ((w_1, w_1^*), \ldots, (w_N, w_N^*))$. Our goal is to (1) learn the new vectors $x \in \boldsymbol{x}$ as similar as possible to the original word vector $w \in \boldsymbol{w}$ and (2) also constrain all word pairs $((x_1, x_1^*), \ldots, (x_N, x_N^*))$ in $\boldsymbol{x}_r$ to have similar geometric properties.

This objective function is motivated by retrofitting (Faruqui et al., 2015). While retrofitting balances semantic synonymy with ontological organization, here we inject regularity over other relation types, too. The objective Ψ is to minimize the pair-wise loss of the distance and slope for each pair in the word pair set:

$$\begin{aligned}\Psi(\boldsymbol{R}, \boldsymbol{w}; \boldsymbol{x}) = \sum_{r \in \boldsymbol{R}} \sum_{i=1}^{N} \Big[& \alpha_r (\|x_i - w_i\|^2 + \|x_i^* - w_i^*\|^2) \\ & + \frac{\beta_r}{(N-1)} \{ \sum_{j=1, i \neq j}^{N} \left| \|x_i - x_i^*\|^2 - \|x_j - x_j^*\|^2 \right| \} \\ & + \frac{\gamma_r}{(N-1)} \{ \sum_{j=1, i \neq j}^{N} 1 - sim(x_i - x_i^*, x_j - x_j^*) \} \Big] \end{aligned} \tag{2.9}$$

where N is the number of word pairs in r and $sim(x_i - x_i^*, x_j - x_j^*)$ is a cosine of the angle between two offset vectors. Note that for the second and third terms, there is a third summation to find another word pair (x_j, x_j^*) from the set of relation $\boldsymbol{w}_r$ so we divide them by $N - 1$, the number of possible j. The time complexity of Ψ is $O(N^2 R)$. α, β and γ are weighting values for each term. The optimal weighting terms are empirically found by choosing the best model from cross-validation for each type of relation r.

Learning

We use gradient descent to learn our objective function. To find the optimal vector $\hat{x}$, we take the gradient from the objective function and update it with a learning rate ρ. $x_i^{(t+1)} \leftarrow x_i^{(t)} - \rho \frac{\partial \Psi}{\partial x_i}$. $x_i^{(t)}$ is an updated value at the tth epoch of training. We also update its paired word x^* in the same manner, just switching between x and x^* in the gradient update. Since the derivative is given in a closed form, it is guaranteed that gradient descent finds the optimal point $\hat{x} \sim x^T$ as training

continues. The derivative $\frac{\partial \Psi}{\partial x_i}$ is as follows:

$$\frac{\partial \Psi}{\partial x_i} = 2\alpha_i(x_i - x_i^*) + \frac{2\beta_i(x_i - x_i^*)}{N-1} \sum_{j=1, i\neq j}^{N} \texttt{sgn}(\|x_i - x_i^*\|^2 - \|x_j - x_j^*\|^2)$$
$$- \frac{\gamma_i}{N-1} \sum_{j=1, i\neq j}^{N} \left(\frac{x_j - x_j^*}{\|x_i - x_i^*\|\|x_j - x_j^*\|} - \frac{x_i(x_i - x_i^* \cdot x_j - x_j^*)}{\|x_i - x_i^*\|^3 \|x_j - x_j^*\|}\right) \tag{2.10}$$

where sgn is a sign function for a real number x, which is +1, 0, and −1 if $x > 0$, $x = 0$ and $x < 0$, respectively. A detailed algorithm is in Appendix. Our code including the whole learning procedure will be publicly released upon acceptance.

2.5.5 Results

We validate two hypotheses in the experiment[9]: (1) Does geofitting properly make word vectors in same relation type have more-similar geometric properties? (2) Are these geofitted vectors extrinsically useful in NLP tasks?

Sanity Checking

Learning Individual Parameter. We first conduct a sanity check of how new word vectors $\boldsymbol{x}$ look after geofitting against the original vectors $\boldsymbol{w}$. To check the learning effect from each term in Ψ clearly, we individually optimize each term, distance (β) for (a) and slope (γ) for (b) in Figure 2.10[10]. Variances (σ_{dist}, σ_{slope}) decrease substantially, from 0.15 to 0.01 and 3.51 to 3.38, respectively. Also, the PCA projections visually show that geofitted vectors hold the similar distances (a) and slopes (b).

Table 2.11: Variances between $\boldsymbol{R}$ and $\boldsymbol{R}^{geo}$ vectors with GloVe. All variances are macro-averaged by *L2* relations. + and $*$ mean that variances are significantly lower ($p < 0.1$) than those from randomly assigned word pair vectors and the original vectors, respectively. Mean-based permutation tests are used for the statistical significance.

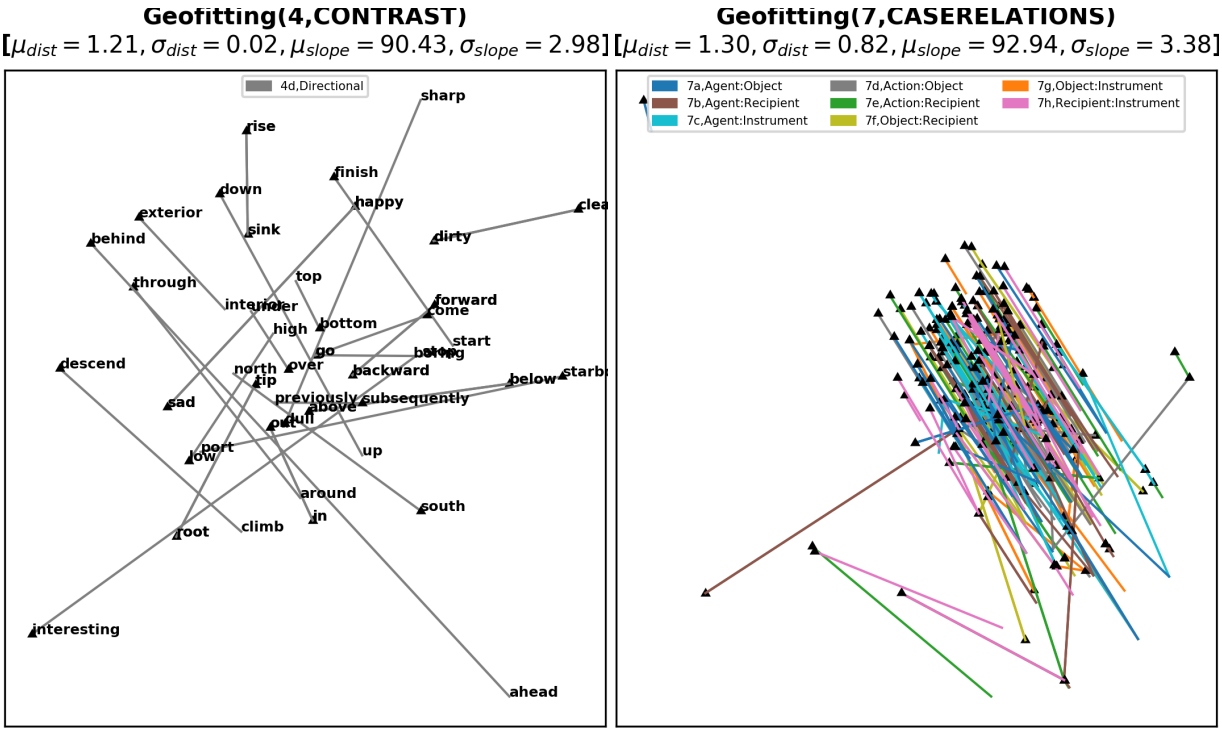
	$\sigma_{slope}(\boldsymbol{R})$	$\sigma_{slope}(\boldsymbol{R}^{geo})$	$\sigma_{dist}(\boldsymbol{R})$	$\sigma_{dist}(\boldsymbol{R}^{geo})$
SemEval	3.05	2.73	0.15	0.04^{+*}
Google	2.59	0.66^{+*}	0.11	0.01^{+*}
Freebase	2.85	2.24	0.07	0.03^{+*}

Regularities and Dependencies. We compare overall mean variances of slope $\frac{1}{N_{rel}} \sum_{r\in\boldsymbol{R}} \sigma_{slope}(r)$ and distance $\frac{1}{N_{rel}} \sum_{r\in\boldsymbol{R}} \sigma_{dist}(r)$[11] across different relation types between $\boldsymbol{R}$ and $\boldsymbol{R}^{geo}$ in Table 2.11.

[9] A detailed experiment setting is in Appendix.

[10] We also checked that α correctly makes $\boldsymbol{x}$ similar as $\boldsymbol{w}$.

[11] N_{rel} = number of relations in arbitrary relation dataset



(a) σ_{dist} (0.15 → 0.01)

(b) σ_{slope} (3.51 → 3.38)

Figure 2.10: PCA projection after geofitting of two `SemEval` relations. We only update a single term in Ψ, distance (β) (left) or slope (γ) (right), while other terms keep zero. Variances inside the parentheses decrease after geofitting. PCA results before geofitting are in Appendix.

Mean variances of both slope and distance in $\boldsymbol{R}^{geo}$ decrease compared to $\boldsymbol{R}$. In addition, we provide a statistical significance using the mean-based permutation test Baker (1995). Macro-averaged p-values are significantly small except for σ_{slope} in `SemEval` and `Freebase`. For further details, see the Appendix.

Figure 2.11 shows PCA projections from original to geofitted word vectors with `SemEval` *L2* relations. All lines are centered to the origin point for better understanding the effect of geofitting. Different colors of lines represent each *L2* relation types. The lines in geofitted vectors are more clustered and have similar distances by color than those in original vectors.

Figure 2.12 shows the dependency across averaged slope and distance of the geofitted vectors, which corresponds to Figure 2.9 of original vectors. If our learning process works properly, then (1) each point in Figure 2.12 is more separable than the points in Figure 2.9, and (2) points under the same *L1* relation type should be clustered more in Figure 2.12 than those in Figure 2.9. We measure the Euclidean distances between relations to check them. In table 2.12, positive values on the first row mean that the averaged distance and slope after *geofitting* is more distant than the original word vectors. On the other hand, negative value shows that *L2* relations under the same *L1* in `SemEval` are closer to each other after *geofitting*.

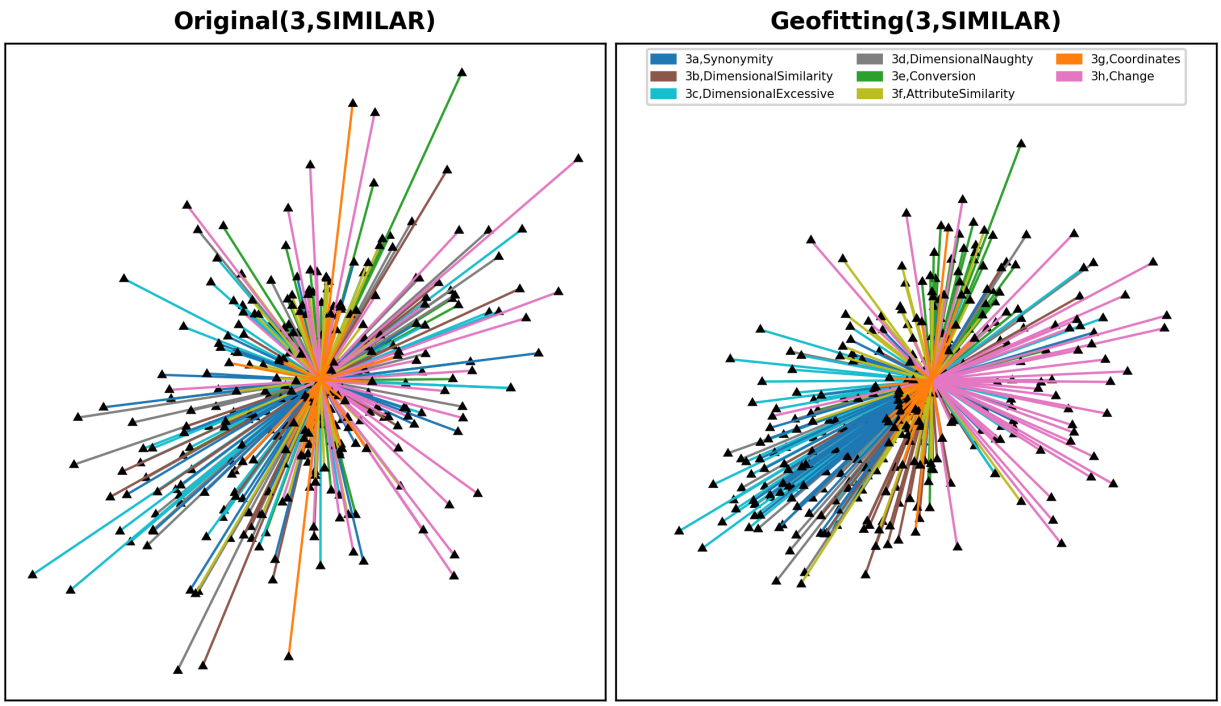


Figure 2.11: Centered view of PCA projection before (left) and after (right) geofitting on `SemEval` with GloVe. Best viewed in color.

Applicability of Geometry to NLP Tasks

Word embeddings are often evaluated by extrinsic (Faruqui et al., 2015) or intrinsic (Tsvetkov et al., 2015) methods. To validate practical applicability of our geofitted vectors, we conduct extrinsic comparison among original (i.e., GloVe, W2V), retrofitted (Faruqui et al., 2015) and geofitted word vectors on different NLP tasks. For retrofitting, we use PPDB (Ganitkevitch et al., 2013), WordNet (Miller, 1995), FrameNet (Baker et al., 1998) and all of three as external lexical databases. For geofitting, we use `SemEval` (*L1*/*L2*), `Google`, `Freebase`, and all of three as external relational databases (See Tables 2.8 and 2.9).

Tasks and Models. We test our word vectors on various NLP tasks: textual entailment with SNLI (Bowman et al., 2015) or SciTail (Khot et al., 2018) dataset; semantic relatedness on SICK dataset (Marelli et al., 2014a); paraphrase detection on the Microsoft Research Paraphrase Corpus (MRPC) (Dolan et al., 2004); classification benchmarks such as movie review sentiment (MR) (Pang and Lee, 2005) and customer product reviews (CR) (Hu and Liu, 2004); subjectivity classification (SUBJ) (Pang and Lee, 2004); opinion polarity (MPQA) (Wiebe et al., 2005); and question-type classification (TREC) (Li and Roth, 2002). We average word embeddings by sentence and train logistic regression to predict the label. For textual entailment, we train Decomposable Attention (Parikh et al., 2016) with only 10% of the training data but test it on full testing data to check how much relational knowledge is helpful in the less-data setting. We use

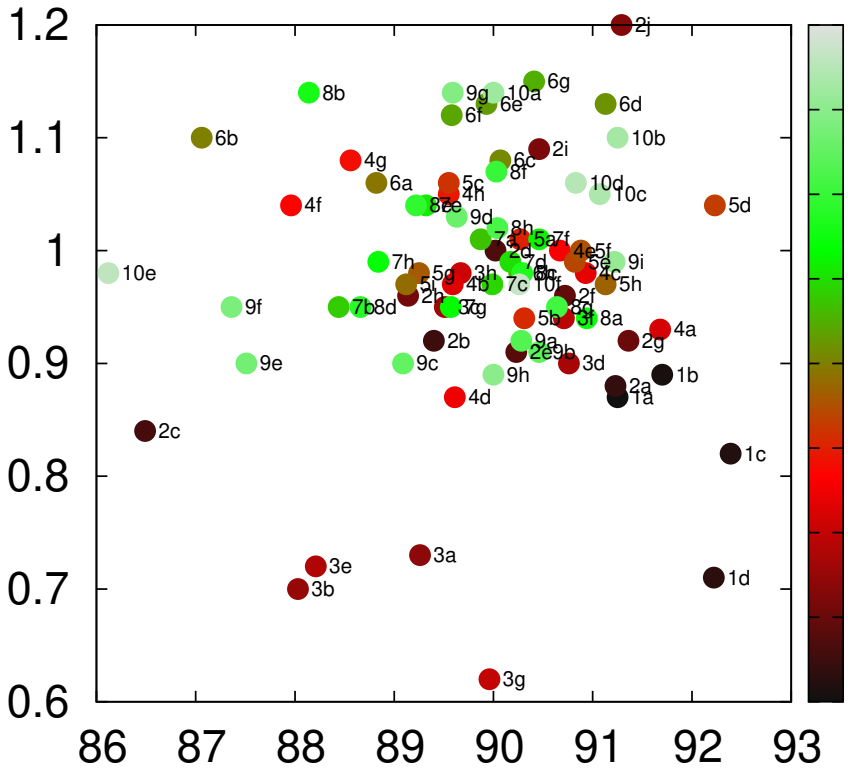


Figure 2.12: Dependency of relations after geofitting (`SemEval`, GloVe). Best viewed in color.

10-fold cross-validation for MR, CR, SUBJ, and MPQA to tune L2 penalty, while TREC has a pre-defined data split. We use Pearson's *r* for SICK, F1 for MRPC, and accuracy for other tasks.

Results. Table 2.13 shows performance comparisons of each application. The best score among different word embeddings (e.g., GloVe, W2V) is reported. Overall, our geofitted vectors outperform the others on six out of nine tasks and achieve the second best on other two tasks. However, neither retrofitting nor geofitting takes a benefit on SNLI, which indicates that the dataset itself does not require much lexical or relational semantics (Gururangan et al., 2018c). Among the relation datasets, `Google` shows the most effectiveness for classification tasks.

Ablation Figure 2.13 shows ablation tests result of which types of relations in geofitted vectors are most useful for each NLP tasks. Every bar represents the difference of accuracy (or F1 for MRPC) rates between the original and the geofitted word vectors. Colors on the chart differ from the types of relations. In general, using all types of the relations outperforms applying only a single relation type, except for the MRPC task in both `SemEval` and `Google`. Even in some tasks (i.e., CR and SUBJ), using one type of relation to train the geofitted vectors leads to worse performance than the original vectors. If geometric properties from only one type of relation are injected, this tends to lose multi-attributional properties of the relation (Levy and Goldberg, 2014), which word vectors should have. This kind of distortion of word vectors can be alleviated when various relation types are considered at the same time, as each word vector tries to find

Table 2.12: Differences of averaged Euclidean distances between original and geofitted vectors. It is calculated across *L1* relations (top) and across SemEval *L2* relations under the same *L1* relation (bottom). The former: more positive, the stronger unique geometric property, while the latter: more negative, the closer between similar *L2* relations. Slopes and distances are normalized.

	SemEval	Google	Freebase
across *L1* relations	0.083	0.007	0.017
across *L2* under the same *L1*	-0.051	-	-

Table 2.13: Results on NLP tasks between original (i.e., GloVe or W2V), retrofitted, and geofitted word vectors. All scores are averaged by 10 times of runs. The **best** and second best scores are bold and underlined, respectively.

		MRPC	SICK	TREC	MPQA	MR	CR	SUBJ	$\text{SNLI}_{10\%}$	$\text{SciTail}_{10\%}$
	Original	81.88	**79.17**	85.20	88.37	77.85	80.03	91.87	**76.37**	61.79
Retrofit	+ PPDB	81.60	77.82	85.00	**88.78**	77.92	80.51	91.97	73.68	58.37
	+ WordNet	81.87	76.90	83.60	88.36	77.72	80.14	91.88	73.25	61.49
	+ FrameNet	81.57	75.88	83.40	87.73	77.03	79.69	92.07	73.16	59.78
	+ All	81.66	77.56	82.40	88.57	77.77	80.22	91.6	73.42	53.33
Geofit	+ SemEval1	**82.13**	78.41	85.20	88.50	77.76	80.51	91.98	72.61	62.05
	+ SemEval2	82.07	78.26	85.00	88.31	77.73	80.40	92.08	73.80	**62.95**
	+ Google	82.00	77.89	**86.40**	88.51	77.86	**80.83**	**92.09**	72.99	57.91
	+ Freebase	81.92	78.13	85.60	88.52	**78.06**	80.56	92.04	73.25	59.83
	+ All	81.86	77.89	85.60	88.40	77.80	80.40	91.83	73.61	54.79

the optimal point among multiple relational properties. In Google, injecting at least one type of the relations has still positive effect on predicting correct labels for MRPC and TREC, whereas SemEval has an useful influence only on MRPC.

Qualitative Analysis Table 2.14 shows a few cases when only geofitted word vectors predict an actual label for each task. Bold words on second and last columns are the words of which geometric properties are injected. Word pairs in relation dataset help make a better representation of the words. In the first two examples in MR and SICK, semantically opposing relations such as "likely"-"unlikely" or "homeless"-"money" help separate them each other for the prediction, leading to making a correct prediction. Another two examples in SUBJ and TREC show how multiple attributes affect their paired word. Meaning of "happy" or "color" is more concretized by injecting their attributional relations via geofitting. This specification of meaning by relations yields correct predictions.

Table 2.14: Predictions across different embeddings. **T** is a True label. **O**, **R**, and **G** are predicted labels by original, retrofitting, and geofitting embeddings, respectively. The last column shows word pairs in `SemEval` or `Google` that contain the words in the input text. Relation names of `SemEval` are shortened. More examples are in Appendix.

	Input text for each task	**T**	**O**	**R**	**G**	**Word pairs used** (format: relation(A,B))
MR	here , adrian lyne comes as close to profundity as he is **likely** to get .	1	0	0	1	gram2-opposite(**likely**,unlikely)
SICK	'There is no man cutting a box', 'A **homeless** man is holding up a sign and is begging for **money**'	1.1	2.9	2.7	1.7	9g(**homeless**,poor), 3c(**money**,rich) 8h(**money**,poverty), 2i(poor,**money**)
SUBJ	so she writes it using info from people who **talk** about him and writes an unflattering piece , which doesn't make him **happy** .	1	0	0	1	3d(**talk**,gossip), 5f(**happy**,smile), 8h(**happy**,sad) 5g(laugh,**happy**), 6f(**happy**,cry) 6d(**happy**,heartbroken), 4g(**happy**,depressed)...
TREC	What **color** is indigo ?	4	5	5	4	1c(**color**,green), 1a(**color**,red) 3h(darken,**color**), 6a(**color**,invisible)

2.5.6 Conclusion

We propose a simple but effective way of characterizing relations in continuous space using geometric properties. Our analyses show that only a few types of relation have own geometric regularities and dependencies between similar relations even if most of them do not. We show how to incorporate the geometric properties of relations into the word embeddings to achieve new word embeddings that are geometrically regular for each relation type. Our geofitted embeddings outperform the baseline models on various NLP tasks.

2.6 Model-driven NSL: Neural-Symbolic Module Integration

2.6.1 Introduction

Textual entailment, a key challenge in natural language understanding, is a sub-problem in many end tasks such as question answering and information extraction. In one of the earliest works on entailment, the PASCAL Recognizing Textual Entailment Challenge, Dagan et al. (2005) define entailment as follows: text (or premise) P entails a hypothesis H if *typically* a human reading P would infer that H is *most likely* true. They note that this informal definition is "based on (and assumes) common human understanding of language as well as *common background knowledge*".

While current entailment systems have achieved impressive performance by focusing on the language understanding aspect, these systems, especially recent neural models (e.g. Parikh et al.,

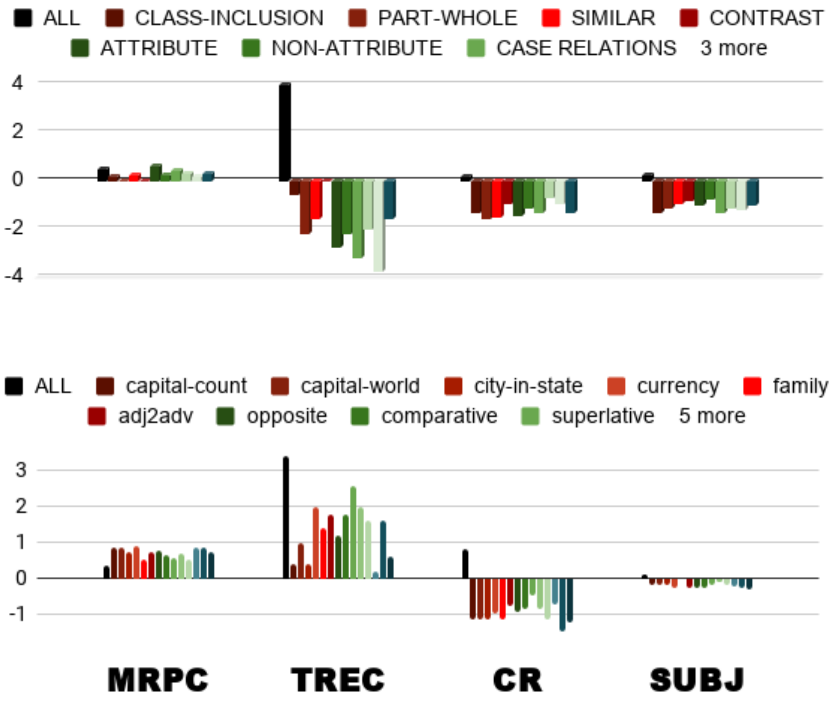


Figure 2.13: Ablation between relation types and tasks: `SemEval` (top) and `Google` (bottom) relations on four different tasks such as MRPC, TREC, CR, and SUBJ. Y-axis shows performance difference between geofitted and original vectors. The positive difference, the better performance in geofitted vectors against original vectors. Best viewed in color.

2016; Khot et al., 2018), do not directly address the need for filling knowledge gaps by leveraging common background knowledge.

P: The aorta is a large blood vessel that moves blood away from the heart to the rest of the body.

H (entailed): Aorta is the major artery carrying recently oxygenated blood away from the heart.

H' (not entailed): Aorta is the major vein carrying recently oxygenated blood away from the heart.

Figure 2.14: Knowledge gap: Aorta is a major artery (not a vein). *Large blood vessel* soft-aligns with *major artery* but also with *major vein*.

Figure 2.14 illustrates an example of P and H from SciTail, a recent science entailment dataset (Khot et al., 2018), that highlights the challenge of *knowledge gaps*—sub-facts of H that aren't stated in P but are universally true. In this example, an entailment system that is strong at filling lexical gaps may align *large blood vessel* with *major artery* to help conclude that P entails H. Such a system, however, would equally well—but incorrectly—conclude that P entails

a hypothetical variant H' of H where *artery* is replaced with *vein*. A typical human, on the other hand, could bring to bear a piece of background knowledge, that aorta is a major artery (not a vein), to break the tie.

Motivated by this observation, we propose a new entailment model that combines the strengths of the latest neural entailment models with a structured knowledge base (KB) lookup module to bridge such knowledge gaps. To enable KB lookup, we use a fact-level decomposition of the hypothesis, and verify each resulting sub-fact against both the premise (using a standard entailment model) and against the KB (using a structured scorer). The predictions from these two modules are combined using a multi-layer "aggregator" network. Our system, called NSnet, achieves 77.9% accuracy on SciTail, substantially improving over the baseline neural entailment model, and comparable to the structured entailment model proposed by Khot et al. (2018).

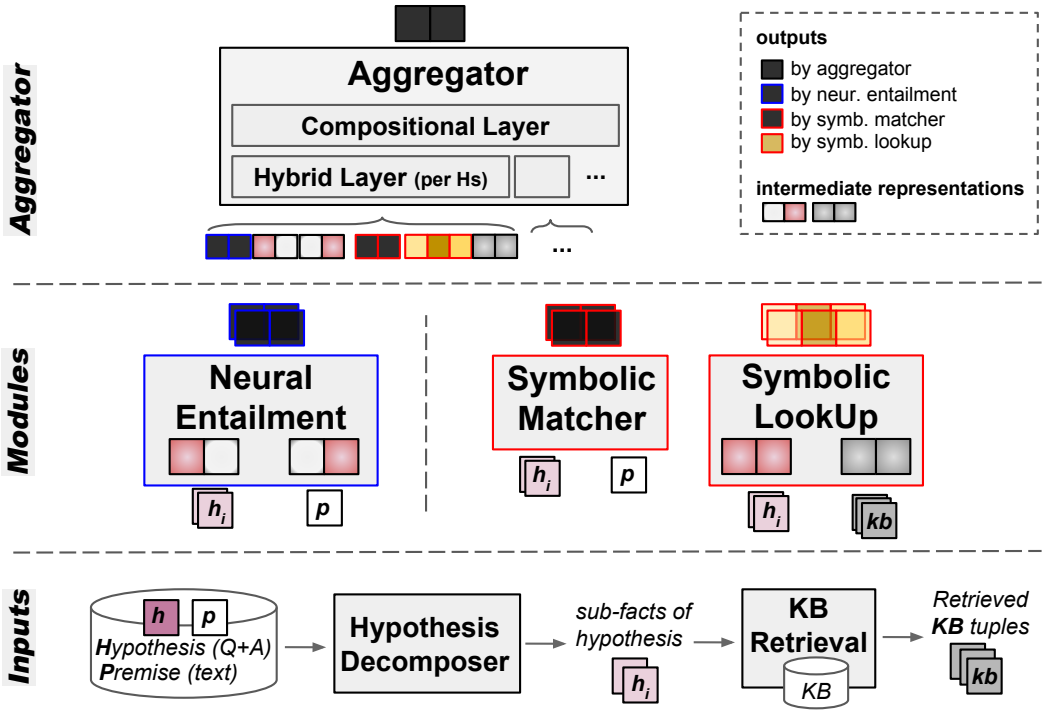


Figure 2.15: Neural-symbolic learning in NSnet. The bottom layer has QA and their supporting text in SciTail, and the knowledge base (KB). The middle layer has three modules: Neural Entailment (blue) and Symbolic Matcher and Symbolic Lookup (red). The top layer takes the outputs (black and yellow) and intermediate representation from the middle modules, and hierarchically trains with the final labels. All modules and aggregator are jointly trained in an end-to-end fashion.

2.6.2 Proposed Model

A general solution for combining neural and symbolic modules remains a challenging open problem. As a step towards this, we present a system in the context of neural entailment that demon-

strates a successful integration of the KB lookup model and simple overlap measures, opening up a path to achieve a similar integration in other models and tasks. The overall system architecture of our neural-symbolic model for textual entailment is presented in Figure 2.15. We describe each layer of this architecture in more detail in the following sub-sections.

Inputs

We decompose the hypothesis and identify relevant KB facts in the bottom "inputs" layer (Fig. 2.15).

Hypothesis Decomposition: To identify knowledge gaps, we must first identify the facts stated in the hypothesis $h = (h_1, h_2..)$. We use ClausIE (Del Corro and Gemulla, 2013) to break h into sub-facts. ClausIE tuples need not be verb-mediated and generate multiple tuples derived from conjunctions, leading to higher recall than alternatives such as Open IE (Banko et al., 2007). [12]

Knowledge Base (KB): To verify these facts, we use the largest available clean knowledge base for the science domain (Dalvi et al., 2017), with 294K simple facts, as input to our system. The knowledge base contains subject-verb-object (SVO) tuples with short, one or two word arguments (e.g., hydrogen; is; element). Using these simple facts ensures that the KB is only used to fill the basic knowledge gaps and not directly prove the hypothesis irrespective of the premise.

KB Retrieval: The large number of tuples in the knowledge base makes it infeasible to evaluate each hypothesis sub-fact against the entire KB. Hence, we retrieve the top-100 relevant knowledge tuples, K', for each sub-fact based on a simple Jaccard word overlap score.

Modules

We use a Neural Entailment model to compute the entailment score based on the premise, as well as two symbolic models, Symbolic Matcher and Symbolic Lookup, to compute entailment scores based on the premise and the KB respectively (middle layer in Fig. 2.15).

Neural Entailment We use a simple neural entailment model, Decomposable Attention (Parikh et al., 2016), one of the state-of-the-art models on the SNLI entailment dataset (Bowman et al., 2015). However, our architecture can just as easily use any other neural entailment model. We initialize the model parameters by training it on the Science Entailment dataset. Given the sub-facts from the hypothesis, we use this model to compute an entailment score $n(h_i, p)$ from the premise to each sub-fact h_i.

[12] While prior work on question answering in the science domain has successfully used Open IE to extract facts from sentences (Khot et al., 2017), one of the key reasons for errors was the lossy nature of Open IE.

Symbolic Matcher In our initial experiments, we noticed that the neural entailment models would often either get distracted by similar words in the distributional space (false positives) or completely miss an exact mention of h_i in a long premise (false negatives). To mitigate these errors, we define a Symbolic Matcher model that compares exact words in h_i and p, via a simple asymmetric bag-of-words overlap score:

$$m(h_i, p) = \frac{|h_i \cap p|}{|p|}$$

One could instead use more complex symbolic alignment methods such as integer linear programming (Khashabi et al., 2016; Khot et al., 2017).

Symbolic Lookup This module verifies the presence of the hypothesis sub-fact h_i in the retrieved KB tuples K', by comparing the sub-fact to each tuple and taking the maximum score. Each field in the KB tuple kb_j is scored against the corresponding field in h_i (e.g., subject to subject) and averaged across the fields. To compare a field, we use a simple word-overlap based Jaccard similarity score, $Sim(a, b) = \frac{|a \cap b|}{|a \cup b|}$. The lookup match score for the entire sub-fact and kb-fact is:

$$Sim_f(h_i, kb_j) = \left(\sum_k Sim(h_i[k], kb_j[k]) \right) / 3$$

and the final lookup module score for h_i is:

$$l(h_i) = \max_{kb_j \in K'} Sim_f(h_i, kb_j)$$

Note that the Symbolic Lookup module assesses whether a sub-fact of H is universally true. Neural models, via embeddings, are quite strong at mediating between P and H. The goal of the KB lookup module is to complement this strength, by verifying universally true sub-facts of H that may not be stated in P (e.g. "aorta is a major artery" in our motivating example).

Aggregator Network

For each sub-fact h_i, we now have three scores: $n(h_i, p)$ from the neural model, $m(h_i, p)$ from the symbolic matcher, and $l(h_i)$ from the symbolic lookup model. The task of the Aggregator network is to combine these to produce a single entailment score. However, we found that using only the final predictions from the three modules was not effective. Inspired by recent work on skip/highway connections (He et al., 2016; Srivastava et al., 2015), we supplement these scores with intermediate, higher-dimensional representations from two of the modules.

From the Symbolic Lookup model, we use the representation of each sub-fact $h_i^{enc} = Enc(h_i)$ obtained by averaging word embeddings (Pennington et al., 2014) and individual similarity scores over the top-100 KB tuples $emb_i = [\ldots, Sim_f(h_i, kb_j), \ldots]$. From the neural entailment model, we use the intermediate representation of both the sub-fact of hypothesis and premise text from the final layer (before the softmax computation), $n^v(h_i, p) = [v_1; v_2]$.

We define a hybrid layer that takes as input a simple concatenation of these representation vectors from the different modules:

$$in(h_i, p) = [h_i^{enc}; l(h_i); m(h_i, p); n(h_i, p)emb_i; n^v(h_i, p)]$$

The hybrid layer is a single layer MLP for each sub-fact h_i that outputs a sub-representation $out_i = MLP(in(h_i, p))$. A compositional layer then uses a two-layer MLP over a concatenation of the hybrid layer outputs from different sub-facts, $\{h_1, \ldots, h_I\}$, to produce the final label,

$$label = \text{MLP}([out_1; out_2; \cdots out_I])$$

Finally, we use the cross-entropy loss to train the Aggregator network jointly with representations in the neural entailment and symbolic lookup models, in an end-to-end fashion. We refer to this entire architecture as the NSnet network.

To assess the effectiveness of the aggregator network, we also use a simpler baseline model, Ensemble, that works as follows. For each sub-fact h_i, it combines the predictions from each model using a probabilistic-OR function, assuming the model score P_m as a probability of entailment. This function computes the probability of at least one model predicting that h_i is entailed, i.e. $P(h_i) = 1 - \Pi_m(1 - P_m)$ where $m \in n(h_i, p), m(h_i, p), l(h_i)$. We average the probabilities from all the facts to get the final entailment probability.[13]

Table 2.15: Entailment accuracies on the SciTail dataset. NSnet substantially improves upon its base model and marginally outperforms DGEM.

Entailment Model	Valid.	Test
Majority classifier	63.3	60.3
DecompAttn (Base model)	73.1	74.3
DecompAttn + HypDecomp	71.8	72.7
DGEM	**79.6**	77.3
Ensemble (this work)	75.2	74.8
NSnet (this work)	77.4	**77.9**

2.6.3 Results

We use the SciTail **dataset**[14] (Khot et al., 2018) for our experiments, which contains 27K entailment examples with a 87.3%/4.8%/7.8% train/dev/test split. The premise and hypothesis in each example are natural sentences authored independently as well as independent of the entailment task, which makes the dataset particularly challenging. We focused mainly on the SciTail dataset, since other crowd-sourced datasets, large enough for training, contained limited linguistic variation (Gururangan et al., 2018a) leading to limited gains achievable via external knowledge.

[13] While more intuitive, performing an AND aggregation resulted in worse performance.

[14] `http://data.allenai.org/scitail`

For **background knowledge**, we use version v4 of the aforementioned Aristo TupleKB[15] (Dalvi et al., 2017), containing 283K basic science facts. We compare our proposed models to Decomposed Graph Entailment Model (DGEM) (Khot et al., 2018) and Decomposable Attention Model (DecompAttn) (Parikh et al., 2016).

Table 2.16: Ablation: Both Symbolic Lookup and Symbolic Matcher have significant impact on NSnet performance.

	Valid.	**Test**
NSnet	77.39	77.94
- Symbolic Matcher	76.46	74.73 (**-3.21%**)
- Symbolic Lookup	75.95	75.80 (**-2.14%**)
- Both	75.10	73.98 (**-3.96%**)

Results

Table 2.15 summarizes the validation and test accuracies of various models on the SciTail dataset. The DecompAttn model achieves 74.3% on the test set but drops by 1.6% when the hypotheses are decomposed. The ENSEMBLE approach uses the same hypothesis decomposition and is able to recover 2.1% points by using the KB. The end-to-end NSnet network is able to further improve the score by 3.1% and is statistically significantly (at p-value 0.05) better than the baseline neural entailment model. The model is marginally better than DGEM, a graph-based entailment model proposed by the authors of the SciTail dataset We show significant gains over our base entailment model by using an external knowledge base, which are comparable to the gains achieved by DGEM through the use of hypothesis structure. These are orthogonal approaches and one could replace the base DecompAttn model with DGEM or more recent models (Tay et al., 2017; Yin et al., 2018).

In Table 2.16, we evaluate the impact of the Symbolic Matcher and Symbolic Lookup module on the best reported model. As we see, removing the symbolic matcher, despite its simplicity, results in a 3.2% drop. Also, the KB lookup model is able to fill some knowledge gaps, contributing 2.1% to the final score. Together, these symbolic matching models contribute 4% to the overall score.

Qualitative Analysis

Table 2.17 shows few randomly selected examples in test set. The first two examples show cases when the symbolic models help to change the neural alignment's prediction (F) to correct prediction (T) by our proposed ENSEMBLE or NSnet models. The third question shows a case where the NSnet architecture learns a better combination of the neural and symbolic methods to correctly identify the entailment relation while ENSEMBLE fails to do so.

[15] `http://data.allenai.org/tuple-kb`

Table 2.17: Few randomly selected examples in the test set between symbolic only, neural only, ENSEMBLE and NSnet inference. The symbolic only model shows its the most similar knowledge from knowledge base inside parenthesis. The first two example shows when knowledge helps fill the gap where neural model can't. The third example shows when NSnet predicts correctly while ENSEMBLE fails.

Premise: plant cells possess a cell wall , animals never . **Hypothesis**: a cell wall is found in a plant cell but not in an animal cell .				
Sub-fact of hypothesis	**neural only**	**symbolic only**	**ENSEMBLE**	**NSnet**
a cell wall is found in a plant cell but not in an animal cell	F(0.47)	T(0.07) (cell is located in animal)	T(0.50)	-
Prediction (true label: **T** (entail))	**F**	**T**	**T**	**T**

Premise: the pupil is a hole in the iris that allows light into the eye . **Hypothesis**: the pupil of the eye allows light to enter .				
Sub-fact of hypothesis	**neural only**	**symbolic only**	**ENSEMBLE**	**NSnet**
the pupil of the eye allows light to enter	F(0.43)	T(0.12), (light enter eye)	T(0.50)	-
Prediction (true label: **T** (entail))	**F**	**T**	**T**	**T**

Premise: binary fission in various single-celled organisms (left) . **Hypothesis**: binary fission is a form of cell division in prokaryotic organisms that produces identical offspring .				
Sub-facts of hypothesis	**neural only**	**symbolic only**	**ENSEMBLE**	**NSnet**
binary fission is a form of cell division in prokaryotic organisms	F(0.49)	T(0.07) (binary fission involve division)	T(0.52)	-
binary fission is a form	F(0.63)	T(0.1) (phase undergo binary fission)	T(0.66)	-
a form of cell division in prokaryotic organisms produces identical offspring	F(0.46)	T(0.05) (cell division occur in tissue)	T(0.48)	-
Prediction (true label: **T** (entail))	**F**	**T**	**F**	**T**

2.6.4 Conclusion

We proposed a new entailment model that attempts to bridge knowledge gaps in textual entailment by incorporating structured knowledge base lookup into standard neural entailment models. Our architecture, NSnet, can be trained end-to-end, and achieves a 5% improvement on SciTail over the baseline neural model used here. The methodology can be easily applied to more complex entailment models (e.g., DGEM) as the base neural entailment model. Accurately identify-

ing the sub-facts from a hypothesis is a challenging task in itself, especially when dealing with negation. Improvements to the fact decomposition should further help improve the model.

2.7 Conclusion

Our proposed architecture of combining symbolic modules (or knowledge) and neural modules (or knowledge) shows significant improvements in various textual entailment or classification tasks. Particularly in the low to medium data regimes, the adversarial augmentation of symbolic knowledge achieves much gains over the neural-only or symbolic-only systems. The symbolic knowledge can also used to align geometry of word pairs and make them geometrically regular, showing improvements on general NLP downstream tasks. Lastly, aggregating modules from the neural and symbolic systems and training them as an end-to-end fashion helps achieve additional gain over the separated training of the modules.

The integration of neural-symbolic systems shows lexical generalization from large amounts of text as well as explicit knowledge power and reasoning capabilities. In a semantic perspective, this helps bridge the gap between lexical and distributional semantics. In an NLG perspective, this helps fill the information gap of two connotations between the speaker and the listener in their communication, particularly on question answering type of conversation. Lastly in a socio-olinguistic perspective, such NLG systems with symbolic and neural capabilities would be able to produce factual, logical, and knowledge-augmented text, making the communication more trustful and productive.

Chapter 3

Coherently-Structured Generation

In this chapter, we incorporated the *structure* facet into language generation. As in the previous chapter, the other facets, such as knowledge and styles, remain fixed so we can focus on studying the effect of the structure facet individually.

When the content of the topic is decided, one has to represent the information in a manner which guides the structure of the (multi-sentence) communication in a coherent way, which is essentially a planning process. We call this process the *structure* facet. The structure here could be setting a specific goal for generating text, ordering the content, or coherently connecting multiple sentences.

"How" can the model compose multiple sentences and string them together coherently, as in telling a story? Every part of a text plays a role in the whole, and the text is coherent ONLY if the reader can assemble the complete picture correctly by putting each piece in its correct place. A system that randomly throws together facts, even if they are all true, will not produce a coherent text (See the machine-generated text in Table 1.2). To ensure communication, two functionalities are needed: (1) the delimitation of sentence-sized chunks from the complete total, and (2) their organization, in a linear or subordinate order. This is called *text planning* in the NLG literature.

We first compare various multi-sentence NLG tasks (§3.1), then introduce our *text planning* framework (§3.2) followed by a literature survey on the prior work (§3.3).

3.1 Task: Multi-Sentence Text Generation

In a communication with multiple sentences, one has to decide how to structure the numerous pieces of information. Table 3.1 shows examples of various NLG tasks that have multiple sentences as their output. Here, "multiple" can mean either a sequence of coherent sentences or a set of diverse sentences not intended to be connected.

For example, when generating a recipe, story, or summary of a document, one has to produce multiple sentences to create a coherent body of text. Organizing the sentences requires an understanding of various factors including tense, coreference, communication, goals, plans, and scripts, among others. On the other hand, when responding to an email or writing a review about a manuscript, one has to consider different intents, such as a positive response or negative response, and different writing qualities, like clarity, novelty, or presentation.

Task: **Abstractive textual summarization** Original document: "...there are two ways to become wealthy: to create wealth or to take wealth away from others. The former adds to society. The latter typically subtracts from it, for in the process of taking it away, wealth gets destroyed. A monopolist who overcharges..." Stiglitz, J.E. (2013). The price of inequality. London. Penguin. Summary: "Stiglitz (2013) suggests that creating wealth adds value to society, but that taking away the wealth of others detracts from it. He uses the example of a monopolist..."
Task: **Data-to-text: NBA headline generation** Data: WinTeam: {Name: Nuggest, Score: 125,.. }, WinTeamPoints: {Name: Murray, Score: 22,..}, LostTeam: {Name: Lakers, Score: 116,..}, .. Text: "Murray nets 22 in Nuggets' 125-116 win over Lakers"
Task: **Recipe generation** Prompt: "Tell me how to cook a pasta" Recipe: "Bring a large pot of lightly salted water to a boil. Cook spaghetti in the boiling water, stirring occasionally until cooked; through about 12 minutes. Combine garlic..."
Task: **Story generation or paragraph completion** Prompt: "Every natural text is written in some style." Paragraph: "The style is formed by a complex combination of different stylistic factors, including formality markers, emotions, metaphors, etc. Some factors implicitly reflect the author's personality, while others are explicitly controlled by the author's choices in order to achieve some personal or social goal."
Task: **Email response generation** Email: "How did the meeting go?" Response with positive intent: "It was very fruitful!" Response with negative intent: "It was a waste of time."
Task: **Aspect-specific review generation** Review about clarity aspect: "The paper is difficult to read because it has many typos." Review about novelty aspect: "This paper lacks novelty."

Table 3.1: Examples of various multi-sentence NLG tasks.

To better understand the nature of each task, we summarize the detailed properties of the tasks in Table 3.1. Every NLG task has its input called *context* and output called *target* text. While textual summarization provides the full content of the target text, the computer needs to focus on *abstracting* the information, whereas Data2Text provides only partial information, like a structured database, so has to focus on surface-level realizations. Other multi-sentence NLG tasks, such as email response generation, story generation, and dialogue generation, do not provide any content for the target text, so *content planning* needs to be done first. The content planning here should include content selection, content ordering, and content aggregation.

Depending on how the text is structured, planning can be classified into categories. Email responses and review generation *vertically* plan multiple intents or aspects and output a collection of multiple responses that *diversify* each other. On the other hand, story and recipe generation produce a single output with a *horizontal* sequence of sentences, making them *coherent*. In summary, vertical planning maximizes the diversity of a text, while horizontal planning max-

	Summarization / Data2Text	EmailResponse, ReviewGen	Story/Recipe-Gen	Goal-oriented dialogues
Facets	Structures	Structures, Knowledge, Styles	Structures, Knowledge, Styles	Structures, Knowledge, Styles
Datasets	SummCorpora (Kang et al., 2019d) / RW(Wiseman et al., 2017)	SmartReply(Kannan et al., 2016), PeerRead(Kang et al., 2018b)	Bridging(Kang et al., 2019c), ParMask(Kang and Hovy, 2020)	NegoDial(Lewis et al., 2017), GoRecDial(Kang et al., 2019a)
Evaluation	Auto (RBM)	Auto (RBM), Human	Auto (RBM), Human	Auto (RBM), Goal, Human
Content given Context?Target	Full or Partial C $\supset$ T / C $\subset$ T	None C $\perp$ T	None C $\perp$ T	None C $\perp$ T
Planning	Hierarchical / -	Vertical	Horizontal	Horizontal
Objectives	Abstraction	Diversity	Coherence	Coherence + Goal

Figure 3.1: Multi-sentence NLG tasks. In the evaluation row, RBM refers to automatic evaluation metrics such as ROUGE, BLEU, or METEOR. Some tasks, like summarization, have full or partial context provided, while others, like StoryGen, have no context given, requiring context creation or prediction process. (Context $\{\supset,\subset,\sim,\perp\}$TTarget) shows the relationship between context and target text, where context is a super/sub/equal/independent set compared to the target text.

imizes the coherence of text. In particular, some goal-oriented dialogue tasks require specific *achievement goals*, like whether or not to negotiate, attempt to change one's perspective, or to recommend an appropriate item. For these tasks, dialogue generation needs to maximize *goal* achievement as well as the other objectives.

In this work, we collect new datasets or benchmark corpora, for each category of multi-sentence NLG tasks. For hierarchical planning of textual summarization, Kang et al. (2019d) collected a benchmark corpus of existing summarization datasets for a comprehensive understanding of how summarization datasets and systems are biased. For vertical planning of review generation, Kang et al. (2018b) collected a dataset of academic manuscripts and corresponding peer-review texts. For horizontal planning of story generation, we proposed two new tasks; paragraph bridging (Kang et al., 2019c) or paragraph unmasking (Kang and Hovy, 2020). the goal of both is to coherently complete a paragraph given a short piece of text as a prompt. Lastly, for goal oriented dialogues, we collected human-human conversations about recommending movie items (Kang et al., 2019a), making the dialogue a game with the goal of giving the best advice. The next section has a more detailed discussion about the datasets and corresponding methodologies.

3.2 Proposed Approach: *Text Planning*

Humans often make structural decisions before producing utterances, such as topic introduction, ordering, and conversation strategies. This ensures coherence between utterances and aligns their utterances with the goal of the conversation. We call such structural decisions *plans*. Our proposed mechanism of *text-planning* is a hierarchical generation process of that guides the generation of multiple sentences by matching surface-level realizations with high-level plans. Such plans can be discourse relationships between texts, framing, strategies in goal-oriented dialogues, multiple or single speech acts, topics, and more.

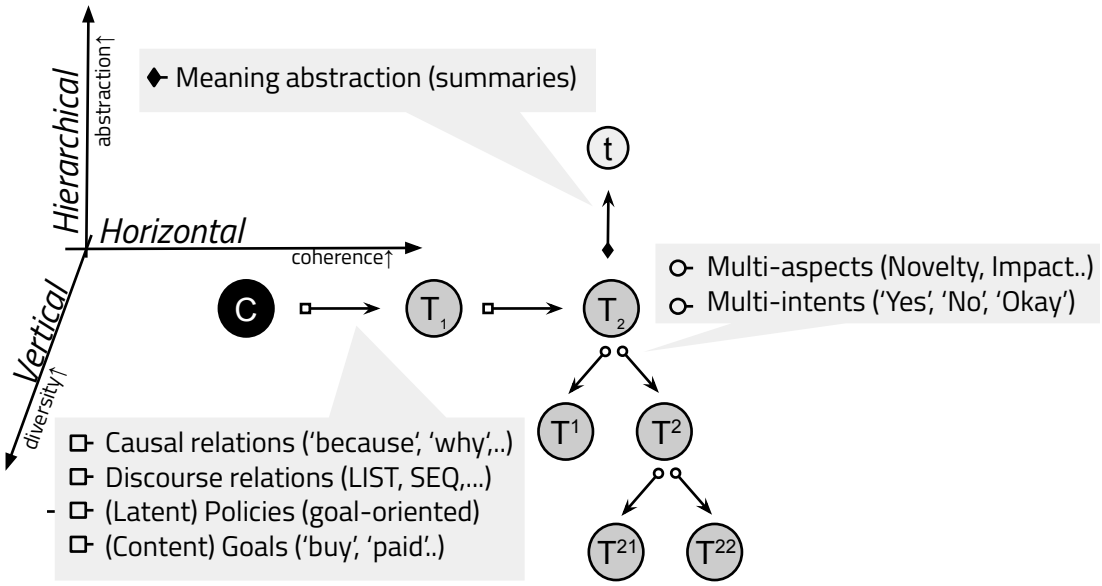


Figure 3.2: Text planning over three dimensions: hierarchical (Kang et al., 2019d), vertical (Kang et al., 2017b, 2019c,a; Kang and Hovy, 2020), and horizontal (Kang et al., 2018b, 2017a). The grey circles represent individual target sentences to be generated, given a contextual text, represented by the dark circle.

As shown in Figure 3.2, we suggest three-dimensional text planning. *Horizontal* planning focuses on producing a coherent long text such as a paragraph or a story, whereas *vertical* planning focuses on producing semantically diverse texts, such as generating disparate reviews about different aspects of the same product. *Hierarchical* planning focuses on abstracting the meaning of multiple sentences into a short summary text.

Planning is a cognitive function commonly used in human language generation. Ideally, the three planning processes should take place simultaneously, as it does in humans. However, working on all three dimensions in parallel, is beyond the scope of this work, even as it remains an important direction for future research. We describe each planning process in detail as follows:

Horizontal Text Planning

We propose four forms of horizontal plans: **causal relations**, **discourse relations**, **content goals**, and **latent policies**. As will be further described in §3.4, Kang et al. (2017b) has produced a chain of causally linked texts to explain the temporal causality between two events in a time-series. The produced chains of text were evaluated by humans as a plausible causal-explanation about the event. Another study (Kang et al., 2019c) (§3.5) extended the causal relation to a general discourse of relationships based on rhetorical structure theory (Mann and Thompson, 1988).

Motivated by script theory (Schank and Abelson, 2013), Kang and Hovy (2020) proposed a text planner that predicts content keywords based on the context. It then guides the surface realizer, which is pre-trained language model like GPT2 (Radford et al., 2019) to generate surface words using copy mechanism according to the planned keywords. Both the causal and discourse relations and content planning improves the coherency of output text on our newly-proposed generation tasks: *paragraph bridging* and *paragraph unmasking*. See §3.6 for details.

The plan can be also represented in a latent form using hierarchical modeling between adjacent sentences (Kang et al., 2019c) or policy learning with bot-playing in machine-machine conversations (Kang et al., 2019a). In particular, Kang et al. (2019a) optimizes the task-specific goal, like recommending the correct movie item, by making two bots communicate with each other. Learning latent strategies, such as conducting longer dialogues, seems to help the bots achieve their goals through communication. See §3.7 for details.

Vertical Text Planning

Due to the lack of appropriate datasets for training and the difficulty of determining success, vertical text planning is an under-explored field, except for some industrial applications like SmartReply (Kannan et al., 2016).

We collected a new dataset, PeerRead (Kang et al., 2018a), comprised of academic papers and their corresponding peer-reviews, from various sources. We analyzed the dataset, which showed interesting trends such as a high correlation between an overall positive recommendation and recommending an oral presentation. Thus, we defined two novel tasks based on the dataset: (1) predicting the acceptance of a paper based on textual features and (2) predicting the score of each aspect of a review based on the paper and review contents. Our experiments showed that certain properties of a paper, such as having an appendix, are correlated with a higher acceptance rate. As a potential future direction, the dataset also suggests a new generation task: **aspect**-specific automatic reviewing, in which a machine writes multiple reviews of a paper based on specific aspects, like novelty and impact. See §3.8 for details.

We also conducted additional research focused on generating multi-intent email responses using a re-ranking decoder with a pre-computed cluster of responses (Kang et al., 2017a). We also annotated actionable email intents and classified them over multiple domains (Lin et al., 2018). Please refer to the original papers for further details.

Hierarchical Text Planning

Lastly, unlike vertical and horizontal planning, the goal of hierarchical planning is to control the level of semantic abstraction. Summarizing a text is a good example of using the hierarchical

structure of semantics to abstractualize an original document into a shorter form.

Instead of developing yet another summarization system, we conducted a more comprehensive analysis of summarization corpora and systems. In particular, we defined three aspects of summarization, position, diversity, and importance, and then analyzed how different domains of summarization dataset are biased with respect to these aspects. We observed that news articles strongly reflect the position aspect, while the others do not. Additionally, we investigated how current summarization systems balance the three aspects of summarization, finding that each type of approach has its own bias, while neural systems rarely do. Merging the above systems creates a more balanced and comparable performance than any single one alone. We believe that a good summarization system should reflect the different aspects harmoniously, regardless of corpus bias. Developing a robust, bias-free model will be particularly important for future research. See §3.9 for details.

3.3 Related Work

Horizontal Planning with Causal Relations. Prior work on causality detection (Acharya, 2014; Anand, 2014; Qiu et al., 2012) in time series data (e.g., gene sequence, stock prices, temperature) mainly use Granger (Granger, 1988) ability for predicting future values of a time series using past values of its own and another time series. (Hlaváčková-Schindler et al., 2007) studies more theoretical investigation for measuring causal influence in multivariate time series based on the entropy and mutual information estimation. However, none of them attempts generating explanation on the temporal causality.

Previous works on text causality detection use syntactic patterns such as $X \overset{verb}{\longmapsto} Y$, where the *verb* is causative (Girju, 2003; Riaz and Girju, 2013; Kang et al., 2014; Kozareva, 2012; Do et al., 2011) with additional features (Blanco et al., 2008). (Kozareva, 2012) extracted cause-effect relations, where the pattern for bootstrapping has a form of $X^* \underset{Z^*}{\overset{verb}{\longmapsto}} Y$ from which terms X^* and Z^* was learned. The syntax based approaches, however, are not robust to semantic variation. (Grivaz, 2010) conducts very insightful annotation study of what features are used in human reasoning on causation. Beyond the linguistic tests and causal chains for explaining causality in our work, other features such as counterfactuality, temporal order, and ontological asymmetry remain as our future direction to study.

Horizontal Planning with Discourse Relations. There has been a variety of NLG systems that incorporate additional information between sentences (Appelt, 1982; Reiter and Dale, 2000; Gatt and Krahmer, 2018). Such constraints could be broadly categorized into two forms: explicit and implicit relations.

Explicit relations are represented using external information such as predefined rules or plans, formats, knowledge base, discourse parses, and more: Hovy (1985, 1990) integrated text planning and production in generation, where the plans are considered in knowledge, emotional state, and so forth. In the same way, Dalianis and Hovy (1996) used predefined rules for generating formatted text (e.g., itemized lists). However, they are limited to small scale (i.e. few examples) and hand-written rules. Gardent et al. (2017); Wang et al. (2018) used external knowledge base

to micro-planning for generating a corresponding text, while our work focuses on comparing two different forms of relations from the text itself. Kang et al. (2017b, 2018d) proposed to generate explanations using causality relations or entailment constraints between sentences. Our work proposes more generalized models that can employ different kinds of relations on text generation.

Moore and Paris (1993); Young and Moore (1994) utilized discourse structures such as rhetorical structure theory (RST) (Mann and Thompson, 1988) for parsing a document. A script (Tomkins, 1978) is another structured representation that describes a typical sequence of events in a particular context. Some recent works (Zhang et al., 2016; Ji and Eisenstein, 2014) proposed better discourse parsers using neural networks. Most prior works, however, used a tree of some kind to describe the structure of the paragraph, while we focus on the application of the discourse relations to language generation context.

Implicit relations use implicit information in a document such as hierarchical structure of the document: Lin et al. (2015a); Chung et al. (2016) used hierarchical recurrent neural network for modeling a document. Similarly, the hierarchical model could be extended to with other variants such as attention (Yang et al., 2016), encoder-decoder framework (Serban et al., 2017; Sordoni et al., 2015), auto-encoding (Li et al., 2015), and multiscale (Chung et al., 2016). However, the hierarchical recurrence of sentences, which is dependent to topics, are less likely modeling a flow of a document.

Horizontal Planning with Contentual Goals. Content can be also used to guide the generation. We categorize various generation tasks based on its inclusion relation (**C**-**T**) between the context text to be given (**C**) and the target text to predict (**T**): $\supset$, $\subset$, $\approx$, and $\perp\!\!\!\perp$.

`Context`$\supset$`Target`: Abstracive summarization is an example when the context information is a superset of the target summaries to predict. Here, the generator needs to pay attention on which content to choose as a summary from the context and generate it by copying them (See et al., 2017).

`Context`$\subset$`Target`: Data-to-text generation is to produce text given a structured form of data (e.g., tables, SQLs, semantic parses). Moryossef et al. (2019); Puduppully et al. (2019); Shen et al. (2019) divided the planning and generation in a two-stage process Miculicich et al. (2019) used the pre-trained language model; GPT2 (Radford et al., 2019) for the generation step. However, content is explicitly provided in the data-to-text task so its planning is mostly ordering and structuring the content, while our proposed task; paragraph unmasking, needs to directly predict the content from the context.

`Context`$\approx$`Target`: A paraphrasing is simply transforming surface patterns of text, while preserving its semantics. Fu et al. (2019) used variational autoencoders for surface realization with a latent bag of words (BOW) model for differentiable content planning, but most content of target is given in the context.

`Context`$\perp\!\!\!\perp$`Target`: Storytelling is a very challenging task where the context and target text have no inclusion so independent on each other, but should be coherently connected. The independence of context and target is called *open-ended* in this work. Fan et al. (2019) developed a surface realization model on anonymized entities using semantic role labeling. Hua and Wang (2019) used the pre-extracted topic phrases to guide the generator to produce stylized argumen-

tation text. However, it is still unknown which types of partial content (e.g., named entities, topic phrases) are more effective.

Kang et al. (2019c) developed language models informed by discourse or latent relations on the bridging task; given the first and last sentences, predicting the intermediate sentences. Unlike the storytelling and bridging tasks, our proposed task is more practical, scalable, and goal-oriented task by providing partial plan keywords, augmenting single paragraph into multiple training instances by permutation, and linking the coherency of contextual sentences to predict the reference text, respectively.

Horizontal Planning with Latent Policies on recommendation dialogues. Recommendation systems often rely on matrix factorization (Koren et al., 2009; He et al., 2017b). Content (Mooney and Roy, 2000) and social relationship features (Ma et al., 2011) have also been used to help with the cold-starting problem of new users. The idea of eliciting users' preference for certain content features through dialogue has led to several works. Wärnestål (2005) studies requirements for developing a conversational recommender system, e.g., accumulation of knowledge about user preferences and database content. Reschke et al. (2013) automatically produces template-based questions from user reviews. However, no conversational recommender systems have been built based on these works due to the lack of a large publicly available corpus of human recommendation behaviors.

Very recently, Li et al. (2018) collected the ReDial dataset, comprising 10K conversations of movie recommendations, and used it to train a generative encoder-decoder dialogue system. In this work, crowdsource workers freely talk about movies and are instructed to make a few movie recommendations before accepting one. Compared to ReDial, our dataset is grounded in real movie preferences (movie ratings from MovieLens), instead of relying on workers' hidden movie tastes. This allows us to make our task goal-directed rather than chit-chat; we can optimize prediction and recommendation strategy based on known ground truth, and train the PREDICT and PLAN modules of our system. That, in turn, allows for novel setups such as bot-play.

To the best of our knowledge, Bordes and Weston (2016) is the only other goal-oriented dialogue benchmark grounded in a database that has been released with a large-scale publicly available dataset. Compared to that work, our database is made of real (not made-up) movies, and the choice of target movies is based on empirical distances between movies and movie features instead of being arbitrary. This, combined with the collaborative set-up, makes it possible to train a model for the seeker in the bot-play setting.

Our recommendation dialogue game is collaborative. Other dialogue settings with shared objectives have been explored, for example a collaborative graph prediction task (He et al., 2017a), and semi-cooperative negotiation tasks (Lewis et al., 2017; Yarats and Lewis, 2018; He et al., 2018).

Vertical Planning with Multiple Intents. To improve productivity in the workplace, email communication has been studied in many areas including thread identification (Sharaff and Nagwani, 2016), email summarization (Corston-Oliver et al., 2004), and activity modeling (Qadir et al., 2016). Most of priors works on email response generation rely on pattern matching (Sneiders, 2010), clustering (Bickel and Scheffer, 2004) or message pairing (Malik et al., 2007). The

recent work on SmartReply (Kannan et al., 2016) is the first attempt to automatically *generate* email response suggestions to reduce composition efforts especially for mobile users.

SmartReply first uses a classifier to determine whether to suggest a response candidate or not (90% of long tail emails are filtered out) and chooses a top ranked response from the response clusters that were constructed offline. To construct clean response clusters, (Kannan et al., 2016) generate responses using LSTM (Hochreiter and Schmidhuber, 1997a) and cluster them using label propagation (Wendt et al., 2016). However, their clustering method is semi-supervised so human annotation is required and the final clusters are clean but focus on highly frequent "head" responses. Our decoder based method focuses on extracting response diversity without any human supervision. Moreover, we incorporate external prior knowledge into our sequence to sequence model to generate stylistically personalized responses.

Hierarchical Planning on Textual Summarization . We provide here a brief review of prior work on summarization biases. Lin and Hovy (1997) studied the position hypothesis, especially in the news article writing (Hong and Nenkova, 2014; Narayan et al., 2018a) but not in other domains such as conversations (Kedzie et al., 2018). Narayan et al. (2018a) collected a new corpus to address the bias by compressing multiple contents of source document in the single target summary. In the bias analysis of systems, Lin and Bilmes (2012, 2011) studied the sub-aspect hypothesis of summarization systems. Our study extends the hypothesis to various corpora as well as systems. With a specific focus on `importance` aspect, a recent work (Peyrard, 2019a) divided it into three sub-categories; redundancy, relevance, and informativeness, and provided quantities of each to measure. Compared to this, ours provide broader scale of sub-aspect analysis across various corpora and systems.

We analyze the sub-aspects on different domains of summarization corpora: news articles (Nallapati et al., 2016; Grusky et al., 2018; Narayan et al., 2018a), academic papers or journals (Kang et al., 2018a; Kedzie et al., 2018), movie scripts (Gorinski and Lapata, 2015), books (Mihalcea and Ceylan, 2007), personal posts (Ouyang et al., 2017), and meeting minutes (Carletta et al., 2005).

Beyond the corpora themselves, a variety of summarization systems have been developed: Mihalcea and Tarau (2004); Erkan and Radev (2004) used graph-based keyword ranking algorithms. Lin and Bilmes (2010); Carbonell and Goldstein (1998) found summary sentences which are highly relevant but less redundant. Yogatama et al. (2015) used semantic volumes of bigram features for extractive summarization. Internal structures of documents have been used in summarization: syntactic parse trees (Woodsend and Lapata, 2011; Cohn and Lapata, 2008), topics (Zajic et al., 2004; Lin and Hovy, 2000), semantic word graphs (Mehdad et al., 2014; Gerani et al., 2014; Ganesan et al., 2010; Filippova, 2010; Boudin and Morin, 2013), and abstract meaning representation (Liu et al., 2015). Concept-based Integer-Linear Programming (ILP) solver (McDonald, 2007) is used for optimizing the summarization problem (Gillick and Favre, 2009; Banerjee et al., 2015; Boudin et al., 2015; Berg-Kirkpatrick et al., 2011). Durrett et al. (2016) optimized the problem with grammatical and anarphorcity constraints.

With a large scale of corpora for training, neural network based systems have recently been developed. In abstractive systems, Rush et al. (2015) proposed a local attention-based sequence-to-sequence model. On top of the seq2seq framework, many other variants have been studied

using convolutional networks (Cheng and Lapata, 2016; Allamanis et al., 2016), pointer networks (See et al., 2017), scheduled sampling (Bengio et al., 2015), and reinforcement learning (Paulus et al., 2017). In extractive systems, different types of encoders (Cheng and Lapata, 2016; Nallapati et al., 2017; Kedzie et al., 2018) and optimization techniques (Narayan et al., 2018b) have been developed. Our goal is to explore which types of systems learns which sub-aspect of summarization.

3.4 Causal Planning for Causal Explanation Generation

3.4.1 Introduction

Producing true causal explanations requires deep understanding of the domain. This is beyond the capabilities of modern AI. However, it is possible to collect large amounts of causally related events, and, given powerful enough representational variability, to construct cause-effect chains by selecting individual pairs appropriately and linking them together. Our hypothesis is that chains composed of locally coherent pairs can suggest overall causation.

In this paper, we view *causality* as (commonsense) cause-effect expressions that occur frequently in online text such as news articles or tweets. For example, "*greenhouse gases causes global warming*" is a sentence that provides an 'atomic' link that can be used in a larger chain. By connecting such causal facts in a sequence, the result can be regarded as a *causal explanation* between the two ends of the sequence (see Table 3.2 for examples).
This paper makes the following contributions:

- we define the problem of causal explanation generation,
- we detect causal features of a time series event (CSpikes) using Granger (Granger, 1988) method with features extracted from text such as N-grams, topics, sentiments, and their composition,
- we produce a large graph called CGraph of local cause-effect units derived from text and develop a method to produce causal explanations by selecting and linking appropriate units, using neural representations to enable unit matching and chaining.

The problem of causal explanation generation arises for systems that seek to determine causal factors for events of interest automatically. For given time series events such as companies' stock market prices, our system called CSpikes detects events that are deemed causally related by time series analysis using Granger Causality regression (Granger, 1988). We consider a large amount of text and tweets related to each company, and produces for each company time series of values for hundreds of thousands of word n-grams, topic labels, sentiment values, etc. Figure 3.3 shows an example of causal features that temporally causes Facebook's stock rise in August.

However, it is difficult to understand how the statistically verified factors actually cause the changes, and whether there is a latent causal structure relating the two. This paper addresses the challenge of finding such latent causal structures, in the form of *causal explanations* that connect the given cause-effect pair. Table 3.2 shows example causal explanation that our system found between *party* and *Facebook's stock fall (↓)*.

To construct a general causal graph, we extract all potential causal expressions from a large

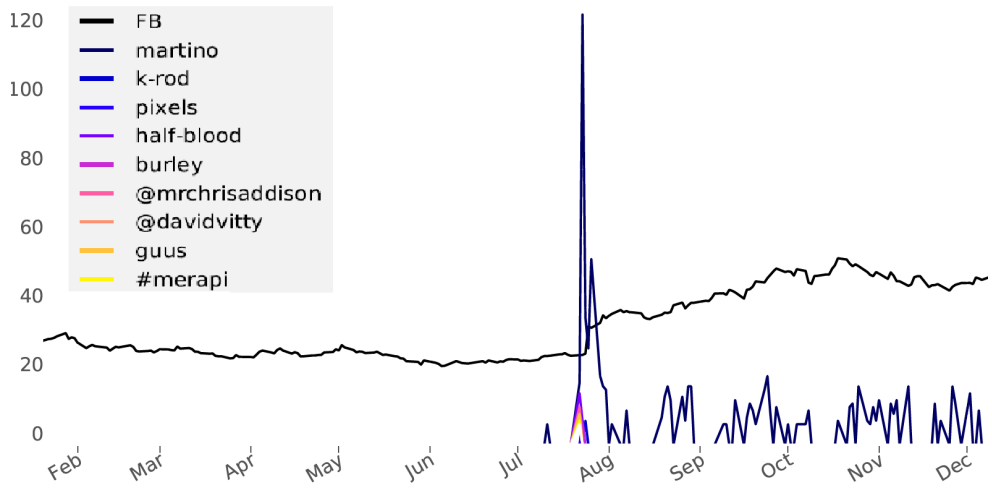


Figure 3.3: Example of causal features for Facebook's stock change in 2013. The causal features (e.g., *martino*, *k-rod*) rise before the Facebook's rapid stock rise in August.

Table 3.2: Examples of generated causal explanation between some temporal causes and target companies' stock prices.

party $\overset{cut}{\longmapsto}$ budget_cuts $\overset{lower}{\longmapsto}$ budget_bill $\overset{decreas}{\longmapsto}$ republicans $\overset{caus}{\longmapsto}$ obama $\overset{leadto}{\longmapsto}$ facebook_polls $\overset{caus}{\longmapsto}$ **facebook's stock** $\downarrow$

corpus of text. We refer to this graph as CGraph. We use FrameNet (Baker et al., 1998) semantics to provide various causative expressions (verbs, relations, and patterns), which we apply to a resource of 183, 253, 995 sentences of text and tweets. These expressions are considerably richer than previous rule-based patterns (Riaz and Girju, 2013; Kozareva, 2012). CGraph contains 5,025,636 causal edges.

Our experiment demonstrates that our causality detection algorithm outperforms other baseline methods for forecasting future time series values. Also, we tested the neural reasoner on the inference generation task using the BLEU score. Additionally, our human evaluation shows the relative effectiveness of neural reasoners in generating appropriate lexicons in explanations.

3.4.2 CSpikes: Temporal Causality Detection from Textual Features

The objective of our model is, given a target time series y, to find the best set of textual features $F = \{f_1, ..., f_k\} \subseteq X$, that maximizes sum of causality over the features on y, where X is the set of all features. Note that each feature is itself a time series:

$$\arg\max_{F} \boldsymbol{C}(y, \Phi(X, y)) \tag{3.1}$$

where $\boldsymbol{C}(y, x)$ is a causality value function between y and x, and Φ is a linear composition function of features f. Φ needs target time series y as well because of our graph based feature selection algorithm described in the next sections.

We first introduce the basic principles of Granger causality in Section 3.4.2. Section 3.4.2 describes how to extract good source features $F = \{f_1, ..., f_k\}$ from text. Section 3.4.2 describes the causality function $\boldsymbol{C}$ and the feature composition function Φ.

Granger Causality

The essential assumption behind Granger causality is that a cause must occur before its effect, and can be used to predict the effect. Granger showed that given a target time series y (effect) and a source time series x (cause), *forecasting* future target value y_t with both past target and past source time series $E(y_t|y_{<t}, x_{<t})$ is significantly powerful than with only past target time series $E(y_t|y_{<t})$ (plain auto-regression), if x and y are indeed a cause-effect pair. First, we learn the parameters α and β to maximize the prediction expectation:

$$E(y_t|y_{<t}, x_{t-l}) = \sum_{j=1}^{m} \alpha_j y_{t-j} + \sum_{i=1}^{n} \beta_i x_{t-i} \tag{3.2}$$

where i and j are size of lags in the past observation. Given a pair of causes x and a target y, if β has magnitude significantly higher than zero (according to a confidence threshold), we can say that x causes y.

Feature Extraction from Text

Extracting meaningful features is a key component to detect causality. For example, to predict future trend of presidential election poll of *Donald Trump*, we need to consider his past poll data as well as people's reaction about his pledges such as *Immigration*, *Syria* etc. To extract such "good" features crawled from on-line media data, we propose three different types of features: F_{words}, F_{topic}, and F_{senti}.

F_{words} is time series of N-gram words that reflect popularity of the word over time in on-line media. For each word, the number of items (e.g., tweets, blogs and news) that contains the N-gram word is counted to get the day-by-day time series. For example, $x^{Michael\_Jordan} = [12, 51, ..]$ is a time series for a bi-gram word *Michael Jordan*. We filter out stationary words by using simple measures to estimate how dynamically the time series of each word changes over time. Some of the simple measures include Shannon entropy, mean, standard deviation, maximum slope, and number of rise and fall peaks.

F_{topic} is time series of latent topics with respect to the target time series. The latent topic is a group of semantically similar words as identified by a standard topic clustering method such as LDA (Blei et al., 2003). To obtain temporal trend of the latent topics, we choose the top ten frequent words in each topic and count their occurrence in the text to get the day-by-day time series. For example, $x^{healthcare}$ means how popular the topic *healthcare* that consists of *insurance*, *obamacare* etc, is through time.

F_{senti} is time series of sentiments (positive or negative) for each topic. The top ten frequent words in each topic are used as the keywords, and tweets, blogs and news that contain at least one of these keywords are chosen to calculate the sentiment score. The day-by-day sentiment series are then obtained by counting positive and negative words using OpinionFinder (Wilson et al., 2005), and normalized by the total number of the items that day.

Temporal Causality Detection

We define a causality function $\boldsymbol{C}$ for calculating causality score between target time series y and source time series x. The causality function $\boldsymbol{C}$ uses Granger causality (Granger, 1988) by fitting the two time series with a Vector AutoRegressive model with exogenous variables (VARX) (Hamilton, 1994): $y_t = \alpha y_{t-l} + \beta x_{t-l} + \epsilon_t$ where ϵ_t is a white Gaussian random vector at time t and l is a lag term. In our problem, the number of source time series x is not single so the prediction happens in the k multi-variate features $X = (f_1, ...f_k)$ so:

$$y_t = \alpha y_{t-l} + \boldsymbol{\beta}(f_{1,t-l} + ... + f_{k,t-l}) + \epsilon_t \tag{3.3}$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ is the coefficient matrix of the target y and source X time series respectively, and ϵ is a residual (prediction error) for each time series. $\boldsymbol{\beta}$ means contributions of each lagged feature $f_{k,t-l}$ to the predicted value y_t. If the variance of $\boldsymbol{\beta_k}$ is reduced by the inclusion of the feature terms $f_{k,t-l} \in X$, then it is said that $f_{k,t-l}$ Granger-causes y.

Our causality function $\boldsymbol{C}$ is then $\boldsymbol{C}(y, f, l) = \Delta(\beta_{y,f,l})$ where Δ is change of variance by the feature f with lag l. The total Granger causality of target y is computed by summing the change of variance over all lags and all features:

$$\boldsymbol{C}(y, X) = \sum_{k,l} \boldsymbol{C}(y, f_k, l) \tag{3.4}$$

We compose best set of features Φ by choosing top k features with highest causality scores for each target y. In practice, due to large amount of computation for pairwise Granger calculation, we make a bipartite graph between features and targets, and address two practical problems: *noisiness* and *hidden edges*. We filter out noisy edges based on TFIDF and fill out missing values using non-negative matrix factorization (NMF) (Hoyer, 2004).

Table 3.3: Example (relation, cause, effect) tuples in different categories (manually labeled): *general*, *company*, *country*, and *people*. FrameNet labels related to causation are listed inside parentheses. The number of distinct relation types are 892.

	Relation	**Cause ↦ Effect**	
General	causes (Causation)	the virus (Cause)	aids (Effect)
	cause (Causation)	greenhouse gases (Cause)	global warming (Effect)
	forced (Causation)	the reality of world war ii (Cause)	the cancellation of the olympics (Effect)
Company	heats (Cause_temperature_change)	microsoft vague on windows (Item)	legislation battle (Agent)
	promotes (Cause_change_of_position_on_a_scale)	chrome (Item)	google (Agent)
	makes (Causation)	twitter (Cause)	love people you 've never met facebook (Effect)
Country	developing (Cause_to_make_progress)	north korea (Agent)	nuclear weapons (Project)
	improve (Cause_to_make_progress)	china (Agent)	its human rights record (Project)
	forced (Causation)	war with china (Cause)	the japanese to admit , in july 1938 (Effect)
People	attracts (Cause_motion)	obama (Agent)	more educated voters (Theme)
	draws (Cause_motion)	on america 's economic brains (Goal)	barack obama (Theme)
	made (Causation)	michael jordan (Cause)	about $ 33 million (Effect)

3.4.3 CGraph Construction

Formally, given source x and target y events that are causally related in time series, if we could find a sequence of cause-effect pairs $(x \mapsto e_1)$, $(e_1 \mapsto e_2)$, ... $(e_t \mapsto y)$, then $e_1 \mapsto e_2$, ... $\mapsto e_t$ might be a good causal explanation between x and y. Section 3.4.3 and 3.4.4 describe how to bridge the causal gap between given events (x, y) by (1) constructing a large general cause-effect graph (CGraph) from text, (2) linking the given events to their equivalent entities in the causal graph by finding the internal paths $(x \mapsto e_1, ...e_t \mapsto y)$ as causal explanations, using neural algorithms.

CGraph is a knowledge base graph where edges are directed and causally related between entities. To address less representational variability of rule based methods (Girju, 2003; Blanco et al., 2008; Sharp et al., 2016) in the causal graph construction, we used FrameNet (Baker et al., 1998) semantics. Using a semantic parser such as SEMAFOR (Chen et al., 2010) that produces a FrameNet style analysis of semantic predicate-argument structures, we could obtain lexical tuples of causation in the sentence. Since our goal is to collect only causal relations, we extract total 36 causation related frames[1] from the parsed sentences.

Table 3.4: Number of sentences parsed, number of entities and tuples, and number of edges (*KB-KB*, *KBcross*) expanded by Freebase in CGraph.

# Sentences	# Entities	# Tuples	# *KB-KB*	# *KBcross*
183,253,995	5,623,924	5,025,636	470,250	151,752

To generate meaningful explanations, high coverage of the knowledge is necessary. We collect six years of tweets and NYT news articles from 1989 to 2007 (See Experiment section for details). In total, our corpus has 1.5 billion tweets and 11 million sentences from news articles. The Table 3.4 has the number of sentences processed and number of entities, relations, and tuples in the final CGraph.

Since the tuples extracted from text are very noisy [2], we constructed a large causal graph by linking the tuples with string match and filter out the noisy nodes and edges based on some graph statistics. We filter out nodes with very high degree that are mostly stop-words or auto-generated sentences. Too long or short sentences are also filtered out. Table 3.3 shows the (case, relation, effect) tuples with manually annotated categories such as *General*, *Company*, *Country*, and *People*.

3.4.4 Causal Reasoning

To generate a causal explanation using CGraph, we need traversing the graph for finding the path between given source and target events. This section describes how to efficiently traverse the graph by expanding entities with external knowledge base and how to find (or generate) appropriate causal paths to suggest an explanation using symbolic and neural reasoning algorithms.

[1]Causation, Cause_change, Causation_scenario, Cause_ benefit_or_detriment, Cause_bodily_experience, etc.
[2]SEMAFOR has around 62% of accuracy on held-out set.

Entity Expansion with Knowledge Base

A simple choice for traversing a graph are the traditional graph searching algorithms such as Breadth-First Search (BFS). However, the graph searching procedure is likely to be incomplete (*low recall*), because simple string match is insufficient to match an effect to all its related entities, as it misses out in the case where an entity is semantically related but has a lexically different name.

To address the *low recall* problem and generate better explanations, we propose the use of knowledge base to augment our text-based causal graph with real-world semantic knowledge. We use Freebase (Google, 2016) as the external knowledge base for this purpose. Among 1.9 billion edges in original Freebase dump, we collect its first and second hop neighbours for each target events.

While our CGraph is lexical in nature, Freebase entities appear as identifiers (MIDs). For entity linking between two knowledge graphs, we need to annotate Freebase entities with their lexical names by looking at the wiki URLs. We refer to the edges with freebase expansion as *KB-KB* edges, and link the *KB-KB* with our CGraph using lexical matching, referring as *KBcross* edges (See Table 3.4 for the number of the edges).

Symbolic Reasoning

Simple traversal algorithms such as BFS are infeasible for traversing the CGraph due to the large number of nodes and edges. To reduce the search space k in $e_t \mapsto \{e^1_{t+1}, ...e^k_{t+1}\}$, we restricted our search by depth of paths, length of words in entity's name, and edge weight.

Algorithm 2 Backward Causal Inference. y is target event, d is depth of BFS, l is lag size, BFS_{back} is Breadth-First search for one depth in backward direction, and $\sum_l \boldsymbol{C}$ is sum of Granger causality over the lags.

$\mathbb{S} \leftarrow y, d = 0$
while ($\mathbb{S} = \emptyset$) or ($d > D_{max}$) **do**
  $\{e^1_{-d}, ...e^k_{-d}\} \leftarrow BFS_{back}(\mathbb{S})$
  $d = d + 1, \mathbb{S} \leftarrow \emptyset$
  for j `in` $\{1, ..., k\}$ **do**
    if $\sum_l \boldsymbol{C}(y, e^j_{-d}, l) < \epsilon$ **then** $\mathbb{S} \leftarrow e^j_{-d}$
    end if
  end for
end while

For more efficient inference, we propose a backward algorithm that searches potential causes (instead of effects) $\{e^1_t, ...e^k_t\} \leftarrow\!\shortmid\ e_{t+1}$ starting from the target node $y = e_{t+1}$ using Breadth-first search (BFS). It keeps searching backward until the node e^j_t has less Granger confident causality with the target node y (See Algorithm 3.4 for causality calculation). This is only possible because our system has temporal causality measure between two time series events. See Algorithm 2 for detail.

Neural Reasoning

While symbolic inference is fast and straightforward, the sparsity of edges may make our inference semantically poor. To address the *lexical sparseness*, we propose a lexically relaxed reasoning using a neural network.

Inspired by recent success on alignment task such as machine translation (Bahdanau et al., 2014), our model learns the causal alignment between cause phrase and effect phrase for each type of relation between them. Rather than traversing the CGraph, our neural reasoner uses CGraph as a training resource. The encoder, a recurrent neural network such as LSTM (Hochreiter and Schmidhuber, 1997a), takes the causal phrase while the decoder, another LSTM, takes the effectual phrase with their relation specific attention.

In original attention model (Bahdanau et al., 2014), the contextual vector c is computed by $c_i = a_{ij} * h_j$ where h_j is hidden state of causal sequence at time j and a_{ij} is soft attention weight, trained by feed forward network $a_{ij} = FF(h_j, s_{i-1})$ between input hidden state h_j and output hidden state s_{i-1}. The global attention matrix a, however, is easy to mix up all local alignment patterns of each relation.

For example, a tuple, *(north korea (Agent)* $\xmapsto[(Cause\_to\_make\_progress)]{developing}$ *nuclear weapons (Project))*, is different with another tuple, *(chrome (Item)* $\xmapsto[(Cause\_change\_of\_position)]{promotes}$ *google (Agent))* in terms of local type of causality. To deal with the *local attention*, we decomposed the attention weight a_{ij} by relation specific transformation in feed forward network:

$$a_{ij} = FF(h_j, s_{i-1}, r)$$

where FF has relation specific hidden layer and $r \in R$ is a type of relation in the distinct set of relations R in training corpus (See Figure 3.4).

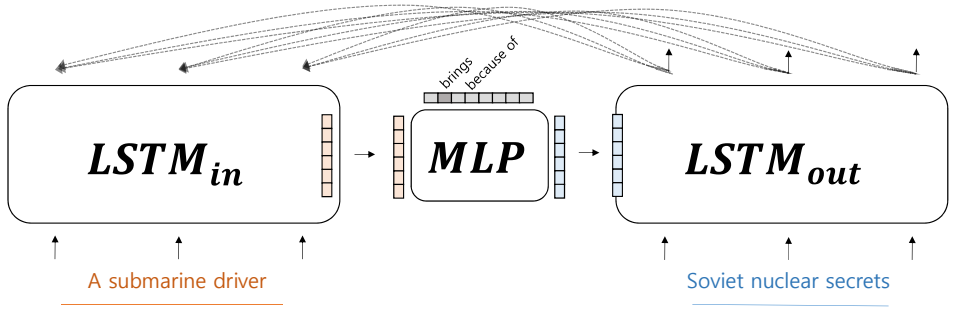


Figure 3.4: Our neural reasoner. The encoder takes causal phrases and decoder takes effect phrases by learning the causal alignment between them. The MLP layer in the middle takes different types of FrameNet relation and locally attend the cause to the effect w.r.t the relation (e.g., "because of", "led to", etc).

Since training only with our causal graph may not be rich enough for dealing various lexical variation in text, we use pre-trained word embedding such as word2vec (Mikolov and Dean,

2013) trained on GoogleNews corpus[3] for initialization. For example, given a cause phrase *weapon equipped*, our model could generate multiple effect phrases with their likelihood: *(*$\overset{result}{\underset{0.54}{\longmapsto}}$*war)*, *(*$\overset{force}{\underset{0.12}{\longmapsto}}$*army reorganized)*, etc, even though there are no tuples exactly matched in CGraph.

We trained our neural reasoner in either forward or backward direction. In prediction, decoder inferences by predicting effect (or cause) phrase in forward (or backward) direction. As described in the Algorithm 2, the backward inference continue predicting the previous causal phrases until it has high enough Granger confidence with the target event.

3.4.5 Results

Data. We collect on-line social media from tweets, news articles, and blogs. Our Twitter data has one million tweets per day from 2008 to 2013 that are crawled using Twitter's Garden Hose API. News and Blog dataset have been crawled from 2010 to 2013 using Google's news API. For target time series, we collect companies' stock prices in NASDAQ and NYSE from 2001 until present for 6,200 companies. For presidential election polls, we collect polling data of the 2012 presidential election from 6 different websites, including USA Today , Huffington Post, Reuters, etc.

Table 3.5: Examples of F_{words} with their temporal dynamics: Shannon entropy, mean, standard deviation, slope of peak, and number of peaks.

	entropy	**mean**	**STD**	**max_slope**	**#-peaks**
#lukewilliamss	0.72	22.01	18.12	6.12	31
happy_thanksgiving	0.40	61.24	945.95	3423.75	414
michael_jackson	0.46	141.93	701.97	389.19	585

Features. For N-gram word features F_{word},we choose the spiking words based on their temporal dynamics (See Table 3.5). For example, if a word is too frequent or the time series is too burst, the word should be filtered out because the trend is too general to be an event. We choose five types of temporal dynamics: Shannon entropy, mean, standard deviation, maximum slope of peak, and number of peaks; and delete words that have too low or high entropy, too low mean and deviation, or the number of peaks and its slope is less than a certain threshold. Also, we filter out words whose frequency is less than five. From the $1,677,583$ original words, we retain $21,120$ words as final candidates for F_{words} including uni-gram and bi-gram words.

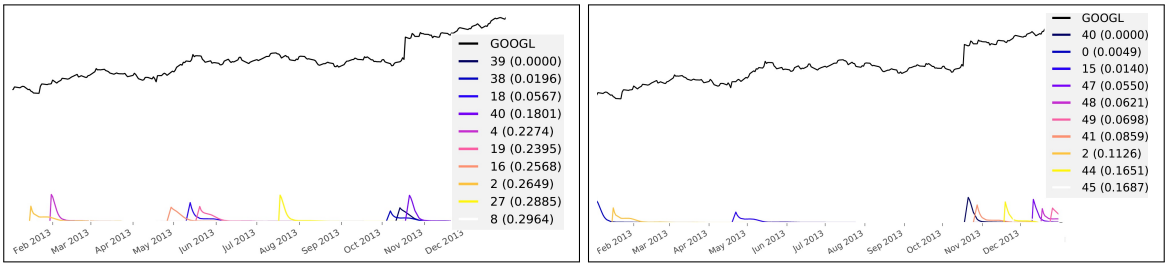
For sentiment F_{senti} and topic F_{topic} features, we choose 50 topics generated for both politicians and companies separately using LDA, and then use top 10 words for each topic to calculate sentiment score for this topic. Then we can analyze the causality between sentiment series of a specific topic and collected time series.

Tasks. To show validity of causality detector, first we conduct random analysis between target time series and randomly generated time series. Then, we tested forecasting stock prices and

[3]https://code.google.com/archive/p/word2vec/

election poll values with or without the detected textual features to check effectiveness of our causal features. We evaluate our reasoning algorithm for generation ability compared to held-out cause-effect tuples using BLEU metric. Then, for some companies' time series, we describe some qualitative result of some interesting causal text features found with Granger causation and explanations generated by our reasoners between the target and the causal features. We also conducted human evaluation on the explanations.

Random Causality Analysis



(a) $y \xleftarrow{lag=3} rf_1, ..., rf_k$ **(b)** $y \xrightarrow{lag=3} rf_1, ..., rf_k$

Figure 3.5: Random causality analysis on **Googles**'s stock price change (y) and randomly generated features (rf) during 2013-01-01 to 2013-12-31. (a) shows how the random features rf cause the target y, while (b) shows how the target y causes the random features rf with lag size of 3 days. The color changes according to causality confidence to the target (blue is the strongest, and yellow is the weakest). The target time series has y scale of prices, while random features have y scale of causality degree $\boldsymbol{C}(y, rf) \subset [0, 1]$.

To check whether our causality scoring function $\boldsymbol{C}$ detects the temporal causality well, we conduct a random analysis between target time series and randomly generated time series (See Figure 3.5). For Google's stock time series, we regularly move window size of 30 over the time and generate five days of time series with a random peak strength using a SpikeM model (Matsubara et al., 2012)[4]. The color of random time series rf changes from blue to yellow according to causality degree with the target $\boldsymbol{C}(y, rf)$. For example, blue is the strongest causality with target time series, while yellow is the weakest.

We observe that the strong causal (blue) features are detected just before (or after) the rapid rise of Google' stock price on middle October in (a) (or in (b)). With the lag size of three days, we observe that the strength of the random time series gradually decreases as it grows apart from the peak of target event. The random analysis shows that our causality function $\boldsymbol{C}$ appropriately finds cause or effect relation between two time series in regard of their strength and distance.

Forecasting with Textual Features

We use time series forecasting task as an evaluation metric of whether our textual features are appropriately causing the target time series or not. Our feature composition function Φ is used

[4]SpikeM has specific parameters for modeling a time series such as peak strength, length, etc.

Table 3.6: Forecasting errors (RMSE) on **Stock** and **Poll** data with time series only (*SpikeM* and *LSTM*) and with time series plus text feature (*random*, *words*, *topics*, *sentiment*, and *composition*).

		Time Series		Time Series + Text				
	Step	SpikeM	LSTM	$\boldsymbol{C}_{rand}$	$\boldsymbol{C}_{words}$	$\boldsymbol{C}_{topics}$	$\boldsymbol{C}_{senti}$	$\boldsymbol{C}_{comp}$
Stock	1	102.13	6.80	3.63	2.97	3.01	3.34	<u>1.96</u>
	3	99.8	7.51	4.47	4.22	4.65	4.87	<u>3.78</u>
	5	97.99	7.79	5.32	<u>5.25</u>	5.44	5.95	5.28
Poll	1	10.13	1.46	1.52	1.27	1.59	2.09	<u>1.11</u>
	3	10.63	1.89	1.84	1.56	1.88	1.94	<u>1.49</u>
	5	11.13	2.04	2.15	1.84	1.88	1.96	<u>1.82</u>

to extract good causal features for forecasting. We test forecasting on stock price of companies (**Stock**) and predicting poll value for presidential election (**Poll**). For stock data, We collect daily closing stock prices during 2013 for ten IT companies[5]. For poll data, we choose ten candidate politicians [6] in the period of presidential election in 2012.

Table 3.7: Beam search results in neural reasoning. These examples could be filtered out by graph heuristics before generating final explanation though.

Cause↦Effect in CGraph	Beam Predictions
the dollar's $\overset{caus}{\longmapsto}$ against the yen	[1] $\overset{caus}{\longmapsto}$ against the yen [2] $\overset{caus}{\longmapsto}$ against the dollar [3] $\overset{caus}{\longmapsto}$ against other currencies
without any exercise $\overset{caus}{\longmapsto}$ news article	[1] $\overset{leadto}{\longmapsto}$ a difference [2] $\overset{caus}{\longmapsto}$ the risk [3] $\overset{make}{\longmapsto}$ their weight

For each of stock and poll data, the future trend of target is predicted only with target's past time series or with target's past time series and past time series of textual features found by our system. Forecasting only with target's past time series uses *SpikeM* (Matsubara et al., 2012) that models a time series with small number of parameters and simple *LSTM* (Hochreiter and Schmidhuber, 1997a; Jiménez, 2015) based time series model. Forecasting with target and textual features' time series use Vector AutoRegressive model with exogenous variables (VARX) (Hamilton, 1994) from different composition function such as $\boldsymbol{C}_{random}$, $\boldsymbol{C}_{words}$, $\boldsymbol{C}_{topics}$, $\boldsymbol{C}_{senti}$, and $\boldsymbol{C}_{composition}$. Each composition function except $\boldsymbol{C}_{random}$ uses top ten textual features that causes each target time series. We also tested LSTM with past time series and textual features but VARX outperforms LSTM.

[5]Company symbols used: TSLA, MSFT, GOOGL, YHOO, FB, IBM, ORCL, AMZN, AAPL and HPO

[6]Name of politicians used: Santorum, Romney, Pual, Perry, Obama, Huntsman, Gingrich, Cain, Bachmann

Table 3.6 shows root mean square error (RMSE) for forecasting with different step size (time steps to predict), different set of features, and different regression algorithms on stock and poll data. The forecasting error is summation of errors over moving a window (30 days) by 10 days over the period. Our $\boldsymbol{C}_{composition}$ method outperforms other time series only models and time series plus text models in both stock and poll data.

Generating Causality with Neural Reasoner

The reasoner needs to predict the next effect phrase (or previous cause phrase) so the model should be evaluated in terms of generation task. We used the BLEU (Papineni et al., 2002) metric to evaluate the predicted phrases on held out phrases in our CGraph . Since our CGraph has many edges, there may be many good paths (explanations), possibly making our prediction diverse. To evaluate such diversity in prediction, we used ranking-based BLEU method on the k set of predicted phrases by beam search. For example, $B@k$ means BLEU scores for generating k number of sentences and $B@kA$ means the average of them.

Table 3.7 shows some examples of our beam search results when $k = 3$. Given a cause phrase, the neural reasoner sometime predicts semantically similar phrases (e.g., *against the yen*, *against the dollar*), while it sometimes predicts very diverse phrases (e.g., *a different*, *the risk*).

Table 3.8 shows BLEU ranking results with different reasoning algorithms: **S2S** is a sequence to sequence learning trained on CGraph by default, **S2S+WE** adds word embedding initialization, and **S2S+REL+WE** adds relation specific attention. Initializing with pre-trained word embeddings (**+WE**) helps us improve on prediction. Our relation specific attention model outperforms the others, indicating that different type of relations have different alignment patterns.

Table 3.8: BLEU ranking. Additional word representation **+WE** and relation specific alignment **+REL** help the model learn the cause and effect generation task especially for diverse patterns.

	B@1	B@3A	B@5A
S2S	10.15	8.80	8.69
S2S + WE	11.86	10.78	10.04
S2S + WE + REL	12.42	12.28	11.53

Generating Explanation by Connecting

Evaluating whether a sequence of phrases is reasonable as an explanation is very challenging task. Unfortunately, due to lack of quantitative evaluation measures for the task, we conduct a human annotation experiment.

Table 3.9 shows example causal chains for the rise (↑) and fall (↓) of companies' stock price, continuously produced by two reasoners: *SYBM* is symbolic reasoner and *NEUR* is neural reasoner.

We also conduct a human assessment on the explanation chains produced by the two reasoners, asking people to choose more convincing explanation chains for each feature-target pair. Table 3.10 shows their relative preferences.

Table 3.9: Example causal chains for explaining the rise (↑) and fall (↓) of companies' stock price. The temporally causal *feature* and *target* are linked through a sequence of predicted cause-effect tuples by different reasoning algorithms: a symbolic graph traverse algorithm *SYMB* and a neural causality reasoning model *NEUR*.

SYMB	**medals** $\overset{match}{\longmapsto}$ gold_and_silver_medals $\overset{swept}{\longmapsto}$ korea $\overset{improving}{\longmapsto}$ relations $\overset{widened}{\longmapsto}$ gap $\overset{widens}{\longmapsto}$ **facebook** ↑
	excess $\overset{match}{\longmapsto}$ excess_materialism $\overset{cause}{\longmapsto}$ people_make_films $\overset{make}{\longmapsto}$ money $\overset{changed}{\longmapsto}$ twitter $\overset{turned}{\longmapsto}$ **facebook** ↓
	clinton $\overset{match}{\longmapsto}$ president_clinton $\overset{raised}{\longmapsto}$ antitrust_case $\overset{match}{\longmapsto}$ government's_antitrust_case_against_microsoft $\overset{match}{\longmapsto}$ microsoft $\overset{beats}{\longmapsto}$ **apple** ↓
NEUR	**google** $\overset{forc}{\longmapsto}$ microsoft_to_buy_computer_company_dell_announces_recall_of_batteries $\overset{cause}{\longmapsto}$ **microsoft** ↑
	the_deal $\overset{make}{\longmapsto}$ money $\overset{rais}{\longmapsto}$ at_warner_music_and_google_with_protest_videos_things $\overset{caus}{\longmapsto}$ **google** ↓
	party $\overset{cut}{\longmapsto}$ budget_cuts $\overset{lower}{\longmapsto}$ budget_bill $\overset{decreas}{\longmapsto}$ republicans $\overset{caus}{\longmapsto}$ obama $\overset{leadto}{\longmapsto}$ facebook_polls $\overset{caus}{\longmapsto}$ **facebook** ↓
	company $\overset{forc}{\longmapsto}$ to_stock_price $\overset{leadto}{\longmapsto}$ investors $\overset{increas}{\longmapsto}$ oracle_s_stock $\overset{increas}{\longmapsto}$ **oracle** ↑

Table 3.10: Human evaluation on explanation chains generated by symbolic and neural reasoners.

Reasoners	SYMB	NEUR
Accuracy (%)	42.5	57.5

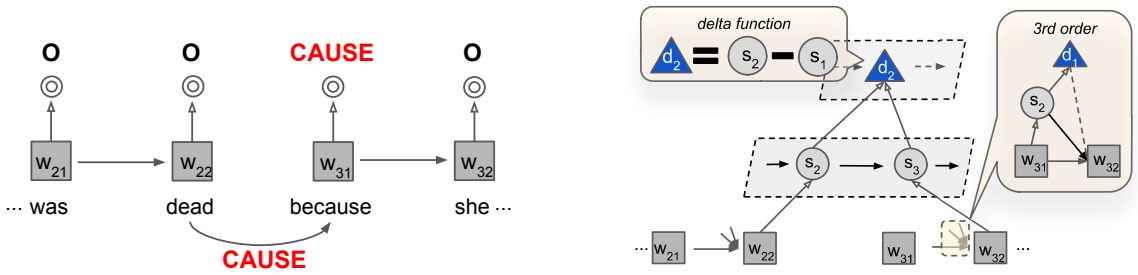
3.4.6 Conclusion

This paper defines the novel task of detecting and explaining causes from text for a time series. First, we detect causal features from online text. Then, we construct a large cause-effect graph using FrameNet semantics. By training our relation specific neural network on paths from this graph, our model generates causality with richer lexical variation. We could produce a chain of cause and effect pairs as an explanation which shows some appropriateness. Incorporating aspects such as time, location and other event properties remains a point for future work. In our following work, we collect a sequence of causal chains verified by domain experts for more solid evaluation of generating explanations.

3.5 Discourse Planning for Paragraph Bridging

3.5.1 Introduction

When composing multiple sentences into a paragraph, as in novels or academic papers, we often make design decisions in advance (Byrne, 1979) such as topic introduction and content ordering to ensure better coherence of the text. For instance, McKeown (1985); Swan (2002) proposed effective patterns for scientific writing: a hypothesis at first, followed by supporting sentences to validate the hypothesis, and lastly a concluding sentence. We call such a logical connection between sentences in a written paragraph as a *flow*. A coherent flow between sentences requires an understanding of various factors including tense, coreference, plans (Appelt, 1982; Hovy, 1991), scripts (Tomkins, 1978) and several others. We focus on the paragraph-level plan between sentences.



(a) Discourse-driven **(b)** Delta-driven

Figure 3.6: FLOWNET with linguistic (i.e., discourse) versus latent (i.e., delta) relation. (a) For each word, a form of discourse relation and next word are jointly predicted using CRF (◎) and language model, respectively. (b) Decoding w_i is conditioned on previous word (w_{i-1}), previous sentence (s_{i-1}), and delta between two previous sentences (d_{i-2}). Best viewed in color.

In text planning, underlying relations in text are broadly categorized into two forms: an explicit human-defined relation (e.g., a discourse tree) (Reiter and Dale, 2000) or an implicitly learned latent relation (Yang et al., 2016). While the former is defined and manuallly annotated based on linguistic theories, the latter is simply determinable from how people in fact put sentences together. In this work, we provide an empirical comparison between a linguistically-informed and a latent form of relations in context of a paragraph generation.

We compare the effectiveness of the two forms of relations using language modeling for paragraph generation. Due to the different characteristics of the two forms, we employ comparable but different components in addition to the base language model. For linguistic relations (e.g., discourse), we cast the problem into multi-task learning of supervised language modeling and discourse relation prediction. On the other hand, for latent relations, we learn an unsupervised hierarchical language model that is hierarchically conditioned by RNNs over linear operations between sentences.

We evaluate our models on partial paragraph generation task; producing the rest of text in a paragraph given some context of text. We observe that linguistically annotated discourse relations help produce more coherent text than the latent relations, followed by other baselines.

3.5.2 FLOWNET: Language Modeling with Inter-sentential Relations

We propose language models that incorporate each relation to capture a high-level flow of text.

Discourse-driven FLOWNET

As a linguistic relation, we employ RST (Mann and Thompson, 1988) trees to represent discourse connections in the text. For simplicity, we limit usage of the discourse trees by only considering relations between adjacent phrases[7]: relations are inserted between adjacent phrases and represented as a flattened sequence of phrases and relations. If two consecutive RST relations are given, the deeper level of relation is chosen. If the central elementary discourse unit (EDU) or

[7]The full discourse tree can be incorporated using other types of language model such as Tai et al. (2015).

phrase is after its dependent, the relation is excluded. We consider each sequence of the flattened discourse relations as a writing flow. For example, people often write a text by elaborating basic information (`Elaboration`) and then describing a following statement attributed to the information (`Attribution`).

We view discourse relations as additional labels to predict at the same time we predict next words in language modeling. Specifically, we propose to jointly train a model that predicts a sequence of words and a sequence of RST labels by taking advantage of shared representations, following previous sequence labeling problems such as named entity recognition (Collobert et al., 2011) and part-of-speech tagging (Huang et al., 2015). Note that the RST relations are only used during training to obtain better representation for the two tasks, but not at test time.

Figure 3.6(a) shows our FlowNet using discourse relations. Let a paragraph be a sequence of sentences D=$\{s_1, s_2, \ldots, s_M\}$. This model treats adjacent sentences as pairs for learning the standard seq2seq model. The first objective is to maximize the likelihood of the current sentence given the previous sentence. Hence, we maximize the following:

$$\mathbb{L}_{s2s} = \sum_j \log P(w_{ij} | w_{i,<j}, s_{i-1}) \tag{3.5}$$

where s_i=$\{w_{i1}, w_{i2}, \ldots, w_{iT_i}\}$, and T_i is the number of tokens of s_i.

To better guide the model with discourse context, we use the shared representations to predict RST relations at the same time. For each paragraph, we run the pre-trained RST parser (Ji and Eisenstein, 2014) and flatten the parse tree to obtain RST relations for each sentence Y_i=$(y_1, \ldots, y_{K_i})$, where K_i is the number of discourse relations in s_i. We then make a label sequence over tokens in the sentence with by placing y at the first word of EDUs and filling up the rest with a *null* relation o: $Y'_i = (o, \ldots, o, \mathbf{y_1}, o, \ldots, \mathbf{y_{K_i}}, o, \ldots, o)$. We incorporate a sequence labeling objective by employing conditional random field (Lafferty et al., 2001) to find the label sequence that maximizes the score function for each sentence s_i: $\mathbf{S}(s_i, Y'_i) = \sum_{j=1}^{T_i-1} W^T_{y'_j, y'_{j+1}} h_j + b_{y'_j, y'_{j+1}}$ where h_j, W and b are the hidden representation of w_{ij}, weight matrix, and the bias vector corresponding to the pair of labels (y'_i, y'_{i+1}), respectively. For training, we maximize the conditional likelihood:

$$\mathbb{L}_{CRF} = \mathbf{S}(s_i, y'_i) - \sum_{\mathbf{y} \in \mathbb{Y}_x} \log \mathbf{S}(s_i, \mathbf{y}) \tag{3.6}$$

where $\mathbb{Y}_x$ represents all possible discourse label sequences. Decoding is done by greedily predicting the output sequence with maximum score. Both training and decoding can be computed using dynamic programming. The final objective is represented as the sum of two objective functions:

$$\mathbb{L}_{\texttt{disc}} = \mathbb{L}_{s2s} + \alpha * \mathbb{L}_{CRF} \tag{3.7}$$

where α is a scaling parameter to control the impact of CRF objective. The value is chosen empirically by searching based on validation set.

Delta-driven FlowNet

In this model, we aim to utilize latent representations to characterize the flow between sentences. Specifically we define *delta*, subtractions of hidden represenations of adjacent sentences as such

latent information. Figure 3.6(b) shows how we hierarchically model different levels of information: words, sentences, and *deltas*.

Each word is encoded using a RNN encoder g_{word}. We take the last hidden representation of word as sentence embeddings $s_1, ..., s_M$. Similar to hierarchical RNN (Lin et al., 2015a), each sentence representation is encoded using another RNN encoder g_{sent}. While discourse flow provides an explicit relation symbols, delta flow calculates a latent relation by subtracting previous representation s_{i-1} from current representation s_i[8]:

$$d(s_{i-1}, s_i) = d_{i-1} = s_i - s_{i-1} \tag{3.8}$$

Given a sequence of M-1 delta relations $d_1, ..., d_{M-1}$ for a paragraph of M sentences, we again encode them using another RNN encoder g_{delta}. The model takes the word, sentence and delta information altogether to predict the next (t-th) word in the m-th sentence:

$$h_t = f(h_{t-1}, x_t, s_{m-1}, d_{m-2}) \tag{3.9}$$

where x_t is a word representation, s_{m-1} is a sentence representation and d_{m-2} is a delta information. Note that sentence representation is from the previous sentence, and delta information is calculated by two previous sentences. If there is no previous information given, the parameters are randomly initialized.

3.5.3 Results

Due to the absence of goal-oriented language generation task, we collect paragraph data and define a new task of generating partial text of a paragraph given some context.

Data

Table 3.11: Number of paragraphs in our dataset.

	Train	Valid	Test
`Papers`	16,173	899	899
`SciFi`	157,031	8,724	8,724
`Fantasy`	317,654	17,649	17,649

We collect paragraphs from three different domains: `Papers` are paragraphs extracted from academic manuscripts in computer science domain from the PeerRead (Kang et al., 2018b), and `Fantasy` and `SciFi` are paragraphs of two frequent categories extracted from the BookCorpus (Zhu et al., 2015), where paragraphs are extracted using the line breaker in the dataset.

We only use paragraphs whose lengths are from 4 to 7, in order to measure the performance change according to paragraph length. The dataset is randomly split by 0.9/0.05/0.05 for train,

[8]Our experiment includes a comparison among other types of linear operations between sentences such as addition or a learnable function.

valid, and test set, respectively. Table 3.11 shows the numbers of paragraphs for each domain. All paragraphs are parsed into RST trees using the state-of-the-art discourse parser by Ji and Eisenstein (2014).

Bridging: Partial Paragraph Generation

We evaluate our models on partial text generation task; given a partial information (e.g., some sentences), producing the rest of text.

Figure 3.7: Bridging task: given [1] and [4] sentences, guessing [2,3] sentences (red, underlined).

---

[1] Inside the club we moved straight for the bar. [2] Devlin ordered a beer for himself and a glass of my favorite wine for me. [3] I love that I didn't have to tell him what I wanted. [4] He knew me well and always thought about what I wanted or needed, in and out of bed.

Figure 3.7 shows our bridging task. It requires a generation of masked sentences in the middle of a paragraph given the first and the last sentences. If only the first sentence is given, the generation can be too divergent. The existence of the last sentence makes the generation more coherent and converged to some point.

We evaluate it with one hard and one soft automatic metrics: METEOR (M) (Banerjee and Lavie, 2005) and VectorExtrema (VE) (Liu et al., 2016) by calculating cosine similarity of averaged word embeddings (Pennington et al., 2014), and human performance.

Models and Setup

We compare various baseline seq2seq models which encode the context; a concatenated first and last sentences, and decode the intermediate words: **S2S** is attentional seq2seq model (Bahdanau et al., 2014), and **HS2S**: is a hierarchical version of the S2S by combining two baselines: HRNN (Lin et al., 2015a) hierarchically models sequence of words and sentences, and HRED (Serban et al., 2017; Sordoni et al., 2015) encodes the given context and decodes the words. **FlowNet (delta/disc.)** is our proposed language model with delta and discourse relations, respectively.

We find the best hyper-parameters on validation set using grid search. Here are the final parameters used: 32 for batch size, 25 for maximum sentence length, 300 for word embedding size initialized by GloVe (Pennington et al., 2014), 1 LSTM layer (Hochreiter and Schmidhuber, 1997a) with 512 size, clipping by 0.25, 0.2 learning rate and 0.5 decay rate with Adagrad (Duchi et al., 2011) optimizer, and 50, 000 for the vocabulary size. The total number of distinct discourse relations is 44.

Results

In Table 3.12, both discourse and delta driven FlowNet outperform the baseline models across most of the metrics except for VecterExtrema on `SciFi`. Especially, as the number of training size increases (`Papers<<SciFi<Fantasy`), the improvements gained from the FlowNet become

Table 3.12: Performance on bridging task. **M**ETEOR and **V**ector**E**xtrema are used. The higher the better.

	Papers		SciFi		Fantasy	
	M	**VE**	**M**	**VE**	**M**	**VE**
S2S	3.7	56.3	3.5	71.0	3.3	66.3
HS2S	3.7	54.7	3.4	**73.0**	3.0	69.7
FLOWNET (delta)	3.1	**58.5**	3.6	69.7	3.6	**73.9**
FLOWNET (disc.)	**4.0**	57.2	**4.2**	70.3	**3.9**	71.8

bigger. This is probably because the model learns more information of the (discourse or latent) relations from the larger data.

	M	VE
SUBTRACT	3.35	**67.20**
ADD	**3.45**	65.35
MLP	3.32	62.97

Figure 3.8: Comparison of different delta functions.

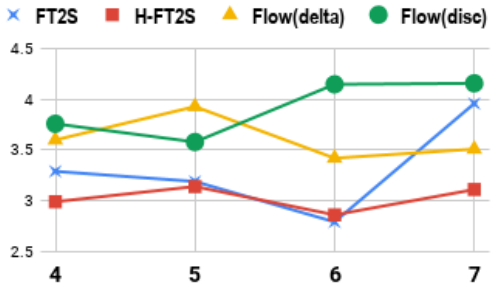


Figure 3.9: Comparison of paragraph lengths. Best viewed in color.

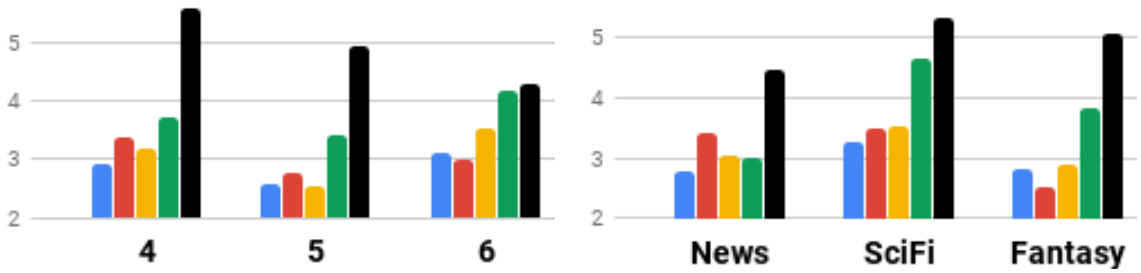
Table 3.8 shows performance comparison among different delta operations: SUBTRACT, ADD, and MLP which is a multi-layer perceptron network. All scores are macro-averaged across datasets. While ADD shows good performance on METEOR, SUBTRACT does on the soft metric (i.e., VecExt), indicating that subtraction can help the model capture the better semantics than the other functions. Figure 3.9 shows how performance changes on Fantasy as the paragraph lengths increase. Both of FLOWNET achieve more improvements when generating longer paragraphs. Especially, discourse relations achieve the best performance at length 6 and 7.

We conduct a comparison with human performance (See Figure 3.10). We randomly choose 100 samples per dataset and per paragraph length and ask an annotator to perform the bridging task on the final 1,000 samples. Human outperforms the models by large margins. FLOWNET with discourse relations outperforms the FLOWNET with latent relations and other baselines by a large margin. As the paragraph length increases or more data is trained, discourse relations become more useful.

Table 3.13 shows an example paragraph with text produced by the models as well as reference and human annotation. Given only the partial context (i.e., first and last sentences), bridging task is very challenging even for human. The reference sentences and human annotations are semantically very different indeed. Among the latent models, FLOWNET (delta) produces more

Table 3.13: An example paragraph and predicted texts in `Fantasy` dataset. Given **First** and **Last** sentences, the models generate middle sentences (e.g., [M1] → [M2]..). **Ref** and **Human** are reference middle sentences and sentences written by human annotator, respectively. Please find more examples in the appendix.

First: Satyrs never wear armor, including helmets, Newel began, using his hands expressively. **Last**: Anyhow, as we actors were laying siege, a big chunk of the battlement dislodged from atop the tower.
Ref: [M1] "But years ago I was in a play, and the helm was part of my costume. [M2] During the big battle scene, a few of us were assailing a castle. [M3] We had quite a set. [M4] The main tower must have been fifteen feet tall, fashioned from real stone. **Human**: [M1] Actually he needed to wear any protectors to prevent him from a big accident. [M2] We planned to make a prank cam to make him wear those always. [M3] "I have a good idea," Newel kept talking continuously. [M4] "Let's play a role like we are under the attack.
S2S: [M1] he's a good man [M2] the UNK, the one who's a man who's a man and the other [M3] and the other, the one who 's a good friend [M4] he's a good man **HS2S**: [M1] i'm not sure that," he said [M2] i'm not sure that i'm not sure [M3] i'm not sure that i'm not a fool [M4] "i'm not sure that," he said **FlowNet (delta)**: [M1] he's a good man [M2] i'm not sure what to do [M3] i'm not sure that i'm not going to be a vampire [M4] he's a good man **FlowNet (disc.)**: [M1] perhaps they were not quite good, but he was not a master, and they were the most powerful [M2] the only way to do not like a little, but i' d been in the world [M3] "you're right," he said "i am not a fool you're here [M4] you're going to be a bit more than the other



(a) By paragraph lengths **(b)** By domains

Figure 3.10: Comparison (METEOR) with human performance (black bars): S2S (blue), HS2S (red), Flow:delta (yellow), and Flow:disc. (green). Best viewed in color.

coherent flow of text compared to S2S and HS2S. Surprisingly, FlowNet (discourse) enables generating more diverse sentences with a bit of coherence, because each sentence is generated based on the representation conditioned on the predicted RST discourse relation.

3.5.4 Conclusion

We explore two forms of inter-sentential relations: linguistic relation such as discourse relations and a latent representation learned from the text. The proposed models for both relations achieve significant improvements over the baselines on partial paragraph generation task.

Despite the empirical effectiveness and difference between the linguistic and latent relations, they are not directly aligned for comparison. A potential direction for future study is to directly couple them together and see whether one form contains the other, or vice versa. Another direction is to check their effectiveness on top of the recent pre-trained language models.

3.6 Goal Planning for Masked Paragraph Generation

3.6.1 Introduction

One may think text coherence can be achieved from a simple language model trained on huge data. That can be true in some scenarios (e.g., a chitchat dialogue), but neither in a long, multi-sentence generation (e.g, narrative story-telling (Chambers and Jurafsky, 2008)) nor in a coherent paragraph generation (Kang et al., 2019c). This is because such word-level predictions can not capture the general flow of textual coherence, while human designs their intents on what/how to say ahead (Byrne, 1979; McKeown, 1985; Swan, 2002) before they speak. Without such a high-level planning on a content, output text from the language model would be simply the most frequent patterns of text generalized from the training data, regardless of the intent.

Where can the model learn the long-term coherence from text? A *paragraph* (or a multi-sentence document) itself can be a pot of golden resources, containing various forms of the inductive bias. From restrictive to prescriptive level, different types of coherence in a paragraph have been studied: a sequence of words/sentences for language modeling (Devlin et al., 2019; Radford et al., 2019), a structure of phrases (e.g., discourse tree) (Kang et al., 2019c), an ordering of sentences (Chambers and Jurafsky, 2008; Barzilay and Lapata, 2008), a connection between phrases (e.g., co-references), a sequence of events (e.g., scripts (Tomkins, 1978; Schank and Abelson, 2013), plans (Appelt, 1982; Hovy, 1991)), and more. In this work, we explore the hybrid approach by combining the content planning and the surface-level language model.

Despite the recent advances of generation systems, their evaluation is very limited to measuring perplexity of the language model or applying on sub-tasks where the content to produce is fully (e.g., summarization, paraphrasing) or partially (e.g., data-to-text) provided: called *close-ended* generation. Some other tasks are also limited to classification instead of generation: narrative close task (Chambers and Jurafsky, 2008; Mostafazadeh et al., 2016), sentence ordering (Barzilay and Lapata, 2008), and next sentence prediction (Devlin et al., 2019). Recently, Kang et al. (2019c) proposed an interesting *open-ended generation task* called Bridging; given the first and the last sentences, predicting intermediate sentences between them. To extend it, we proposed a new open-ended generation task; 2$\mathbb{PM}$ (**P**artially **Mask**ed **P**aragraph) generation; predicting the masked sentences in a paragraph with a little help by a small number of keywords for selecting content of the masked text. Unlike the closed-ended tasks, the target (masked) text in 2$\mathbb{PM}$ is neither a subset nor a superset of the context (unmasked) text, but independent as a coherent form in both forward and backward ways. Thus, 2$\mathbb{PM}$ requires not only surface-level

generation but various components for text planning; content selection, content prediction, and content ordering.

To address 2PM, we take the hybrid generation approach; a combination of content selection and then surface generation motivated by the prior works (Moryossef et al., 2019; Miculicich et al., 2019; Shen et al., 2019; Hua and Wang, 2019). We propose PLANNER; an end-to-end content planning and generation with the pre-trained language models (Devlin et al., 2019; Radford et al., 2019). In particular, our content planner predicts a set of candidate keywords (i.e., pseudo plans) for the masked text to predict based on a context text, where the true keywords extracted are given during training. The pseudo plan is an approximation of intents in text planning, providing a hint to guide the generator to bridge the coherency between the context and content to say. The distribution of predicted keywords (i.e., what to say) is then combined with the word distribution from the language model.

PLANNER combines two advantages; micro-level language fluency from the extensively trained language model (bottom-up, restrictive) and macro-level content choice controlled by the high-level planning (top-down, prescriptive). PLANNER achieved significant improvements on masked sentence prediction over the baselines in both automatic and human evaluation.

3.6.2 Partially, Masked Paragraph Generation

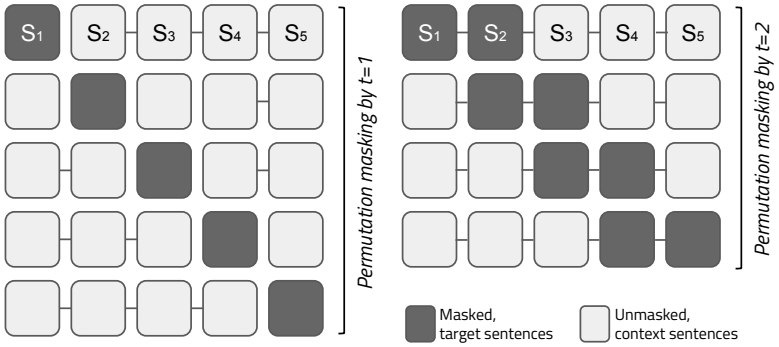
Evaluating how the model produces coherent text is very challenging due to the lack of appropriate tasks. The recently proposed task; Bridging (Kang et al., 2019c), predicts intermediate sentences of a paragraph, given the first and the last sentences. However, the task needs to understand how two extreme sentences are coherently linked and produce multiple sentences between them, making the task itself too challenging even for human[9]. Moreover, the context (i.e., first and last sentences) is too sparse to produce the multiple (from 2 to 5) intermediate text, increasing the complexity exponentially. The data usage of a paragraph is also very inefficient; training a single instance per paragraph.

To address the context sparsity and the inefficient paragraph usage, we propose a new open-ended generation task 2PM: (Partially, Masked Paragraph) generation. We describe the two-stage of how to produce the training/testing instances in PLANNER: first step is called *sentence masking with permutation* to mask the fixed-number of consecutive sentences over a paragraph with a permutation (Figure 3.11(a)), and second step is called *partial plan extraction* to provide partial information; a small number of keywords extracted from the reference, target text in order to guide content selection of generation (Figure 3.11(b)). Our work is motivated by the training technique; *word masking* for improving contextualized language modeling of BERT (Devlin et al., 2019), where 2PM extends it to *sentence-level masking* for longer coherence of text.

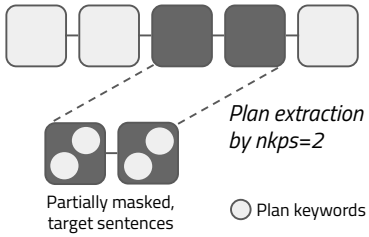
Sentence Masking with Permutation

Let t be the number of target, masked and consecutive sentences (dark-gray shaded blocks) to predict and c be the number of context, unmasked sentences given (unshaded blocks) where $l = t + c$ is the total length of a paragraph. For example in Figure 3.11, we have a l=5 paragraph

[9]The METEOR score from the human-predicted text is only about 4.5 (Kang et al., 2019c)



(a) Sentence masking with permutation: t=1 (left) or t=2 (right): One paragraph has total 5+4=9 training instances.



(b) Content plan extraction on target sentence: nkps=2

Figure 3.11: Partially, masked paragraph generation task (2$\mathbb{PM}$): predicting the masked target sentences given the unmasked context sentences, where each masked target sentence has partial information; a small number of keywords extracted from the original target sentence. (a) The number of target sentences (t) is always less than the length of context sentences (c): (t=1,c=4) (left) and (t=2,c=3) (right). (b) The maximum number of keywords per sentence ($nkps$=2) is given.

of five sentences ($s_1..s_5$). We restrict the number of context to be larger than the number of target ($t < c$) and the maximum number of $t = 3$ in order to avoid context sparsity from the bridging task (Kang et al., 2019c). For example, given a paragraph with length 5, we can produce total 5+4+3=12 training instances when t is from 1 to 3. By permuting all possible consecutive sentences masked, we can make more efficient training in data augmentation perspective.

Content Plan Extraction

Despite the restriction to the number of target sentences, generating a sentence is still difficult due to the exponential possibility of selecting content under the context given. We then provide an extra partial information to guide for content selection. This is similar to data-to-text tasks, but we only provide partial information as a few number of keywords or *pseudo plans*. The plan keywords are only provided while training the generator, but not in testing time.

In this work, we have a fundamental question of what types of plan keywords are related to the general quality of generation in terms of completeness and consistency with its context.

We extract the plan keywords using various keyword extraction algorithms: `off-the-shelf` systems, `syntactic` features, and `attention` weights. We describe each algorithm as follows:

- `Off-the-shelf` systems. There has been many automatic keyword extraction systems developed: YAKE (Campos et al., 2020) using statistical features (e.g., TF, IDF), RAKE (Rose et al., 2010) using graph-based features (e.g., word degree), and PositionRank (Florescu and Caragea, 2017) using position-based PageRank. We extract keywords for each sentence of a paragraph from the three systems using PKE tool (Boudin, 2016) and then choose the duplicate keywords by the majority vote[10].
- `Syntactic` features. Syntactic features (e.g., part-of-speech tags, named entities) are often regarded as the most salient content to complete the text. For example, script theory (Tomkins, 1978; Schank and Abelson, 2013) is an good example of how a typical sequence of events occur in a particular context where the events usually consist of actions (i.e., verbs) and objects (i.e., nouns). Using the the off-the-shelf[11] Part-of-speech (PoS) tagger, we extract three types of syntactic features: *nouns*, *verbs*, and *nouns+verbs*.
- `Attention` weights. The off-the-shelf and syntactic types of keywords are extracted from only the target sentences. However, an importance of keywords sometimes depends on *context* as well. To capture the context-aware keywords, we use BERT (Devlin et al., 2019) based keyword extractor: We encode the context and target text using the pre-trained BERT model, and then average the attention weights of context words for each target word. We only use the first head's attentions, and then average them over all 12 layers[12]. We finally choose the words with the maximum weight except for the special tokens (e.g., [CLS]) and punctuation marks.

We set the maximum number of keywords per sentence (*nkps*) to 5. Some extractors may output empty keywords so the number of keywords across the systems might be different. We restrict the keywords to be always uni-grams. In case that the keywords are not uni-grams, we split them by whitespaces and treat individual unigram as unique keywords. In case the target text is multiple sentences, we combine all keyword from the sentences together and randomly shuffle them[13].

3.6.3 Planner

We describe how our Planner works on $2\mathbb{PM}$. Following the two-stage approaches (Moryossef et al., 2019; Miculicich et al., 2019; Fu et al., 2019; Hua and Wang, 2019); content selection and then surface realization, we also develop two-level hybrid system (Figure 3.12): content planning on top of the language models (or fine-tuning the language models guided by content planning as an additional head like planner head).

Given l length of a paragraph $s_1..s_l$ where each sentence s_i consists a w_i number of words $s_i = w_{i,1}..w_{i,w_i}$, $2\mathbb{PM}$ splits it into the context sentences $\mathtt{x}{=}s_1..s_{j-1}, s_{j+t}..s_n$ and t target sentences

[10]In our experiment, the ensemble of the three systems shows better performance compared to the individual systems.

[11]`https://spacy.io/`

[12]Vig (2019) conducted various experiments of which layer and head is important for syntactic and semantic tasks, concluding no consistent conclusion though.

[13]Instead of shuffling the keywords, keeping the original keywords' order and sequentially generating target sentences can be an interesting direction.

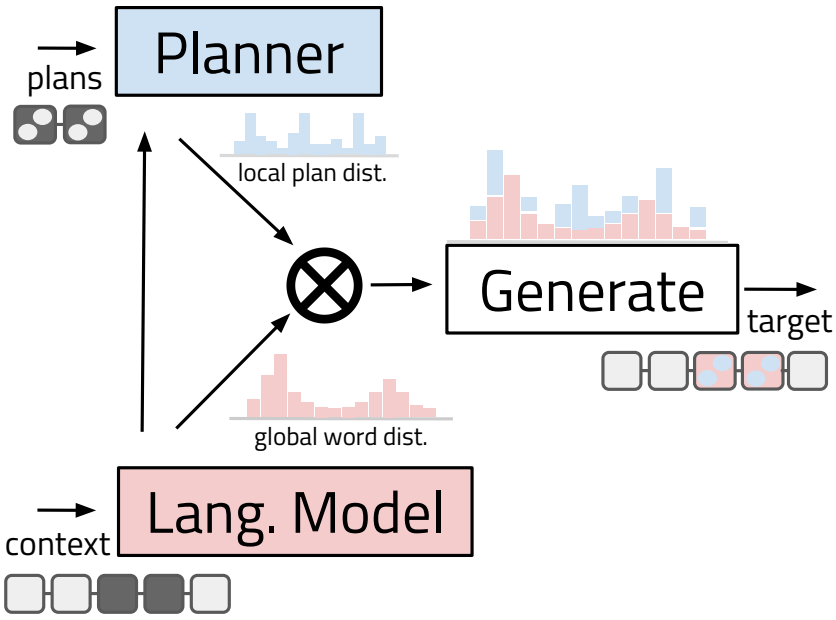


Figure 3.12: PLANNER: a combination of planner on top of the pre-trained language models for 2PM; given unmasked context, fill out the masked sentences. The planner first (1) predicts high-level plan keywords and then (2) merge its local distribution of plan keywords (blue-shaded) with the global distribution of entire vocabulary (red-shaded) from the pre-trained language model using the copy mechanism. At the training time, the ground-truth plan keywords and target sentences are given, while not in the testing time. Best viewed in color.

to predict $\mathbf{y}=s_j..s_{j+t-1}$. Each target sentence has p number of plan keywords: $k_{j,1}..k_{j,p}$ for arbitrary target sentence s_j. In this work, we let plan keywords be chosen from the entire vocabulary $\mathbb{V}^W$ instead of restrict the possible keywords by specific categories (e.g., speech acts,). By doing so, we can easily combine the two distributions from top-down content planning and bottom-up surface realization. Now, we describe how to encode context $\mathbf{x}$ from the pre-trained language models and how the probability distribution of word y_w is derived from the combination of the planner and the language model.

Pre-trained Language Models. We use two different types of language models: BERT (Devlin et al., 2019) and GPT2 (Radford et al., 2019). While GPT2 is trained on bidirectionally tied language modeling, BERT is trained on masked language modeling in addition to another head (i.e., sub-task) of predicting the next sentence. Due the different nature of BERT language modeling, we use the sequential sampling method (Wang and Cho, 2019) by using our masked target sentences as a target of its masked language modeling[14]. Using the pre-trained language models, we encode the context $\mathbf{x}$ and output the word representation $h_{j,i}$:

$$h_{j,i} = \mathtt{f}(h_{j-1,i}, \mathbf{x}_{k<(j,i)}) \tag{3.10}$$

where $\mathtt{f} \in$ {BERT,GPT2} is the transformer language model encoder and $h_{j,i}$ is its output hidden

[14]For the non-autoregressive model, we use the sequential sampling, showing much better generation quality, even though it is much slower.

state corresponding to the j^{th} word in i^{th} sentence. We then output the sentence vector h_i by averaging all word vectors in a sentence.

Sentence Position Embedding. We concatenate the encoded output state per sentence with its position embedding. The final representation of the context vector is then:

$$h^c = \frac{1}{n}\sum_i h_i; \texttt{pos}^c_i \tag{3.11}$$

where n is a number of sentences in a text and $\texttt{pos}^c$ is the position embedding of i^{th} sentence in the context paragraph. By adding the sentence position embeddings into the context encoding, the model is aware of where the contextual sentences come from. Compared to the simple concatenation of context sentences (Kang et al., 2019c), our sentence position embedding helps better model the bi-directional coherence modeling in $2\mathbb{PM}$ task.

Plan Prediction. The high-level plan can be either categorical labels defined by human (e.g., speech acts) or the same-level of lexical vocabulary as the surface-level generation. The former has a less number of conceptual categories, but it often suffers from mapping them with the vocabulary in the surface realization. This work assumes that such high-level plan consists of back-of-words (Fu et al., 2019) so the model directly predicts the plan keywords from the vocabulary used in surface realization.

We then calculate the plan probabilities over the entire vocabularies $\mathbb{V}$ given the context vector h^c and choose the p number of keywords with maximum probability estimates over vocabulary:

$$\hat{p}_{k\in\mathbb{V}} = \texttt{softmax}(h^c W^{cv}\mathbb{V}) \tag{3.12}$$

where $\mathbb{V}$ is the vocabulary from the training data and W^{cv} is the trainable model parameter. We do not control any explicit cut-off in the $p_{k\in\mathbb{V}}$ in order to make the distribution differentiable. The objective of plan prediction is then:

$$\mathbb{L}_{\texttt{plan}} = -\sum_{k\in\mathbb{V}} \log p^*_k \log \hat{p}_k \tag{3.13}$$

where the loss is calculated by the cross-entropy, $\hat{p}$ is the estimated probability distribution over vocabulary and $p*$ is the true one-hot distribution over plan keywords extracted from the extraction algorithms (i.e., [0,1..0,1] over $\mathbb{V}$).

Next Sentence Prediction. Motivated by the BERT training, we also add an auxiliary task of predicting whether the corresponding target sentence is exactly the next one or not. As negative samples, $2\mathbb{PM}$ assigns around 50% of next sentences randomly. We optimize

$$\hat{p}_{next} = softmax(W^c h^c) \tag{3.14}$$

where W^c is the trainable parameter for the binary classification. The objective of next sentence prediction is then:

$$\mathbb{L}_{\texttt{next}} = -\sum_j p^*_{next} \log \hat{p}_{next} \tag{3.15}$$

where the loss is calculated by binary cross-entropy, p^*_{next} is the true label for next sentences and $\hat{p}_{next}$ is the predicted label.

Surface Realization. We decode using the pointer network (See et al., 2017) with a copy-mechanism, but between the predicted plan distribution and vocabulary distribution. For j^{th} sentence, We learn the probability of choosing the plan keyword per each keyword k, based on the context vectors, plan keyword distributions, and sentence position embedding of target sentences $\texttt{pos}^t$:

$$\texttt{P}_{plan}(v_k) = \sigma(W^{ck}[h^c; p_k; \texttt{pos}^t_j]) \tag{3.16}$$

where σ is a sigmoid function, W^{ck} is the trainable parameter, and $\texttt{v} \in [0, 1]$ is a probability of whether choosing the plan keyword or not. We then decode each target sentence using the same Transformer decoder:

$$s_j = \texttt{g}(s_{j-1}, \hat{y}_{j-1}) \tag{3.17}$$

where $\texttt{g} \in \{BERT, GPT2\}$ is the transformer decoder and s is its output hidden state. We can obtain the attention over plan keywords k:

$$\alpha^{plan}_k = \texttt{softmax}(p_k W^{kj}[s_j; \texttt{pos}^t_j]) \tag{3.18}$$

where W^{kj} is the trainable parameter. Lastly, we combine the distribution of plan probabilities $\texttt{P}_{plan}$ and word probabilities in decoding $\texttt{P}_{lm}$.

$$P(y) = \texttt{p}_{plan} \sum_k (\alpha^{plan}_k) + (1 - \texttt{p}_{plan}) P_{lm}(y) \tag{3.19}$$

The objective of the pointer-generation is:

$$\mathbb{L}_{\texttt{gen}} = - \sum_{i \in \texttt{t}, j=1..n} P(\hat{y}_{i,j}) \log P(y^*_{i,j}) \tag{3.20}$$

Final Objective. The final goal of our training is to minimize the three objectives at the same time:

$$\mathbb{L}_{\texttt{PLANNER}} = \lambda_{plan} \mathbb{L}_{\texttt{plan}} + \lambda_{next} \mathbb{L}_{\texttt{next}} + \mathbb{L}_{\texttt{gen}} \tag{3.21}$$

where the weighting terms; λ_{plan} and λ_{next}, are obtained through the cross-validation.

3.6.4 Experiment

We have three questions to validate: Q1. Does PLANNER module help produce more coherent generation? Q2. What types of plan keywords (e.g., noun, verb, attention) are most effective in terms of generation quality? Q3. Is $2\mathbb{PM}$ a valid open-ended generation task to measure coherence of text?

dataset	D	S	P	L	T	I	K
SciFi	book	825K	174K	4.7	10M	1.6M	1.6M/10M
Fantasy	book	1.6M	352K	4.7	21M	3.2M	3.2M/19M
Romance	book	5.3M	1.1M	4.6	67M	10.4M	27M/62M
WikiText	wiki	510K	3.3M	6.5	82M	29.2M	14M/78M
CNNDM	news	12M	311K	39.3	246M	35.1M	63M/315M

Table 3.14: Data statistics: **D**omain of text, the number of **S**entences, the number of **P**aragraphs, the averaged **L**ength (number of sentences) of paragraph, the number of **T**okens, the number of training **I**nstances permuted from the paragraphs, and min/max-imum number of **K**eywords extracted.

Paragraph Dataset.

Table 3.14 shows the paragraph datasets collected in our experiment. We collect paragraphs from various domains: three most frequent sub-genres extracted from the BookCorpus (Zhu et al., 2015) dataset; SciFi, Fantasy and SciFi, wikipedia text from the wikiText-103 (Merity et al., 2016) dataset, and news articles from the preprocessed version of CNN/DailyMail (CNNDM) dataset (Kang et al., 2019d). While CNNDM and WikiText are factual, SciFi, Fantasy, and Romance are more narrative.

For a fair comparison, we restrict the number of sentences in a paragraph from 4 to 7, same as the FlowNet model (Kang et al., 2019c) setup. Since CNNDM has no specific line breakers in the document, most documents are regarded as a full paragraph (39.3 length on average). Each dataset is randomly split by 0.9/0.05/0.05 for train, valid, and test set, respectively.

Models.

As a baseline, we compare several encoder-decoder models: **BiLSTM** (Hochreiter and Schmidhuber, 1997a) and hierarchical seq2seq **HRED** (Serban et al., 2017; Sordoni et al., 2015) by encoding the concatenation of context sentences and then decoding the target sentences. We compare two strong paragraph generation models: **FlowNet**$_{disc}$ using discourse relations and **FlowNet**$_{latent}$ using latent delta relations (Kang et al., 2019c). For **FlowNet**, we use the same setups (e.g., discourse parser, hyper-parameters) as the original paper. We also compare with the pre-trained BERT or GPT2 models (**BERT**$_{pretr}$, **GPT2**$_{pretr}$) and their fine-tuned models on the training dataset of 2$\mathbb{PM}$ (**BERT**$_{finet}$, **GPT2**$_{finet}$). PLANNER is trained by either BERT or GPT2 with different types of plan keywords.

We find the best hyper-parameters on validation set using a grid search. For variants of **BERT** and **GPT2** models, we follow the default parameters used in HuggingFace's transformer models (Wolf et al., 2019). We use the uncased BERT and GPT2 models, which show comparable performance with the large models. For BERT, we use the sequential sampling method (Wang and Cho, 2019) with Nucleus sampling strategies for producing more diverse text (Holtzman et al., 2019). For a pointer-generator, we follow the default setup in (See et al., 2017). The maximum number of plan keywords per sentence is 3. For more details, see the Appendix.

	SciFi			Fantasy			Romance			WikiText			CNNDM		
Models	**B**	**M**	**VE**	**B**	**M**	**VE**	**B**	**M**	**VE**	**B**	**M**	**VE**	**B**	**M**	**VE**
BERT$_{pretr}$ (Devlin et al., 2019)	1.9	3.2	28.2	1.7	3.2	25.2	1.5	2.9	19.2	1.6	1.8	29.0	1.8	1.9	24.1
GPT2$_{pretr}$ (Radford et al., 2019)	2.0	3.7	29.5	1.9	3.3	27.1	1.5	2.9	21.6	1.9	2.0	29.5	1.8	1.9	25.8
BiLSTM	2.6	2.8	27.9	2.6	2.9	26.5	2.2	2.4	25.6	2.7	2.9	31.2	2.5	2.5	30.0
HRED (Sordoni et al., 2015)	2.8	2.8	27.8	2.8	2.9	25.2	2.4	2.5	28.2	2.8	2.7	32.4	2.8	2.9	31.9
FlowNet$_{disc}$ (Kang et al., 2019c)	3.3	3.9	40.6	3.6	4.5	41.4	3.2	3.7	38.6	3.2	3.6	38.9	3.4	3.8	40.1
FlowNet$_{latent}$ (Kang et al., 2019c)	3.1	3.3	39.2	3.7	4.5	42.8	3.1	3.6	35.2	3.1	3.5	37.5	3.3	3.7	38.7
BERT$_{finet}$	3.7	4.7	39.5	3.7	4.6	38.5	4.1	4.4	42.8	4.2	4.7	48.9	4.4	4.8	47.5
GPT2$_{finet}$	3.8	4.9	40.0	3.9	5.0	42.8	4.3	4.7	48.5	4.6	4.8	50.1	4.5	5.0	50.2
Planner (BERT)	6.3	8.4	58.1	5.7	6.7	57.0	5.9	6.8	54.0	6.1	6.4	54.3	6.4	6.9	57.0
Planner (GPT2)	**6.7**	**8.9**	**62.8**	**7.1**	**9.2**	**69.5**	**7.2**	**8.1**	**73.9**	**7.6**	**7.7**	**66.8**	**6.9**	**7.8**	**59.9**
Planner (GPT2) w $\hat{p}$	10.2	12.6	79.3	11.1	11.8	79.6	12.7	13.3	84.4	12.5	13.0	87.8	12.1	12.9	84.9

Table 3.15: Automatic evaluation on generation in $2\mathbb{PM}$. **B** is BLEU, **M** is METEOR, and **VE** is vector extrema. For all metrics, the higher the better. Planner used keywords from the off-the-shelf system for training. $\hat{p}$ is the ground-truth plan keywords extracted the off-the-shelf system used for testing. Note that {BERT,GPT2}$_{pretr}$ do not use the training data from $2\mathbb{PM}$.

Metrics.

We evaluate our models using automatic metrics and human evaluation. For automatic metrics, we measure two hard metrics: BLEU (Papineni et al., 2002) and METEOR (Banerjee and Lavie, 2005). We also use an embedding similarity-based soft metric to measure a semantic similarity: vector extrema (VE) (Liu et al., 2016).

For human evaluation, we measure fluency, coherence w.r.t. context, and overall quality (scored between 1 and 5), where the score of each sample is averaged by three annotators. We use the 100 random test samples from `Roamnce`, `WikiText`, and `CNNDM` per each (e.g., total 300 different paragraphs). We also measure how human performs on the task by asking one annotator to predict the masked text in these 300 paragraphs.

Evaluation on Generation

Table 3.15 shows an automatic evaluation on $2\mathbb{PM}$ task. The pretrained {BERT,GPT2}$_{pretr}$ models themselves produce natural text but have poor performance in our task, showing the effectiveness of $2\mathbb{PM}$ task to measure textual coherence w.r.t the context. The fine-tuned models and FlowNet models show significant improvements over the encoder-decoder baselines by large margins. In all datasets, Planner shows significant improvements on both hard and soft metrics. This shows the importance of high-level plan prediction ahead from the context before starting generation. Interestingly, Planner with GPT2 outperforms Planner with BERT, because GPT2 is trained for left-to-right language model but BERT for masked language model. Finally, in order to present the upper bound of Planner model, we show the performance of Planner with the true keywords ($\hat{p}$) extracted from the off-the-shelf extractor, achieving the dramatic performance gain.

	Romance			WikiText			CNNDM		
Models	**F**	**C**	**Q**	**F**	**C**	**Q**	**F**	**C**	**Q**
GPT2$_{finet}$	4.4	2.1	3.6	3.9	1.9	1.9	3.8	1.6	1.8
PLANNER	4.2	3.8	3.9	3.8	3.6	3.8	3.6	3.1	3.2
PLANNER W $\hat{p}$	4.1	4.6	4.3	3.8	4.1	4.0	4.0	4.4	4.1
Human	4.8	4.9	4.9	4.6	4.5	4.5	4.5	4.4	4.4

Table 3.16: Human evaluation on generation in $2\mathbb{PM}$. **F** is fluency, **C** is coherence with context, and **Q** is overall quality. Each metric is scaled out of 5. **PLANNER** only used GPT2 with off-the-shelf keywords.

Table 3.16 shows a human evaluation on different systems and human-generated sentences. The fine-tuned GPT2 model shows high fluency but extremely low coherence because $2\mathbb{PM}$ requires not only fluent and natural output but also context-aware generation. PLANNER shows much higher coherence and overall quality but still is far behind the PLANNER with the true keywords and human generation.

Next sentence and Plan Prediction

We measure the performance (i.e., accuracy) of each module in PLANNER: next sentence prediction (NSP) and plan prediction (PP) on the test samples. Our PLANNER achieves very high accuracy for the NSP as expected. Surprisingly, however, for the PP, PLANNER predicts almost the half of the keywords correctly from the huge number of candidate keywords (i.e., entire vocabulary size). This indicates that PLANNER can capture certain consistency between context and the selected keywords during training, then predict the right keywords when it detects a similar patterns in testing. We find the similar performances among the datasets.

	Romance		WikiText		CNNDM	
Models	**NSP**	**PP**	**NSP**	**PP**	**NSP**	**PP**
PLANNER	91.6	48.1	92.7	50.2	90.7	49.4

Table 3.17: Accuracies of each module in PLANNER. **NSP** is accuracy of next sentence prediction, and **PP** is accuracy of plan prediction.

Ablation

Table 3.18 shows various ablation tests. Each module; sentiment positional embedding, plan prediction, and next sentence prediction helps improve the overall performance (Table 3.18(top)). In particular, plan prediction helps the most; improvement on ~2.1% in METEOR and ~13.8% on VE. Among the different types of keywords used in training (middle), the syntactic keywords of nouns+verbs and the keywords extracted from the off-the-shelf algorithm outperform the other

	Romance		WikiText		CNNDM	
Models	**M**	**VE**	**M**	**VE**	**M**	**VE**
PLANNER	8.1	73.9	7.7	66.8	7.8	59.9
w/o Sent. Position	-1.4	-8.5	-1.7	-9.7	-1.1	-6.2
w/o Plan Predict	-2.1	-12.7	-2.1	-13.9	-2.0	-13.0
w/o Next Predict.	-0.2	-2.9	-0.6	-3.2	-0.6	-4.7
PLANNER trained by:						
w Random	6.4	65.2	6.2	53.9	5.9	42.9
w Syntac(Verb)	7.8	73.7	7.5	62.8	7.6	54.6
w Syntac(Noun)	7.6	71.3	7.5	61.5	7.5	53.8
w Syntac(N+V)	**8.3**	**74.5**	**7.8**	65.9	**7.9**	58.6
w Off-the-shelf	8.1	73.9	7.7	**66.8**	7.8	**59.9**
w Attention	7.9	72.4	7.4	63.0	7.6	55.4
PLANNER tested with different plan keywords						
w Random	5.7	52.9	5.8	55.0	5.9	58.9
w predicted	8.1	73.9	7.7	66.8	7.8	59.9
w Off-the-shelf ($\hat{p}$)	13.3	84.4	13.0	87.8	12.9	85.9

Table 3.18: Ablation on PLANNER's modules (top), plan types for training (middle), and plan types for testing (bottom).

types. Attention keywords do not improve the models' performance much because the averaged attention weights themselves may not be good explanation of the features' salience. At testing (bottom), the predicted keywords from PLANNER shows dramatic improvements over the random keywords, but far behind the case where the true keywords of the target sentences are explicitly given[15].

Table 3.19 shows an example paragraph with keywords extracted from different algorithms for training instances in 2$\mathbb{PM}$ and predicted target sentences from PLANNER and a human.

3.6.5 Conclusion

We propose a partially, masked paragraph generation task to measure text coherence of a long document. The hybrid combination of high-level content planning ahead surface realization produce more coherent output text 2$\mathbb{PM}$ than other language model baselines.

[15]The true keywords are extracted from the off-the-shelf algorithm, being regarded as the upper bound performance.

Task: given context sentences [0,1,5], predict target sentences [2,3,4]

[0] ”They reached the raised sanctuary with the slab-marble altar and the tall-backed cathedra , the bishop ’ s seat .” [1] ”Vigor and his niece made the sign of the cross .” [2] ”Vigor dropped to one knee , then got up .” [3] ”He led them through a gate in the chancel railing .” [4] ”Beyond the railing , the altar was also marked in chalk , the travertine marble stained .” [5] ”Police tape cordoned off a section to the right .”

Plan keywords extracted by systems in 2PM

`Off-the-shelf`	[2][”vigor”,”dropped”,”one”], [3][”chancel”,”railing”,”led”], [4][”travertine”,”marble”,”stained”]
`Syntactic` (noun)	[2][”vigor”,”knee”], [3][”gate”,”chancel”], [4][”railing”,”altar”, ”chalk”]
`Syntactic` (verb)	[2][”dropped”, ”got”], [3][”led”,”railing”], [4][”marked”, ”stained”]
`Syntactic` (nounverb)	[2][”vigor”,”dropped”, ”knee”], [3][”led”,”gate”,”chancel”], [4][”railing”,”altar”,”marked”]
`Attention`	[2][”vigor”,”dropped”,”got”], [3][”led”,”gate”,”railing”], [4][”altar”,”chalk”,”travertine”]

Plan and target text predicted by PLANNER (human evaluation result: **F : 4.3, C: 3.9, Q: 3.8**)

Plans: [2][”vigor”,”mark”,”caught”], [3][”gate”,”catholics”,”police”], [4][”altar”, ”mark”, ”bishop”]
Text: [2] “vigor continuously walked down the road .” [3] ”he opened the gate which has a sign of Catholics .” [4] ”both bishop and vigor met a police officer .”

Plan and target text predicted by human writer (human evaluation result: **F : 4.8, C: 4.9, Q: 4.8**)

Plans: [2] [“vigor”,“show”,“sanctuary”], [3] [“altar”,“blood”,“trace”], [4] [“kill”,“sacrifice”,“recently”]
Text: [2] “Then vigor showed around the sanctuary to them.” [3] “In there, they found a trace of the blood on the altar.” [4] “They thought that recently the sacrifice was killed in here.”

Table 3.19: Example paragraph with the plan keywords extracted from different algorithms and output predictions by PLANNER and human.

3.7 Policy Planning for Goal-oriented Recommendation Dialogue

3.7.1 Introduction

Traditional recommendation systems factorize users’ historical data (i.e., ratings on movies) to extract common preference patterns (Koren et al., 2009; He et al., 2017b). However, besides making it difficult to accommodate new users because of the *cold-start problem*, relying on aggregated history makes these systems static, and prevents users from making specific requests, or exploring a temporary interest. For example, a user who usually likes horror movies, but is in the mood for a fantasy movie, has no way to indicate their preference to the system, and would likely get a recommendation that is not useful. Further, they cannot iterate upon initial recommendations with clarifications or modified requests, all of which are best specified in natural language.

Recommending through dialogue interactions (Reschke et al., 2013; Wärnestål, 2005) offers a promising solution to these problems, and recent work by Li et al. (2018) explores this approach in detail. However, the dataset introduced in that work does not capture higher-level strategic be-

haviors that can impact the quality of the recommendation made (for example, it may be better to elicit user preferences first, before making a recommendation). This makes it difficult for models trained on this data to learn optimal recommendation strategies. Additionally, the recommendations are not grounded in real observed movie preferences, which may make trained models less consistent with actual users. This paper aims to provide *goal-driven recommendation dialogues grounded in real-world data*. We collect a corpus of goal-driven dialogues grounded in real user movie preferences through a carefully designed gamified setup (see Figure 3.13) and show that models trained with that corpus can learn a successful recommendation dialogue strategy. The training is conducted in two stages: first, a *supervised* phase that trains the model to mimic human behavior on the task; second, a *bot-play* phase that improves the goal-directed strategy of the model.

The contribution of this work is thus twofold. (1) We provide the first (to the best of our knowledge) large-scale goal-driven recommendation dialogue dataset with specific goals and reward signals, grounded in a real-world knowledge base. (2) We propose a two-stage recommendation strategy learning framework and empirically validate that it leads to better recommendation conversation strategies.

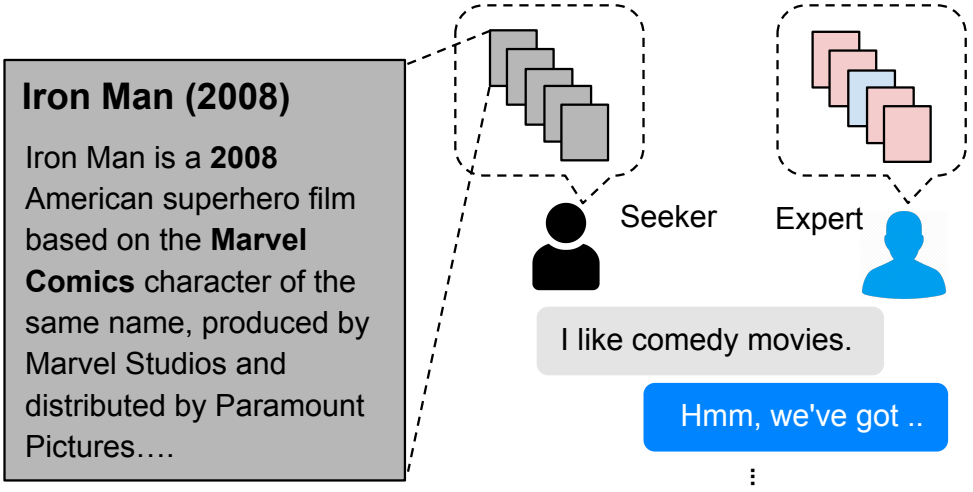


Figure 3.13: Recommendation as a dialogue game. We collect 81,260 recommendation utterances between pairs of human players (experts and seekers) with a collaborative goal: the expert must recommend the correct (blue) movie, avoiding incorrect (red) ones, and the seeker must accept it. A chatbot is then trained to play the expert in the game.

3.7.2 Recommendation Dialogue Task Design

In this section, we first describe the motivation and design of the dialogue-based recommendation game that we created. We then describe the data collection environment and present detailed dataset statistics.

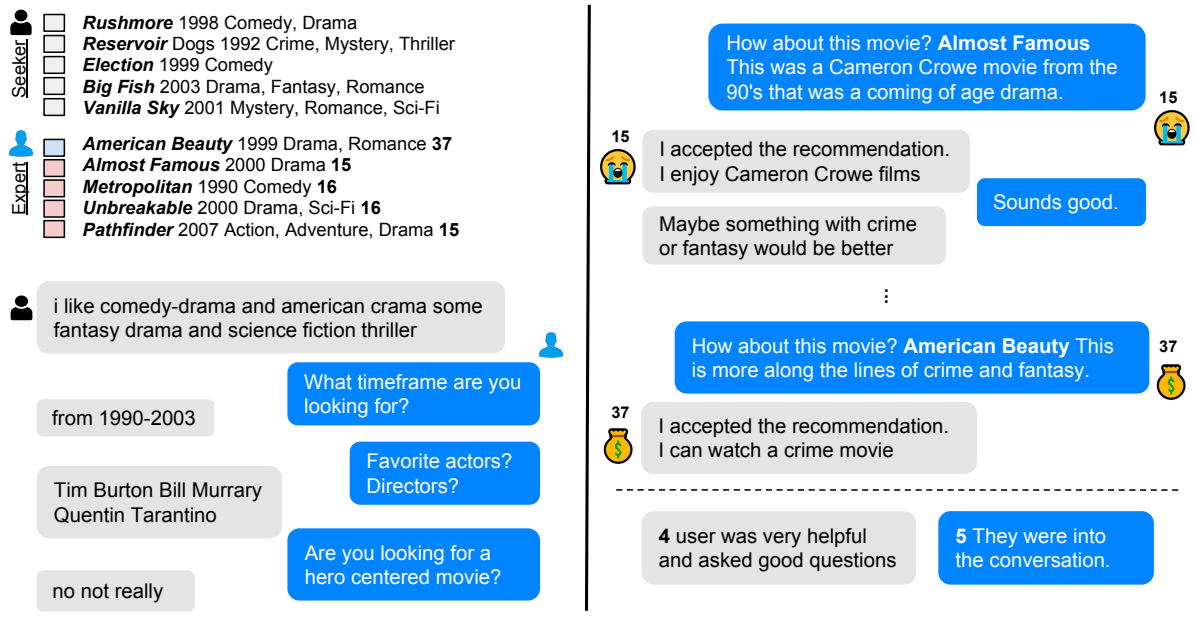


Figure 3.14: An example dialogue from our dataset of movie recommendation between two human workers: seeker (grey) and expert (blue). The goal is for the expert to find and recommend the correct movie (light blue) out of incorrect movies (light red) which is similar to the seeker movies. Best viewed in color.

Dialogue Game: Expert and Seeker

The game is set up as a conversation between a *seeker* looking for a movie recommendation, and an *expert* recommending movies to the seeker. Figure 3.14 shows an example movie recommendation dialogue between two-paired human workers on Amazon Mechanical Turk.

Game Setting.

Each worker is given a set of five movies[16] with a description (first paragraph from the Wikipedia page for the movie) including important features such as director's name, year, and genre. The seeker's set represents their watching history (movies they are supposed to have liked) for the game's sake. The expert's set consists of candidate movies to choose from when making recommendations, among which only one is the *correct* movie to recommend. The correct movie is chosen to be similar to the seeker's movie set (see Sec. 3.7.2), while the other four movies are dissimilar. The expert is not told by the system which of the five movies is the correct one. The expert's goal is to find the correct movie by chatting with the seeker and recommend it after a minimal number of dialogue turns. The seeker's goal is to accept or reject the recommendation

[16]We deliberately restricted the set of movies to make the task more tractable. One may argue that the expert can simply ask these candidates one by one (at the cost of low engagingness). However, this empirically doesn't happen: experts make on average only 1.16 incorrect movie recommendations.

from the expert based on whether they judge it to be similar to their set. The game ends when the expert has recommended the correct movie. The system then asks each player to rate the other for engagingness.

Justification.

Players are asked to provide reasons for recommending, accepting, or rejecting a movie, so as to get insight into human recommendation strategies[17].

Gamification.

Rewards and penalties are provided to players according to their decisions, to make the task more engaging and incentivize better strategies. Bonus money is given if the expert recommends the correct movie, or if the seeker accepts the correct movie or rejects an incorrect one.

Picking Expert and Seeker movie sets

This section describes how movie sets are selected for experts and seekers.

Pool of movies

To reflect movie preferences of real users, our dataset uses the MovieLens dataset[18], comprising 27M ratings applied to 58K movies by 280K real users. We obtain descriptive text for each movie from Wikipedia[19] (i.e., the first paragraph). We also extract entity-level features (e.g., directors, actors, year) using the MovieWiki dataset (Miller et al., 2016) (See Figure 3.13). We filter out less frequent movies and user profiles (see Appendix), resulting in a set of 5,330 movies and 65,181 user profiles with their ratings.

Movie similarity metric

In order to simulate a natural setting, the movies in the seeker's set should be similar to each other, and the correct movie should be similar to these, according to a metric that reflects coherent empirical preferences. To compute such a metric, we train an embedding-driven recommendation model (Wu et al., 2018).[20] Each movie is represented as an embedding, which is trained so that embeddings of movies watched by the same user are close to each other. The closeness metric between two movies is the cosine similarity of these trained embeddings. A movie is deemed close to a set of movies if its embedding is similar to the average of the movie embeddings in the set.

[17]Our model doesn't utilize this or the engagingness scores for learning, but these are potential future directions.

[18]`https://grouplens.org/datasets/movielens/`

[19]`https://dumps.wikimedia.org/`

[20]We also tried a classical matrix-factorization based recommendation model, which shows comparable performance to the embedding model.

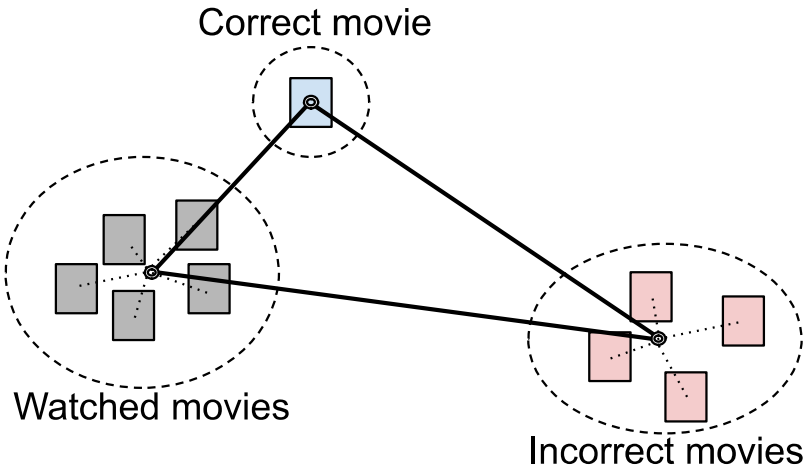


Figure 3.15: Movie set selection: watched movies for seeker (grey) and correct (light blue) / incorrect (light red) movies for expert.

Movie Set Selection

Using these trained embeddings, we design seeker and expert sets based on the following criteria (See Figure 3.15):

- Seeker movies (grey) are a set of five movies which are close to each other, chosen from the set of all movies watched by a real user.
- The correct movie (light blue) is close to the average of the five embeddings of the seeker set.
- The expert's incorrect movies (light red) are far from the seeker set and the correct movie.

We filter out movie sets that are too difficult or easy for the recommendation task (see Appendix), and choose 10,000 pairs of seeker-expert movie sets at random.

Data Collection

For each dialogue game, a movie set is randomly chosen without duplication. We collect dialogues using ParlAI (Miller et al., 2017) to interface with Amazon Mechanical Turk. More details about data collection are included in the Appendix.

Table 3.20 shows detailed statistics of our dataset regarding the movie sets, the annotated dialogues, actions made by expert and seeker, dialogue games, and engagingness feedback.

The collected dialogues contain a wide variety of action sequences (recommendations and accept/reject decisions). Experts make an average of 1.16 incorrect recommendations, which indicates a reasonable difficulty level. Only 37.6% of dialogue games end at first recommendation, and 19.0% and 10.8% at second and third recommendations, respectively.

Figure 3.16 shows histogram distributions of (a) expert's decisions between speaking utterance and recommendation utterance and (b) correct and incorrect recommendations over the normalized turns of dialogue. In (a), recommendations increasingly occur after a sufficient number of speaking utterances. In (b), incorrect recommendations are much more frequent earlier in the dialogue, while the opposite is true later on.

Dialogue statistics	
Number of dialogues	9,125
Number of utterances	170,904
Number of unique utterances	85,208
Avg length of a dialogue	23.0
Avg duration (minutes) of a dialogue	5.2
Expert's utterance statistics	
Avg utterance length	8.40
Unique tokens	11,757
Unique utterances	40,550
Seeker's utterance statistics	
Avg utterance length	8.47
Unique tokens	10,766
Unique utterances	45,196
Action statistics (all scores are averaged)	
# of correct/incorrect recs. by expert	1.0 / 1.16
# of correct/incorrect decisions by seeker	1.1 / 1.04
Game statistics (all scores are averaged)	
min/max movie scores	12.3 / 46.0
correct/incorrect movies	39.9 / 15.0
real game score by expert/seeker	61.3 / 50.8
random game score by expert/seeker	43.2 / 38.1
Engagingness statistics (all scores are averaged)	
engagingness score by expert/seeker	4.3 / 4.4
engagingness scores & feedback collected	18,308

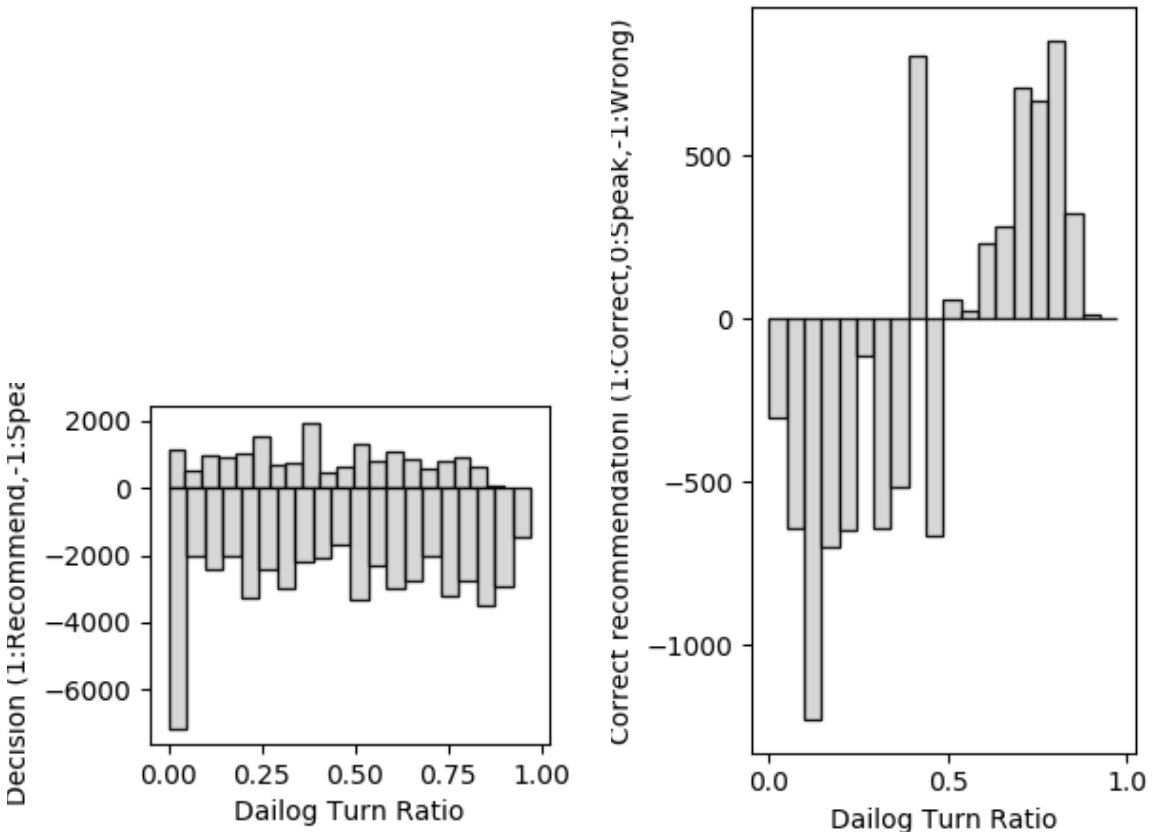
Table 3.20: Data statistics. "correct/incorrect" in the action stats means that the expert recommends the correct/incorrect movie or the seeker correctly accepts/rejects the movie.

3.7.3 Our Approach

In order to recommend the right movie in the role of the expert, a model needs to combine several perceptual and decision skills. We propose to conduct learning in two stages (See Figure 3.17): *supervised multi-aspect learning* and *bot-play*.

Supervised Multi-Aspect Learning

The supervised stage of training the expert model combines three sources of supervision, corresponding to the three following subtasks: (1) GENERATE dialogue utterances to speak with the seeker in a way that matches the utterances of the human speaker, (2) PREDICT the correct movie based on the dialogue history and the movie description representations, and (3) DECIDE whether



(a) Decision to speak (-1) or recommend (+1)

(b) Correct (+1) or incorrect (-1) recommendations

Figure 3.16: Histogram distribution of (a) experts' decisions of whether to speak or recommend and (b) correct/incorrect recommendations over the normalized dialogue turns.

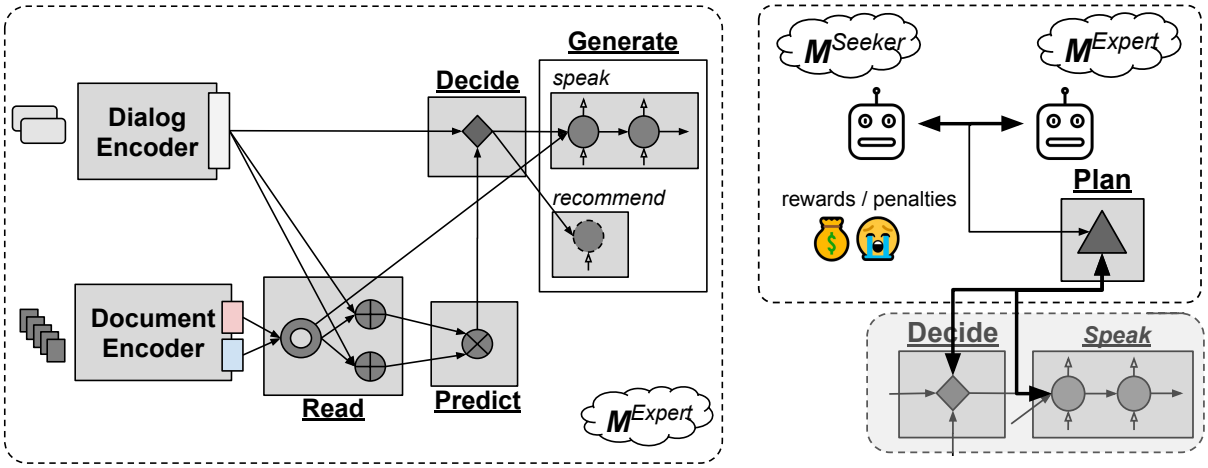
to recommend or speak in a way that matches the observed decision of the human expert.

Using an LSTM-based model (Hochreiter and Schmidhuber, 1997a), we represent the dialogue history context h_t of utterances x_1 to x_t as the average of LSTM representations of $x_1, \cdots, x_t$, and the description m_k of the k-th movie as the average of the bag-of-word representations[21] of its description sentences. Let (x_{t+1}, y, d_{t+1}) denote the ground truth next utterance, correct movie index, and ground truth decision at time $t+1$, respectively. We cast the supervised problem as an end-to-end optimization of the following loss:

$$\mathcal{L}_{sup} = \alpha\mathcal{L}_{gen} + \beta\mathcal{L}_{predict} + (1-\alpha-\beta)\mathcal{L}_{decide}, \tag{3.22}$$

where α and β are weight hyperparameters optimized over the validation set, and $\mathcal{L}_{predict}$, $\mathcal{L}_{decide}$, $\mathcal{L}_{gen}$

[21] We empirically found that BOW works better than other encoders such as LSTM in this case.



(a) Supervised multi-aspect learning

(b) Bot-play

Figure 3.17: (a) Supervised learning of the expert model $\mathcal{M}^{expert}$ and (b) bot-play game between the expert $\mathcal{M}^{expert}$ and the seeker $\mathcal{M}^{seeker}$ models. The former imitates multiple aspects of humans' behaviors in the task, while the later fine-tunes the expert model w.r.t the game goal (i.e., recommending the correct movie).

are negative log-likelihoods of probability distributions matching each of the three subtasks:

$$\mathcal{L}_{gen} = -\log p_{gen}(x_{t+1}|h_t, m_1, \cdots, m_K), \tag{3.23}$$

$$\mathcal{L}_{predict} = -\log p(y|c_1, \cdots, c_K), \quad \text{where} \tag{3.24}$$

$$c_j = h_t \cdot m_j \qquad \text{for } j \in 1..K, \tag{3.25}$$

$$\mathcal{L}_{decide} = p_{MLP}(d_{t+1}|h_t, c_1, \cdots, c_K), \tag{3.26}$$

with p_{gen} the output distribution of an attentive seq2seq generative model (Bahdanau et al., 2015), p a softmax distribution over dot products $h_t \cdot m_k$ that capture how aligned the dialogue history h_t is with the description m_k of the k-th movie, and p_{MLP} the output distribution of a multi-layer perceptron predictor that takes $c_1, \cdots, c_K$ as inputs[22].

Bot-Play

Motivated by the recent success of self-play in strategic games (Silver et al., 2017; Vinyals et al., 2019; OpenAI, 2018) and in negotiation dialogues (Lewis et al., 2017), we show in this section how we construct a reward function to perform bot-play between two bots in our setting, with the aim of developing a better expert dialogue agent for recommendation.

PLAN

optimizes long-term policies of the various aspects over multiple turns of the dialogue game by maximizing game-specific rewards. We first pre-train expert and seeker models individually:

[22]We experimented with various other encoding functions, detailed in the Appendix.

the expert model $\mathcal{M}^{expert}(\theta) = \min_\theta \mathcal{L}_{sup}$ is pre-trained by minimizing the supervised loss in Eq 3.22, and the seeker model $\mathcal{M}^{seeker}(\phi)$ is a retrieval-based model that retrieves seeker utterances from the training set based on cosine similarity of the preceding dialogue contexts encoded using the BERT pre-trained encoder[23]. θ and ϕ are model parameters of the expert and seeker model, respectively. Then, we make them chat with each other, and fine-tune the expert model by maximizing its reward in the game (See Figure 3.17, Right).

The dialogue game ends if the expert model recommends the correct movie, or a maximum dialogue length is reached[24], yielding T turns of dialogue; $g = (x_1^{expert}, x_1^{seeker}..x_T^{expert}, x_T^{seeker})$. Let T_{REC} the set of turns when the expert made a recommendation. We define the expert's reward as:

$$r_t^{expert} = \frac{1}{|T_{REC}|} \cdot \sum_{t \in T_{REC}} \delta^{t-1} \cdot b_t, \tag{3.27}$$

where δ is a discount factor[25] to encourage earlier recommendations, b_t is the reward obtained at each recommendation made, and $|T_{REC}|$ is the number of recommendations made. b_t is 0 unless the correct movie was recommended.

We define the reward function $\mathcal{R}$ as follows:

$$\mathcal{R}(x_t) = \sum_{x_t \in X^{expert}} \gamma^{T-t}(r_t^{expert} - \mu) \tag{3.28}$$

where $\mu = \frac{1}{t}\sum_{1..t} r_t^{expert}$ is the average of the rewards received by the expert until time t and γ is a discount factor to diminish the reward of earlier actions. We optimize the expected reward for each turn of dialogue x_t and calculate its gradient using REINFORCE (Williams, 1992b). The final role-playing objective $\mathcal{L}_{RP}$ is:

$$\nabla \mathcal{L}_{RP}(\theta; \mathrm{z}) = \sum_{x_t \in X^{expert}} \mathbb{E}_{x_t}[\nabla \log p(x_t|x_{<t})\mathcal{R}(x_t)] \tag{3.29}$$

We optimize the role-playing objective with the pre-trained expert model's decision ($\mathcal{L}_{decide}$) and generation ($\mathcal{L}_{gen}$) objectives at the same time. To control the variance of the RL loss, we alternate optimizing the RL loss and other two supervised losses for each step. We do not fine-tune the prediction loss, in order not to degrade the prediction performance during bot-play.

3.7.4 Experiments

We describe our experimental setup in §3.7.4. We then evaluate our supervised and unsupervised models in §3.7.4 and §3.7.4, respectively.

[23] See Sec. 3.7.4 for details on BERT. We also experimented with sequence-to-sequence models for modeling the seeker but performance was much worse.

[24] We restrict the maximum length of a dialogue to 20.

[25] we use $\delta = 0.5$.

		Generation		Recommendation				Decision
		F1	BLEU	Turn@1	Turn@3	Chat@1	Chat@3	Acc
Baseline	TFIDF-RANKER	32.5	**27.8**	-	-	-	-	-
	BERT-RANKER	38.3	23.9	-	-	-	-	
	RANDOM RECC.	3.6	0.1	21.3	59.2	23.1	62.2	-
	BERT RECC.	16.5	0.2	25.5	66.3	26.4	68.3	-
Ours	GENERATE	39.5	26.0	-	-	-	-	-
	+PREDICT	40.2	26.4	76.4	96.9	75.7	97.0	-
	+DECIDE	**41.0**	27.4	**77.8**	**97.1**	**78.2**	**97.7**	**67.6**
	+PLAN	40.9	26.8	76.3	95.7	77.5	97.6	53.6

Table 3.21: Evaluation on supervised models. We incrementally add different aspects of modules: GENERATE, PREDICT, and DECIDE for supervised multi-aspect learning and PLAN for bot-play fine-tuning.

Setup

We select 5% of the training corpus as validation set in our training.

All hyper-parameters are chosen by sweeping different combinations and choosing the ones that perform best on the validation set. In the following, the values used for the sweep are given in brackets. Tokens of textual inputs are lower-cased and tokenized using byte-pair-encoding (BPE) (Sennrich et al., 2016) or the Spacy[26] tokenizer. The seq-to-seq model uses 300-dimensional word embeddings initialized with GloVe (Pennington et al., 2014) or Fasttext (Joulin et al., 2016) embeddings, $[1, 2]$ layers of $[256, 512]$-dimensional Uni/Bi-directional LSTMs (Hochreiter and Schmidhuber, 1997a) with 0.1 dropout ratio, and soft attention (Bahdanau et al., 2015). At decoding, we use beam search with a beam of size 3, and choose the maximum likelihood output. For each turn, the initial movie text and all previous dialogue turns including seeker's and expert's replies are concatenated as input to the models.

Both supervised and bot-play learning use Adam (Kingma and Ba, 2015) optimizer with batch size 32 and learning rates of $[0.1, 0.01, 0.001]$ with 0.1 gradient clipping. The number of softmax layers (Yang et al., 2018) is $[1, 2]$. For each turn, the initial movie description and all previous dialogue utterances from the seeker and the expert are concatenated as input text to the other modules. Each movie textual description is truncated at 50 words for efficient memory computation.

We use annealing to balance the different supervised objectives: we only optimize the GENERATE loss for the first 5 epochs, and then gradually increase weights for the PREDICT and DECIDE losses. We use the same movie-sets as in the supervised phase to fine-tune the expert model. Our models are implemented using PyTorch and ParlAI (Miller et al., 2017). Code and dataset will be made publicly available through ParlAI[27].

[26] https://spacy.io/

[27] https://github.com/facebookresearch/ParlAI

Evaluation of Supervised Models

Metrics.

We first evaluate our supervised models on the three supervised tasks: dialogue generation, movie recommendation, and per-turn decision to speak or recommend. The dialogue generation is evaluated using the F1 score and BLEU (Papineni et al., 2002) comparing the predicted and ground-truth utterances. The F1 score is computed at token-level. The recommendation model is evaluated by calculating the percentage of times the correct movie is among the top k recommendations (hit@k). In order to see the usefulness of dialogue for recommendation, precision is measured per each expert turn of the dialogue (`Turn@k`) regardless of the decision to speak or recommend, and at the end of the dialogue (`Chat@k`).

Models.

We compare our models with Information Retrieval (IR) based models and recommendation-only models. The IR models retrieve the most relevant utterances from the set of candidate responses of the training data and rank them by comparing cosine similarities using TFIDF features or BERT (Devlin et al., 2019) encoder features. Note that IR models make no recommendation. The recommendation-only models always produce recommendation utterances following the template (e.g., "how about this movie, `[MOVIE]`?") where the `[MOVIE]` is chosen randomly or based on cosine similarities between dialogue contexts and the text descriptions of candidate movies. We use the pre-trained BERT encoder (Devlin et al., 2019) to encode dialogue contexts and movie text descriptions.

We incrementally add each module to our base GENERATE model: PREDICT and DECIDE for supervised learning and PLAN for bot-play fine-tuning. Each model is chosen from the best model in our hyper-parameter sweeping.

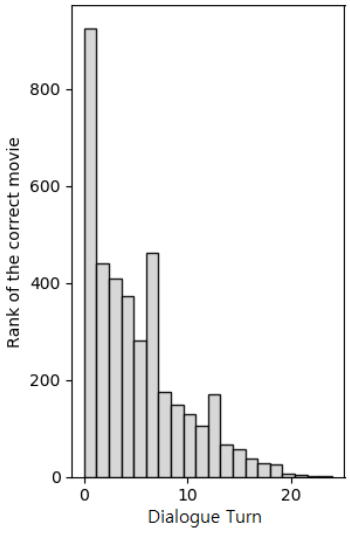
Results.

Table 3.21 shows performance comparison on the test set. Note that only the full supervised model (+DECIDE) and the fine-tuned model (+PLAN) can appropriately operate every function required of an expert agent such as producing utterances, recommending items, and deciding to speak or recommend.

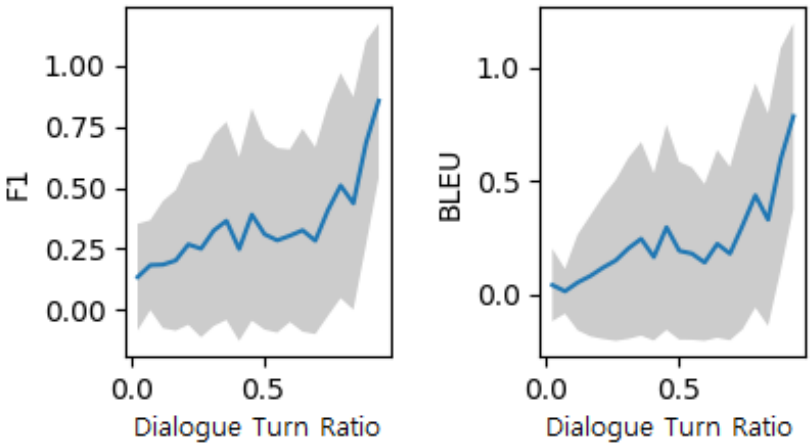
Compared to recommendation-only models, our prediction PREDICT modules show significant improvements over the recommendation baselines on both per-turn and per-chat recommendations: 52% on `Turn@1` and 34% on `Turn@3`. `Chat` scores are always higher than `Turn`, indicating that recommendations get better as more dialogue context is provided. The DECIDE module yields additional improvements over the PREDICT model in both generation and recommendation, with 67.6% decision accuracy, suggesting that the supervised signal of decisions to speak or recommend can contribute to better overall representations.

In generation, our proposed models show comparable performance as the IR baseline models (e.g., BERTRANKER). The +DECIDE model improves on the F1 generation score because it learns when to predict the templated recommendation utterance.

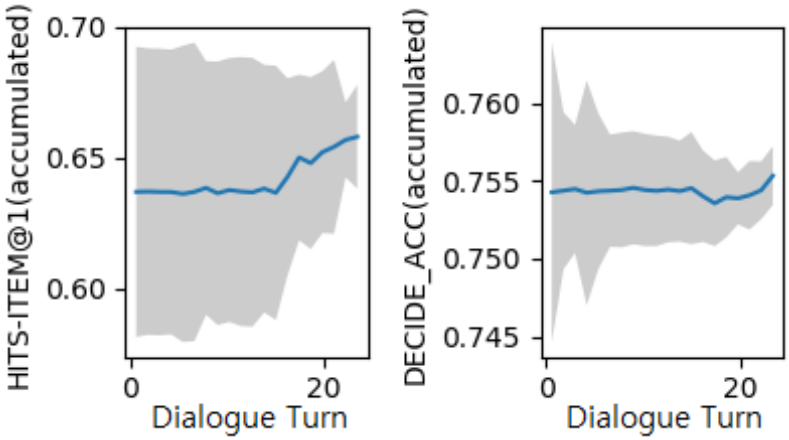
As expected, +PLAN slightly hurts most metrics of supervised evaluation, because it optimizes a different objective (the game objective), which might not systematically align with the supervised metrics. For example, a system optimized to maximize game objective should try to avoid incorrect recommendations even if humans made them. Game-related evaluations are shown in §3.7.4.



(a) Rank of recommendation



(b) F1/BLEU over dialogue turn ratio



(c) Turn@1 /Decision Acc over dialogue turns

Figure 3.18: Analysis of the expert's model: as the dialogue continues (x-axis is either fraction of the full dialogue, or index of dialogue turn), y-axis is (a) rank of the correct recommendation (the lower rank, the better) and (b,c) F1/BLEU/Turn@1/Decision Accuracy (the higher the better) with the variance shown in grey.

Analysis

We analyze how each of the supervised modules acts over the dialogue turns on the test set. Figure 3.18(a) shows a histogram of the rank of the ground-truth movie over turns. The rank of the model's prediction is very high for the first few turns, then steadily decreases as more utterances are exchanged with the seeker. This indicates that the dialogue context is crucial for finding good recommendations.

The evolution of generation metrics (`F1`, `BLEU`) for each turn is shown in Fig. 3.18(b), and the (accumulated) recommendation and decision metrics (`Turn@1`/`Accuracy`) in Fig. 3.18(c)[28]. The accumulated recommendation and decision performance sharply rises at the end of the dialogue and variance decreases. The generation performance increases, because longer dialogue contexts helps predict the correct utterances.

Evaluation on Dialogue Games

Metrics.

In the bot-play setting, we provide game-specific measures as well as human evaluations. We use three automatic game measures: `Goal` to measure the ratio of dialogue games where the goal is achieved (i.e., recommending the correct movie or not), `Score` to measure the total game score, and `Turn2G` to count the number of dialogue turns taken until the goal is achieved.

We conduct human evaluation by making the expert model play with human seekers. We measure automatic metrics as well as dialogue quality scores provided by the player: fluency, consistency, and engagingness (scored between 1 and 5) (Zhang et al., 2018). We use the full test set (i.e., 911 movie sets) for bot-bot games and use 20 random samples from the test set for {bot,human}-human games.

Models.

We compare our best supervised model with several variants of our fine-tuned bot-play models. We consider bot-play of an expert model with different seeker models such as BERT-Ranker based seeker and Seq-to-Seq based seeker. Each bot-play model is trained on the same train set that is used for training the original supervised model. The seeker model uses retrieval based on BERT pretrained representations of dialogue context (`BERT-R`) [29].

Results.

Compared to the supervised model, the self-supervised model fine-tuned by seeker models shows significant improvements in the game-related measures. In particular, the `BERT-R` model shows a +27.7% improvement in goal success ratio. Interestingly, the number of turns to reach the

[28]For better understanding of the effect of recommendation and decision, we show accumulated values, and per-turn values for generation.

[29]A potential direction for future work may have more solid seeker models and explore which aspect of the model makes the dialogue with the expert model more goal-oriented or human-like.

Players		Automatic			Human		
Expert	**Seeker**	**Goal**	**Sco**	**T2G**	**F**	**C**	**E**
Supervised*	BERT-R	30.9	38.3	1.4	-	-	-
Bot-play \w S2S	BERT-R	42.1	49.6	2.8	-	-	-
Bot-play \w BERT-R	BERT-R	**48.6**	**52.4**	**3.2**	-	-	-
Supervised*	Human	55.0	51.2	2.1	3.1	2.2	**2.0**
Bot-play*	Human	**68.5**	**54.7**	**3.1**	**3.2**	**2.6**	**2.0**
Human	Human	95.0	64.3	8.5	4.8	4.7	4.2

Table 3.22: Evaluation on dialogue recommendation games: bot-bot (top three rows) and {bot,human}-human (bottom three rows). We use automatic game measures (**Goal**, **Score**, **Turn2Goal**) and human quality ratings (**Fluency**, **Consistency**, **Engagingness**).

goal increases from 1.4 to 3.2, indicating that conducting longer dialogues seems to be a better strategy to achieve the game goal throughout our role-playing game.

In dialogue games with human seeker players, the bot-play model also outperforms the supervised one, even though it is still far behind human performance. When the expert bot plays with the human seeker, performance increases compared to playing with the bot seeker, because the human seeker produces utterances more relevant to their movie preferences, increasing overall game success.

3.7.5 Conclusion and Future Directions

In conclusion, we have posed recommendation as a goal-oriented game between an expert and a seeker, and provided a framework for both training agents in a supervised way by learning to mimic a large set of collected human-human dialogues, as well as by bot-play between trained agents. We have shown that a combination of the two stages leads to learning better expert recommenders.

Our results suggest several promising directions. First, we noted that the recommendation performance linearly increases as more dialogue context is provided. An interesting question is how to learn to produce the best questions that will result in the most informative dialogue context.

Second, as the model becomes better at the game, we observe an increase in the length of dialogue. However, it remains shorter than the average length of human dialogues, possibly because our reward function is designed to minimize it, which worked better in experiments. A potential direction for future work is to study how different game objectives interact with each other.

Finally, our evaluation on movie recommendation is made only within the candidate set of movies given to expert. Future work should evaluate if our training scheme generalizes to a fully open-ended recommendation system, thus making our task not only useful for research and model development, but a useful end-product in itself.

3.8 Multi-aspect Planning: A Dataset for Aspect-specific Review Generation

3.8.1 Introduction

Prestigious scientific venues use peer reviewing to decide which papers to include in their journals or proceedings. While this process seems essential to scientific publication, it is often a subject of debate. Recognizing the important consequences of peer reviewing, several researchers studied various aspects of the process, including consistency, bias, author response and general review quality (e.g., Greaves et al., 2006; Ragone et al., 2011; De Silva and Vance, 2017). For example, the organizers of the NIPS 2014 conference assigned 10% of conference submissions to two different sets of reviewers to measure the consistency of the peer reviewing process, and observed that the two committees disagreed on the accept/reject decision for more than a quarter of the papers (Langford and Guzdial, 2015).

Despite these efforts, quantitative studies of peer reviews had been limited, for the most part, to the few individuals who had access to peer reviews of a given venue (e.g., journal editors and program chairs). The goal of this paper is to lower the barrier to studying peer reviews for the scientific community by introducing the first public dataset of peer reviews for research purposes: PeerRead.

We use three strategies to construct the dataset: (i) We collaborate with conference chairs and conference management systems to allow authors and reviewers to opt-in their paper drafts and peer reviews, respectively. (ii) We crawl publicly available peer reviews and annotate textual reviews with numerical scores for aspects such as 'clarity' and 'impact'. (iii) We crawl arXiv submissions which coincide with important conference submission dates and check whether a similar paper appears in proceedings of these conferences at a later date. In total, the dataset consists of 14.7K paper drafts and the corresponding accept/reject decisions, including a subset of 3K papers for which we have 10.7K textual reviews written by experts. We plan to make periodic releases of PeerRead, adding more sections for new venues every year. We provide more details on data collection in §3.8.2.

The PeerRead dataset can be used in a variety of ways. A quantitative analysis of the peer reviews can provide insights to help better understand (and potentially improve) various nuances of the review process. For example, in §3.8.3, we analyze correlations between the overall recommendation score and individual aspect scores (e.g., clarity, impact and originality) and quantify how reviews recommending an oral presentation differ from those recommending a poster. Other examples might include aligning review scores with authors to reveal gender or nationality biases. From a pedagogical perspective, the PeerRead dataset also provides inexperienced authors and first-time reviewers with diverse examples of peer reviews.

As an NLP resource, peer reviews raise interesting challenges, both from the realm of sentiment analysis—predicting various properties of the reviewed paper, e.g., clarity and novelty, as well as that of text generation—given a paper, automatically generate its review. Such NLP tasks, when solved with sufficiently high quality, might help reviewers, area chairs and program chairs in the reviewing process, e.g., by lowering the number of reviewers needed for some paper submission.

Section	#Papers	#Reviews	Asp.	Acc / Rej
NIPS 2013–2017	2,420	9,152	×	2,420 / 0
ICLR 2017	427	1,304	✓	172 / 255
ACL 2017	137	275	✓	88 / 49
CoNLL 2016	22	39	✓	11 / 11
arXiv 2007–2017	11,778	—	—	2,891 / 8,887
total	14,784	10,770		

Table 3.23: The PeerRead dataset. **Asp.** indicates whether the reviews have aspect specific scores (e.g., clarity). Note that ICLR contains the aspect scores assigned by our annotators (see Section 3.8.2). **Acc/Rej** is the distribution of accepted/rejected papers. Note that NIPS provide reviews only for accepted papers.

In §3.8.4, we introduce two new NLP tasks based on this dataset: (i) predicting whether a given paper would be accepted to some venue, and (ii) predicting the numerical score of certain aspects of a paper. Our results show that we can predict the accept/reject decisions with 6–21% error reduction compared to the majority reject-all baseline, in four different sections of PeerRead. Since the baseline models we use are fairly simple, there is plenty of room to develop stronger models to make better predictions.

3.8.2 Peer-Review Dataset (PeerRead)

Here we describe the collection and compilation of PeerRead, our scientific peer-review dataset. For an overview of the dataset, see Table 3.23.

Review Collection

Reviews in PeerRead belong to one of the two categories:

Opted-in reviews. We coordinated with the Softconf conference management system and the conference chairs for CoNLL 2016[30] and ACL 2017[31] conferences to allow authors and reviewers to opt-in their drafts and reviews, respectively, to be included in this dataset. A submission is included only if (i) the corresponding author opts-in the paper draft, and (ii) at least one of the reviewers opts-in their anonymous reviews. This resulted in 39 reviews for 22 CoNLL 2016 submissions, and 275 reviews for 137 ACL 2017 submissions. Reviews include both text and aspect scores (e.g., calrity) on a scale of 1–5.

Peer reviews on the web. In 2013, the NIPS conference[32] began attaching all accepted papers with their anonymous textual review comments, as well as a confidence level on a scale of 1–3.

[30]The 20th SIGNLL Conference on Computational Natural Language Learning; `http://www.conll.org/2016`
[31]The 55th Annual Meeting of the Association for Computational Linguistics; `http://acl2017.org/`
[32]The Conference on Neural Information Processing Systems; `https://nips.cc/`

We collected all accepted papers and their reviews for NIPS 2013–2017, a total of 9,152 reviews for 2,420 papers.

Another source of reviews is the OpenReview platform:[33] a conference management system which promotes open access and open peer reviewing. Reviews include text, as well as numerical recommendations between 1–10 and confidence level between 1–5. We collected all submissions to the ICLR 2017 conference,[34] a total of 1,304 official, anonymous reviews for 427 papers (177 accepted and 255 rejected).[35]

arXiv Submissions

arXiv[36] is a popular platform for pre-publishing research in various scientific fields including physics, computer science and biology. While arXiv does not contain reviews, we automatically label a subset of arXiv submissions in the years 2007–2017 (inclusive)[37] as accepted or probably-rejected, with respect to a group of top-tier NLP, ML and AI venues: ACL, EMNLP, NAACL, EACL, TACL, NIPS, ICML, ICLR and AAAI.

Accepted papers. In order to assign 'accepted' labels, we use the dataset provided by Sutton and Gong (2017) who matched arXiv submissions to their bibliographic entries in the DBLP directory[38] by comparing titles and author names using Jaccard's distance. To improve our coverage, we also add an arXiv submission if its title matches an accepted paper in one of our target venues with a relative Levenshtein distance (Levenshtein, 1966) of < 0.1. This results in a total of 2,891 accepted papers.

Probably-rejected papers. We use the following criteria to assign a 'probably-rejected' label for an arXiv submission:

- The paper wasn't accepted to any of the target venues.[39]
- The paper was submitted to one of the arXiv categories `cs.cl`, `cs.lg` or `cs.ai`.[40]
- The paper wasn't cross-listed in any non-`cs` categories.
- The submission date[41] was within one month of the submission deadlines of our target venues (before or after).
- The submission date coincides with at least one of the arXiv papers accepted for one of the target venues.

This process results in 8,887 'probably-rejected' papers.

[33] `http://openreview.net`

[34] The 5th International Conference on Learning Representations; `https://iclr.cc/archive/www/2017.html`

[35] The platform also allows any person to review the paper by adding a comment, but we only use the official reviews of reviewers assigned to review that paper.

[36] `https://arxiv.org/`

[37] For consistency, we only include the first arXiv version of each paper (accepted or rejected) in the dataset.

[38] `http://dblp.uni-trier.de/`

[39] Note that some of the 'probably-rejected' papers may be published at workshops or other venues.

[40] See `https://arxiv.org/archive/cs` for a description of the computer science categories in arXiv.

[41] If a paper has multiple versions, we consider the submission date of the first version.

Data quality. We did a simple sanity check in order to estimate the number of papers that we labeled as 'probably-rejected', but were in fact accepted to one of the target venues. Some authors add comments to their arXiv submissions to indicate the publication venue. We identified arXiv papers with a comment which matches the term "accept" along with any of our target venues (e.g., "nips"), but not the term "workshop". We found 364 papers which matched these criteria, 352 out of which were labeled as 'accepted'. Manual inspection of the remaining 12 papers showed that one of the papers was indeed a false negative (i.e., labeled as 'probably-rejected' but accepted to one of the target venues) due to a significant change in the paper title. The remaining 11 papers were not accepted to any of the target venues (e.g., "accepted at WMT@ACL 2014").

Organization and Preprocessing

We organize v1.0 of the PeerRead dataset in five sections: CoNLL 2016, ACL 2017, ICLR 2017, NIPS 2013–2017 and arXiv 2007–2017.[42] Since the data collection varies across sections, different sections may have different license agreements. The papers in each section are further split into standard training, development and test sets with 0.9:0.05:0.05 ratios. In addition to the PDF file of each paper, we also extract its textual content using the `Science Parse` library.[43] We represent each of the splits as a json-encoded text file with a list of paper objects, each of which consists of paper details, accept/reject/probably-reject decision, and a list of reviews.

Aspect Score Annotations

In many publication venues, reviewers assign numeric aspect scores (e.g., clarity, originality, substance) as part of the peer review. Aspect scores could be viewed as a structured summary of the strengths and weaknesses of a paper. While aspect scores assigned by reviewers are included in the opted-in sections in PeerRead, they are missing from the remaining reviews. In order to increase the utility of the dataset, we annotated 1.3K reviews with aspect scores, based on the corresponding review text. Annotations were done by two of the authors. In this subsection, we describe the annotation process in detail.

Feasibility study. As a first step, we verified the feasibility of the annotation task by annotating nine reviews for which aspect scores are available. The annotators were able to infer about half of the aspect scores from the corresponding review text (the other half was not discussed in the review text). This is expected since reviewer comments often focus on the key strengths or weaknesses of the paper and are not meant to be a comprehensive assessment of each aspect. On average, the absolute difference between our annotated scores and the gold scores originally provided by reviewers is 0.51 (on a 1–5 scale, considering only those cases where the aspect was discussed in the review text).

Data preprocessing. We used the official reviews in the ICLR 2017 section of the dataset for this annotation task. We excluded unofficial comments contributed by arbitrary members

[42] We plan to periodicly release new versions of PeerRead.
[43] `https://github.com/allenai/science-parse`

of the community, comments made by the authors in response to other comments, as well as "meta-reviews" which state the final decision on a paper submission. The remaining 1,304 official reviews are all written by anonymous reviewers assigned by the program committee to review a particular submission. We randomly reordered the reviews before annotation so that the annotator judgments based on one review are less affected by other reviews of the same paper.

Annotation guidelines. We annotated seven aspects for each review: appropriateness, clarity, originality, soundness/correctness, meaningful comparison, substance, and impact. For each aspect, we provided our annotators with the instructions given to ACL 2016 reviewers for this aspect. Our annotators' task was to read the detailed review text (346 words on average) and select a score between 1–5 (inclusive, integers only) for each aspect.[44] When review comments do not address a specific aspect, we do not select any score for that aspect, and instead use a special "not discussed" value.

Data quality. In order to assess annotation consistency, the same annotators re-annotated a random sample consisting of 30 reviews. On average, 77% of the annotations were consistent (i.e., the re-annotation was exactly the same as the original annotation, or was off by 1 point) and 2% were inconsistent (i.e., the re-annotation was off by 2 points or more). In the remaining 21%, the aspect was marked as "not discussed" in one annotation but not in the other. We note that different aspects are discussed in the textual reviews at different rates. For example, about 49% of the reviews discussed the 'originality' aspect, while only 5% discussed 'appropriateness'.

3.8.3 Data-Driven Analysis of Peer Reviews

In this section, we showcase the potential of using PeerRead for data-driven analysis of peer reviews.

Overall recommendation vs. aspect scores. A critical part of each review is the overall recommendation score, a numeric value which best characterizes a reviewer's judgment of whether the draft should be accepted for publication in this venue. While aspect scores (e.g., clarity, novelty, impact) help explain a reviewer's assessment of the submission, it is not necessarily clear which aspects reviewers appreciate the most about a submission when considering their overall recommendation.

To address this question, we measure pair-wise correlations between the overall recommendation and various aspect scores in the ACL 2017 section of PeerRead and report the results in Table 3.24.

The aspects which correlate most strongly with the final recommendation are substance (which concerns the amount of work rather than its quality) and clarity. In contrast, soundness/-correctness and originality are least correlated with the final recommendation. These observations raise interesting questions about what we collectively care about the most as a research community when evaluating paper submissions.

[44]Importantly, our annotators only considered the review text, and did not have access to the papers.

Aspect	ρ
Substance	0.59
Clarity	0.42
Appropriateness	0.30
Impact	0.16
Meaningful comparison	0.15
Originality	0.08
Soundness/Correctness	0.01

Table 3.24: Pearson's correlation coefficient ρ between the overall recommendation and various aspect scores in the ACL 2017 section of PeerRead.

Oral vs. poster. In most NLP conferences, accepted submissions may be selected for an oral presentation or a poster presentation. The presentation format decision of accepted papers is based on recommendation by the reviewers. In the official blog of ACL 2017,[45] the program chairs recommend that reviewers and area chairs make this decision based on the expected size of interested audience and whether the ideas can be grasped without back-and-forth discussion. However, it remains unclear what criteria are used by reviewers to make this decision.

To address this question, we compute the mean aspect score in reviews which recommend an oral vs. poster presentation in the ACL 2017 section of PeerRead, and report the results in Table 3.25. Notably, the average 'overall recommendation' score in reviews recommending an oral presentation is 0.9 higher than in reviews recommending a poster presentation, suggesting that reviewers tend to recommend oral presentation for submissions which are holistically stronger.

Presentation format	Oral	Poster	Δ	stdev
Recommendation	3.83	2.92	0.90	0.89
Substance	3.91	3.29	0.62	0.84
Clarity	4.19	3.72	0.47	0.90
Meaningful comparison	3.60	3.36	0.24	0.82
Impact	3.27	3.09	0.18	0.54
Originality	3.91	3.88	0.02	0.87
Soundness/Correctness	3.93	4.18	-0.25	0.91

Table 3.25: Mean review scores for each presentation format (oral vs. poster). Raw scores range between 1–5. For reference, the last column shows the sample standard deviation based on all reviews.

ACL 2017 vs. ICLR 2017. Table 3.26 reports the sample mean and standard deviation of various measurements based on reviews in the ACL 2017 and the ICLR 2017 sections of PeerRead.

[45] https://acl2017.wordpress.com/2017/03/23/conversing-or-presenting-poster-or-oral/

Most of the mean scores are similar in both sections, with a few notable exceptions. The comments in ACL 2017 reviews tend to be about 50% longer than those in the ICLR 2017 reviews. Since review length is often thought of as a measure of its quality, this raises interesting questions about the quality of reviews in ICLR vs. ACL conferences. We note, however, that ACL 2017 reviews were explicitly opted-in while the ICLR 2017 reviews include all official reviews, which is likely to result in a positive bias in review quality of the ACL reviews included in this study.

Another interesting observation is that the mean appropriateness score is lower in ICLR 2017 compared to ACL 2017. While this might indicate that ICLR 2017 attracted more irrelevant submissions, this is probably an artifact of our annotation process: reviewers probably only address appropriateness explicitly in their review if the paper is inappropriate, which leads to a strong negative bias against this category in our ICLR dataset.

Measurement	ACL'17	ICLR'17
Review length (words)	531±323	346±213
Appropriateness	4.9±0.4	2.6±1.3
Meaningful comparison	3.5±0.8	2.9±1.1
Substance	3.6±0.8	3.0±0.9
Originality	3.9±0.9	3.3±1.1
Clarity	3.9±0.9	4.2±1.0
Impact	3.2±0.5	3.4±1.0
Overall recommendation	3.3±0.9	3.3±1.4

Table 3.26: Mean ± standard deviation of various measurements on reviews in the ACL 2017 and ICLR 2017 sections of PeerRead. Note that ACL aspects were written by the reviewers themselves, while ICLR aspects were predicted by our annotators based on the review.

3.8.4 NLP Tasks

Aside from quantitatively analyzing peer reviews, PeerRead can also be used to define interesting NLP tasks. In this section, we introduce two novel tasks based on the PeerRead dataset. In the first task, given a paper draft, we predict whether the paper will be accepted to a set of target conferences. In the second task, given a textual review, we predict the aspect scores for the paper such as novelty, substance and meaningful comparison.[46]

Both these tasks are not only challenging from an NLP perspective, but also have potential applications. For example, models for predicting the accept/reject decisions of a paper draft might be used in recommendation systems for arXiv submissions. Also, a model trained to predict the aspect scores given review comments using thousands of training examples might result in better-calibrated scores.

[46]We also experiment with conditioning on the paper itself to make this prediction.

	ICLR	cs.cl	cs.lg	cs.ai
Majority	57.6	68.9	67.9	92.1
Ours	65.3	75.7	70.7	92.6
(Δ)	+7.7	+6.8	+2.8	+0.5

Table 3.27: Test accuracies (%) for acceptance classification. Our best model outperforms the majority classifiers in all cases.

Paper Acceptance Classification

Paper acceptance classification is a binary classification task: given a paper draft, predict whether the paper will be accepted or rejected for a predefined set of venues.

Models. We train a binary classifier to estimate the probability of accept vs. reject given a paper, i.e., $P(\texttt{accept=True} \mid \texttt{paper})$. We experiment with different types of classifiers: logistic regression, SVM with linear or RBF kernels, Random Forest, Nearest Neighbors, Decision Tree, Multi-layer Perceptron, AdaBoost, and Naive Bayes. We use hand-engineered features, instead of neural models, because they are easier to interpret.

We use 22 coarse features, e.g., length of the title and whether jargon terms such as 'deep' and 'neural' appear in the abstract, as well as sparse and dense lexical features.

Experimental setup. We experiment with the ICLR 2017 and the arXiv sections of the PeerRead dataset. We train separate models for each of the arXiv category: `cs.cl`, `cs.lg`, and `cs.ai`. We use python's sklearn's implementation of all models (Pedregosa et al., 2011a).[47] We consider various regularization parameters for SVM and logistic regression. We use the standard test split and tune our hyperparameters using 5-fold cross validation on the training set.

Results. Table 3.27 shows our test accuracies for the paper acceptance task. Our best model outperforms the majority classifier in all cases, with up to 22% error reduction. Since our models lack the sophistication to assess the quality of the work discussed in the given paper, this might indicate that some of the features we define are correlated with strong papers, or bias reviewers' judgments.

We run an ablation study for this task for the ICLR and arXiv sections. We train only one model for all three categories in arXiv to simplify our analysis. Table 3.28 shows the absolute degradation in test accuracy of the best performing model when we remove one of the features. The table shows that some features have a large contribution on the classification decision: adding an appendix, a large number of theorems or equations, the average length of the text preceding a citation, the number of papers cited by this paper that were published in the five years before the submission of this paper, whether the abstract contains a phrase "state of the art" for ICLR or "neural" for arXiv, and length of title.

[47] `http://scikit-learn.org/stable/`

ICLR	%
Best model	65.3
– appendix	–5.4
– num_theorems	–3.8
– num_equations	–3.8
– avg_len_ref	–3.8
– $\text{abstract}_{\text{state-of-the-art}}$	–3.5
– #recent_refs	–2.5

arXiv	%
Best model	79.1
– avg_len_ref	–1.4
– num_uniq_words	–1.1
– num_theorems	–1.0
– $\text{abstract}_{\text{neural}}$	–1.0
– num_refmentions	–1.0
– title_length	–1.0

Table 3.28: The absolute % difference in accuracy on the paper acceptance prediction task when we remove only one feature from the full model. Features with larger negative differences are more salient, and we only show the six most salient features for each section. The features are num_X: number of X (e.g., theorems or equations), avg_len_ref: average length of context before a reference, appendix: does paper have an appendix, abstract_X: does the abstract contain the phrase X, num_uniq_words: number of unique words, num_refmentions: number of reference mentions, and #recent_refs: number of cited papers published in the last five years.

Review Aspect Score Prediction

The second task is a multi-class regression task to predict scores for seven review aspects: 'impact', 'substance', 'appropriateness', 'comparison', 'soundness', 'originality' and 'clarity'. For this task, we use the two sections of PeerRead which include aspect scores: ACL 2017 and ICLR 2017.[48]

Models. We use a regression model which predicts a floating-point score for each aspect of interest given a sequence of tokens. We train three variants of the model to condition on (i) the paper text only, (ii) the review text only, or (iii) both paper and review text.

We use three neural architectures: convolutional neural networks (CNN, Zhang et al., 2015), recurrent neural networks (LSTM, Hochreiter and Schmidhuber, 1997b), and deep averaging networks (DAN, Iyyer et al., 2015). In all three architectures, we use a linear output layer to make the final prediction. The loss function is the mean squared error between predicted and gold scores. We compare against a baseline which always predicts the mean score of an aspect, computed on the training set.[49]

Experimental setup. We train all models on the standard training set for 100 iterations, and select the best performing model on the standard development set. We use a single 100 dimension layer LSTM and CNN, and a single output layer of 100 dimensions for all models. We use GloVe 840B embeddings (Pennington et al., 2014) as input word representations, without tuning, and keep the 35K most frequent words and replace the rest with an UNK vector. The CNN model uses 128 filters and 5 kernels. We use an RMSProp optimizer (Tieleman and Hinton, 2012) with

[48]The CoNLL 2016 section also includes aspect scores but is too small for training.

[49]This baseline is guaranteed to obtain mean square errors less than or equal to the majority baseline.

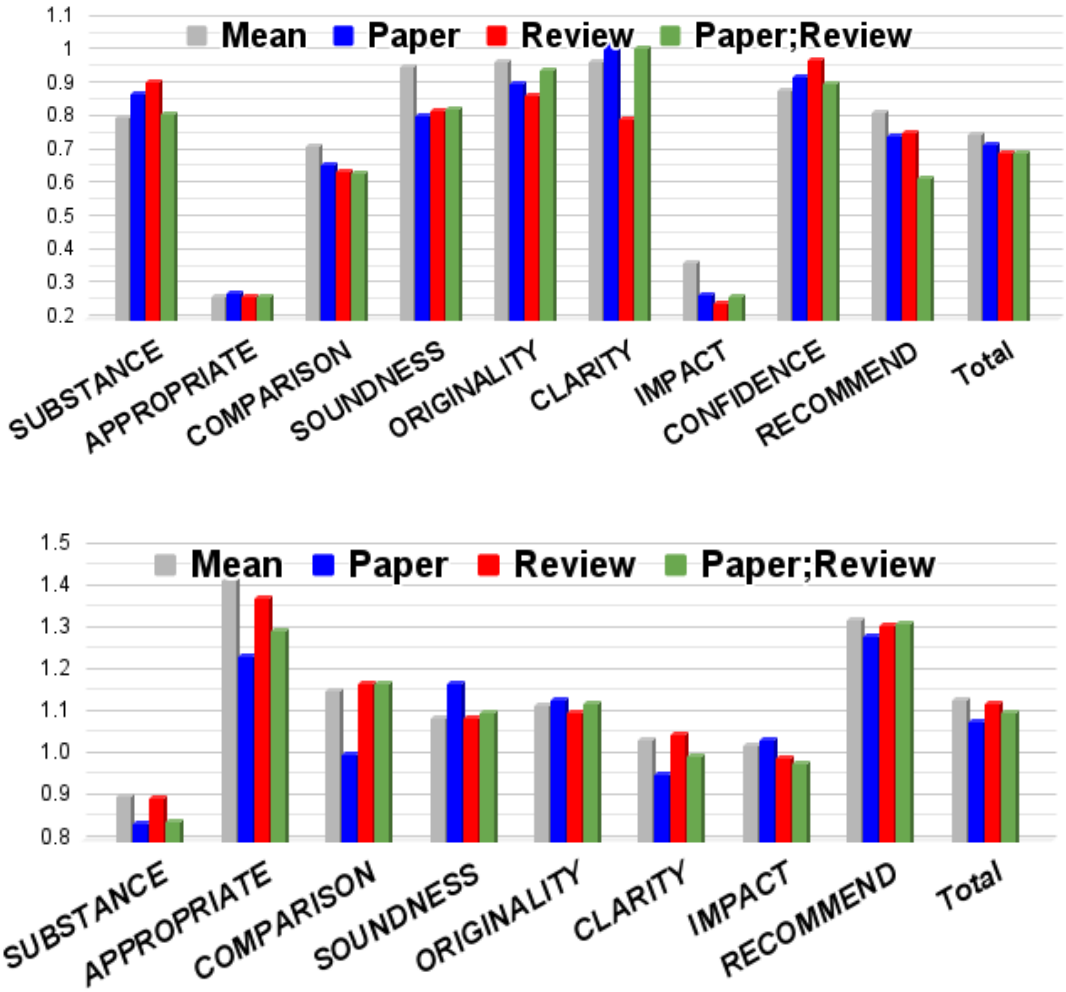


Figure 3.19: Root mean squared error (RMSE, lower is better) on the test set for the aspect prediction task on the ACL 2017 (top) and the ICLR 2017 (bottom) sections of PeerRead.

0.001 learning rate, 0.9 decay rate, 5.0 gradient clipping, and a batch size of 32. Since scientific papers tend to be long, we only take the first 1000 and 200 tokens of each paper and review, respectively, and concatenate the two prefixes when the model conditions on both the paper and review text.[50]

Results. Figure 3.19 shows the test set root mean square error (RMSE) on the aspect prediction task (lower is better). For each section (ACL 2017 and ICLR 2017), and for each aspect, we report the results of four systems: 'Mean' (baseline), 'Paper', 'Review' and 'Paper;Review' (i.e., which information the model conditions on). For each variant, the model which performs best on the development set is selected.

[50]We note that the goal of this paper is to demonstrate potential uses of PeerRead, rather than develop the best model to address this task, which explains the simplicity of the models we use.

We note that aspects with higher RMSE scores for the 'Mean' baseline indicate higher variance among the review scores for this aspect, so we focus our discussion on these aspects. In the ACL 2017 section, the two aspects with the highest variance are 'originality' and 'clarity'. In the ICLR 2017 section, the two aspects with the highest variance are 'appropriateness' and 'meaningful comparison'. Surprisingly, the 'Paper;Review' model outperforms the 'Mean' baseline in all four aspects, and the 'Review' model outperforms the 'Mean' baseline in three out of four. On average, all models slightly improve over the 'Mean' baseline.

3.8.5 Related Work

We first survey previous works that attempt to collect and analze the review dataset.

Several efforts have recently been made to collect peer reviews. Publons[51] consolidates peer reviews data to build public reviewer profiles for participating reviewers. Crossref maintains the database of DOIs for its 4000+ publisher members. They recently launched a service to add peer reviews as part of metadata for the scientific articles.[52] Surprisingly, however, most of the reviews are not made publicly available. In contrast, we collected and organized PeerRead such that it is easy for other researchers to use it for research purposes, replicate experiments and make a fair comparison to previous results.

There have been several efforts to analyze the peer review process (e.g., Bonaccorsi et al., 2018; Rennie, 2016). Editors of the British Journal of Psychiatry found differences in courtesy between signed and unsigned reviews (Walsh et al., 2000). (Ragone et al., 2011) and (Birukou et al., 2011) analyzed ten CS conferences and found low correlation between review scores and the impact of papers in terms of future number of citations. (Fang et al., 2016) presented similar observations for NIH grant application reviews and their productivity. Langford and Guzdial (2015) pointed to inconsistencies in the peer review process.

Several recent venues had single vs. double blind review experiments, which pointed to single-blind reviews leading to increased biases towards male authors (Roberts and Verhoef, 2016) and famous institutions (Tomkins et al., 2017). Further, Le Goues et al. (2017) showed that reviewers are unable to successfully guess the identity of the author in a double-blind review. Recently, there have been several initiatives by program chairs in major NLP conferences to study various aspects of the review process, mostly author response and general review quality.[53] In this work, we provide a large scale dataset that would enable the wider scientific community to further study the properties of peer review, and potentially come up with enhancements to current peer review model.

Finally, the peer review process is meant to judge the quality of research work being disseminated to the larger research community. With the ever-growing rates of articles being submitted to top-tier conferences in Computer Science and pre-print repositories (Sutton and Gong, 2017), there is a need to expedite the peer review process. Balachandran (2013) proposed a method for automatic analysis of conference submissions to recommend relevant reviewers. Also related to our acceptance predicting task are (Tsur and Rappoport, 2009) and (Ashok et al., 2013), both

[51] publons.com/dashboard/records/review/

[52] https://www.crossref.org/blog/peer-reviews-are-open-for-registering-at-crossref/

[53] See https://nlpers.blogspot.com/2015/06/some-naacl-2013-statistics-on-author.html and https://acl2017.wordpress.com/2017/03/27/author-response-does-it-help/

of which focuses on predicting book reviews. Various automatic tools like Grammerly[54] can assist reviewers in discovering grammar and spelling errors. Tools like Citeomatic[55] (Bhagavatula et al., 2018) are especially useful in finding relevant articles not cited in the manuscript. We believe that the NLP tasks presented in this paper, predicting the acceptance of a paper and the aspect scores of a review, can potentially serve as useful tools for writing a paper, reviewing it, and deciding about its acceptance.

3.8.6 Conclusion

We introduced PeerRead, the first publicly available peer review dataset for research purposes, containing 14.7K papers and 10.7K reviews. We analyzed the dataset, showing interesting trends such as a high correlation between overall recommendation and recommending an oral presentation. We defined two novel tasks based on PeerRead: (i) predicting the acceptance of a paper based on textual features and (ii) predicting the score of each aspect in a review based on the paper and review contents. Our experiments show that certain properties of a paper, such as having an appendix, are correlated with higher acceptance rate. Our primary goal is to motivate other researchers to explore these tasks and develop better models that outperform the ones used in this work. More importantly, we hope that other researchers will identify novel opportunities which we have not explored to analyze the peer reviews in this dataset. As a concrete example, it would be interesting to study if the accept/reject decisions reflect author demographic biases (e.g., nationality).

3.9 Hierarchical Planning: Sub-aspect Bias Analysis on Summarization

3.9.1 Introduction

Despite numerous recent developments in neural summarization systems (Narayan et al., 2018b; Ha et al., 2015; Nallapati et al., 2016; See et al., 2017; Kedzie et al., 2018; Gehrmann et al., 2018; Paulus et al., 2017) the underlying rationales behind the improvements and their dependence on the training corpus remain largely unexplored. Edmundson (1969) put forth the position hypothesis: important sentences appear in preferred positions in the document. Lin and Hovy (1997) provide a method to empirically identify such positions. Later, Hong and Nenkova (2014) showed an intentional lead bias in news writing, suggesting that sentences appearing early in news articles are more important for summarization tasks. More generally, it is well known that recent state-of-the-art models (Nallapati et al., 2016; See et al., 2017) are often marginally better than the first-k baseline on single-document news summarization.

In order to address the position bias of news articles, Narayan et al. (2018a) collected a new dataset called XSum to create single sentence summaries that include material from multiple

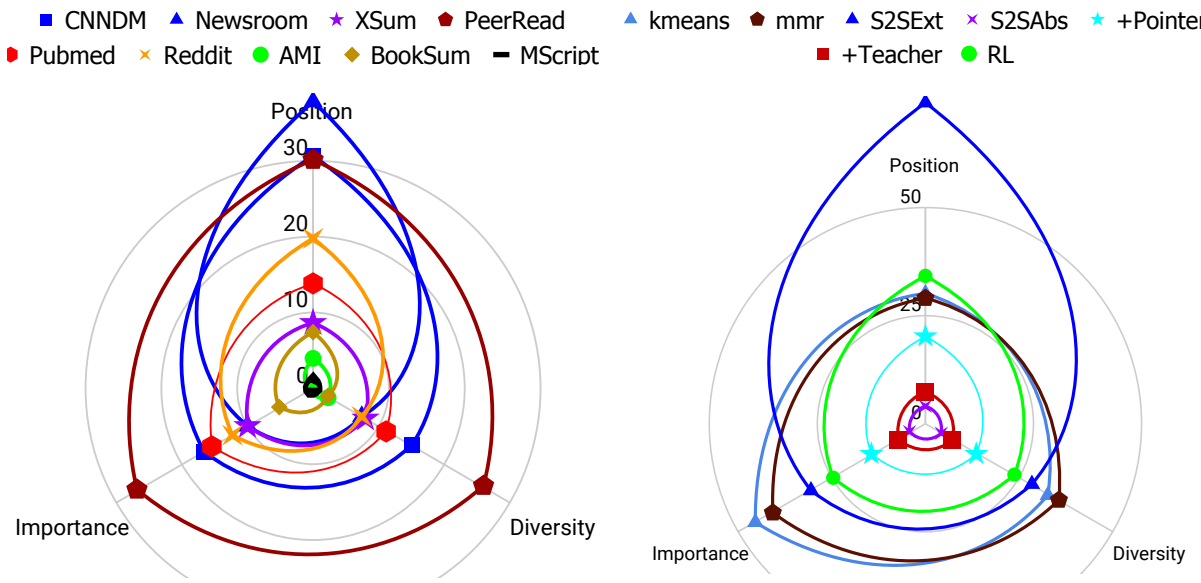
[54] https://www.grammarly.com/
[55] http://allenai.org/semantic-scholar/citeomatic/

positions in the source document. Kedzie et al. (2018) showed that the position bias in news articles is not the same across other domains such as meeting minutes (Carletta et al., 2005).

In addition to `position`, Lin and Bilmes (2012) defined other sub-aspect functions of summarization including `coverage`, `diversity`, and `information`. Lin and Bilmes (2011) claim that many existing summarization systems are instances of mixtures of such sub-aspect functions; for example, maximum marginal relevance (MMR) (Carbonell and Goldstein, 1998) can be seen as an combination of diversity and importance functions.

Following the sub-aspect theory, we explore three important aspects of summarization (§3.9.2): POSITION for choosing sentences by their position, IMPORTANCE for choosing relevant contents, and DIVERSITY for ensuring minimal redundancy between summary sentences.

We then conduct an in-depth analysis of these aspects over nine different domains of summarization corpora (§3.9.4) including news articles, meeting minutes, books, movie scripts, academic papers, and personal posts. For each corpus, we investigate which aspects are most important and develop a notion of **corpus bias** (§3.9.5). We provide an empirical result showing how current summarization systems are compounded of which sub-aspect factors called **system bias** (§3.9.6). At last, we summarize our actionable messages for future summarization researches. We summarize some notable findings as follows:



(a) Corpus bias

(b) System bias

Figure 3.20: Corpus and system biases with the three sub-aspects, showing what portion of aspect is used for each corpus and each system. The portion is measured by calculating ROUGE score between (a) summaries obtained from each aspect and target summaries or (b) summaries obtained from each aspect and each system.

- Summarization of personal post and news articles except for XSum (Narayan et al., 2018a) are biased to the position aspect, while academic papers are well balanced among the three aspects (see Figure 3.20 (a)). Summarizing long documents (e.g. books and movie scripts) and

conversations (e.g. meeting minutes) are extremely difficult tasks that require multiples aspects together.

- Biases do exist in current summarization systems (Figure 3.20 (b)). Simple ensembling of multiple aspects of systems show comparable performance with simple single-aspect systems.
- Reference summaries in current corpora include less than 15% of new words that do not appear in the source document, except for abstract text of academic papers.
- Semantic volume (Yogatama et al., 2015) overlap between the reference and model summaries is not correlated with the hard evaluation metrics such as ROUGE (Lin, 2004).

3.9.2 Sub-aspects of Summarization

We focus on three crucial aspects : Position, Diversity, and Importance. For each aspect, we use different extractive algorithms to **capture how much of the aspect is used in the oracle extractive summaries**[56]. For each algorithm, the goal is to select k extractive summary sentences (equal to the number of sentences in the target summaries for each sample) out of N sentences appearing in the original source. The chosen sentences or their indices will be used to calculate the various evaluation metrics described in §3.9.3

For some algorithms below, we use vector representation of sentences. We parse a document x into a sequence of sentences $x = x_1..x_N$ where each sentence consists of a sequence of words $x_i = w_1..w_s$. Each sentence is then encoded:

$$\mathrm{E}(x_i) = \texttt{BERT}(w_{i,1}..w_{i,s}) \tag{3.30}$$

where BERT (Devlin et al., 2019) is a pre-trained bidirectional encoder from transformers (Vaswani et al., 2017)[57]. We use the last layer from BERT as a representation of each token, and then average them to get final representation of a sentence. All tokens are lower cased.

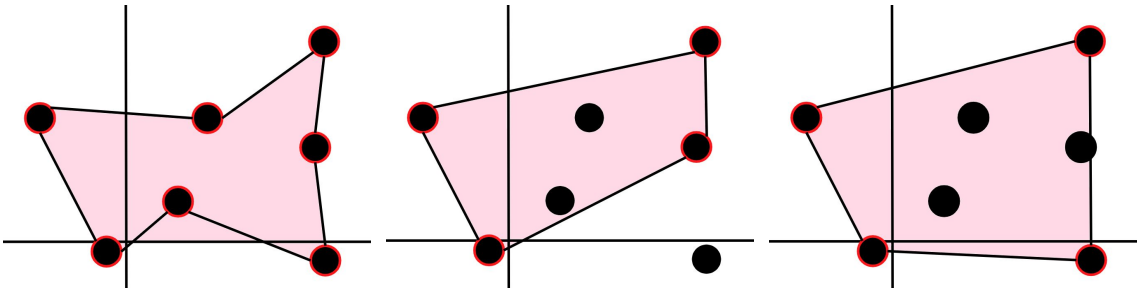
Position Position of sentences in the source has been suggested as a good indicator for choosing summary sentences, especially in news articles (Lin and Hovy, 1997; Hong and Nenkova, 2014; See et al., 2017). We compare three position-based algorithms: **First**, **Last**, and **Middle**, by simply choosing k number of sentences in the source document from these positions.

Diversity Yogatama et al. (2015) assume that extractive summary sentences which maximize the semantic volume in a distributed semantic space are the most diverse but least redundant sentences. Motivated by this notion, our goal is to find a set of k sentences that maximizes the volume size of them in a continuous embedding space like the BERT representations in Eq 3.30. Our objective is to find the optimal search function $\mathcal{S}$ that maximizes the volume size $\mathcal{V}$ of searched sentences: $\arg\max_{1..k} \mathcal{V}(\mathcal{S}_{1..c}(\mathrm{E}(x_1), \ldots, \mathrm{E}(x_N)))$.

If k=N, we use every sentence from the source document. (Figure 3.21 (a)). However, its volume space does not guarantee to maximize the volume size because of the non-convex polygonality. In order to find a convex maximum volume, we consider two different algorithms described below.

[56] See §3.9.3 for our oracle set construction.

[57] The other encoders such as averaging word embeddings (Pennington et al., 2014) show comparable performance.



(a) Default **(b)** Heuristic **(c)** ConvexFall

Figure 3.21: Volume maximization functions. Black dots are sentences in source document, and red dots are chosen summary sentences. The red-shaded polygons are volume space of the summary sentences.

Heuristic. Yogatama et al. (2015) heuristically choose a set of summary sentences using a greedy algorithm: It first chooses a sentence which has the farthest vector representation from the centroid of whole source sentences, and then repeatedly finds sentences whose representation is farthest from the centroid of vector representations of the chosen sentences. Unlike the original algorithm in (Yogatama et al., 2015) restricting the number of words, we constrain the total number of selected sentences to k. This heuristic algorithm can fail to find the maximum volume depending on its starting point and/or the farther distance between two points detected (Figure 3.21 (b)).

ConvexFall. Here we first find the convexhull[58] using Quickhull (Barber et al., 1996), implemented by Qhull library[59]. It guarantees the maximum volume size of selected points with minimum number of points (Figure 3.21 (c)). However, it does not reduce a redundancy between the points over the convex-hull, and usually choose larger number of sentences than k. Marcu (1999) shows an interesting study regarding an importance of sentences: given a document, if one deletes the least central sentence from the source text, then at some point the similarity with the reference text rapidly drops at sudden called the *waterfall* phenomena. Motivated by his study, we similarly prune redundant sentences from the set chosen by convex-hull search. For each turn, the sentence with the lowest volume reduction ratio is pruned until the number of remaining sentences is equivalent to k.

IMPORTANCE We assume that contents that repeatedly occur in one document contain *important* information. We find sentences that are nearest to the neighbour sentences using two distance measures: **N-Nearest** calculates an averaged Pearson correlation between one and the rest for all source sentence vector representations. k sentences having the highest averaged correlation are selected as final extractive summaries. On the other hand, **K-Nearest** chooses the K nearest sentences per each sentence, and then averages distances between each nearest sentence and the selected one. The one has the lowest averaged distance is chosen. This calculation is repeated k times and the selected sentences are removed from the remaining pool.

[58]Definition: a set of points is defined as the smallest convex set that includes the points.
[59]`http://www.qhull.org/`

3.9.3 Metrics

In order to determine the aspects most crucial to the summarization task, we use three evaluation metrics:

ROUGE is Recall-Oriented Understudy for Gisting Evaluation (Lin and Hovy, 2000) for evaluating summarization systems. We use ROUGE-1 (R1), ROUGE-2 (R2), and ROUGE-L (RL) F-measure scores which corresponds to uni-gram, bigrams and longest common subsequences, respectively, and their averaged score (R).

Volume Overlap (VO) ratio. Hard metrics like ROUGE often ignore semantic similarities between sentences. Based on the volume assumption in (Yogatama et al., 2015), we measure overlap ratio of two semantic volumes calculated by the model and target summaries. We obtain a set of vector representations of the reference summary sentences $\hat{Y}$ and the model summary sentences Y predicted by any algorithm *algo* in §3.9.2 for the i-th document:

$$\hat{Y}_i = (\,\hat{y}_{i,1} \,..\, \hat{y}_{i,k}\,), \quad Y_i^{algo} = (\,y_{i,1}^{algo} \,..\, y_{i,k}^{algo}\,) \tag{3.31}$$

Each volume V is then calculated using the convex-hull algorithm and their overlap ($\sqcap$) is calculated using a shapely package[60][61]. The final VO is then:

$$\text{VO}_{algo} = \sum_{i=1}^{N} \frac{\text{V}(\,E(Y_i^{algo})) \sqcap \text{V}\,(E(\hat{Y}_i))}{\text{V}\,(E(\hat{Y}_i))} \tag{3.32}$$

where N is the total number of input documents, E is the BERT sentence encoder in Eq 3.30, and $E(\hat{Y}_i)$ and $E(Y_i^{algo})$ are a set of vector representations of the reference and model summary sentences, respectively. The volume overlap indicates how two summaries are semantically overlapped in a continuous embedding space.

Sentence Overlap (SO) ratio. Even though ROUGE provides a recall-oriented lexical overlap, we don't know the upper-bound on performance (called `oracle`) of the extractive summarization. We extract the oracle extractive sentences (i.e. a set of input sentences) which maximizes ROUGE-L F-measure score with the reference summary. We then measure sentence overlap (SO) which determines how many extractive sentences from our algorithms are in the oracle summary. The SO is:

$$\text{SO}_{algo} = \sum_{i=1}^{n} \frac{\text{C}(Y_i^{algo} \cap \hat{Y}_i)}{\text{C}(\hat{Y}_i)} \tag{3.33}$$

where C is a function for counting the number of elements in a set. The sentence overlap indicates how well the algorithm finds the oracle summaries for extractive summarization.

3.9.4 Summarization Corpora

We use various domains of summarization datasets to conduct the bias analysis across corpora and systems. Each dataset has source documents and corresponding abstractive target summaries. We provide a list of datasets used along with a brief description and our pre-processing scheme:

[60] `https://pypi.org/project/Shapely/`

[61] Due to the lack of overlap calculation between two polygons of high dimensions, we reduce it to 2D PCA space.

	CNNDM	Newsroom	Xsum	PeerRead	PubMed	Reddit	AMI	BookSum	MScript
Source	News	News	News	Papers	Papers	Post	Minutes	Books	Script
Multi-sents.	✓	✓	X	✓	✓	X	✓	✓	✓
Data size	287K/11K	992K/109K	203K/11K	10K/550	21K/2.5K	404/48	98/20	- /53	- /1K
Avg src sents.	40/34	24/24	33/33	45/45	97/97	19/15	767/761	- /6.7K	- /3K
Avg tgt sents.	4/4	1.4/1.4	1/1	6/6	10/10	1/1	17/17	- /336	- /5
Avg src tokens	792/779	769 /762	440/442	1K/1K	2.4K/2.3K	296/236	6.1K/6.4K	- /117K	- /23.4K
Avg tgt tokens	55/58	30/31	23/23	144/146	258/258	24/25	281/277	- /6.6K	- /104

Table 3.29: Data statistics on summarization corpora. Source is the domain of dataset. Multi-sents. is whether the summaries are multiple sentences or not. All statistics are divided by Train/Test except for BookSum and MScript.

- CNNDM (Nallapati et al., 2016): contains 300K number of online news articles. It has multiple sentences (4.0 on average) as a summary.
- Newsroom (Grusky et al., 2018): contains 1.3M news articles and written summaries by authors and editors from 1998 to 2017. It has both extractive and abstractive summaries.
- XSum (Narayan et al., 2018a): has news articles and their single but abstractive sentence summaries mostly written by the original author.
- PeerRead (Kang et al., 2018a): consists of scientific paper drafts in top-tier computer science venues as well as arxiv.org. We use full text of introduction section as source document and of abstract section as target summaries.
- PubMed (Kedzie et al., 2018): is 25,000 medical journal papers from the PubMed Open Access Subset.[62] Unlike PeerRead, full paper except for abstract is used as source documents.
- MScript (Gorinski and Lapata, 2015): is a collection of movie scripts from ScriptBase corpus and their corresponding user summaries of the movies.
- BookSum (Mihalcea and Ceylan, 2007): is a dataset of classic books paired to summaries from Grade Saver[63] and Cliff's Notes[64]. Due to a large number of sentences, we only choose the first 1K sentences for source document and the first 50 sentences for target summaries.
- Reddit (Ouyang et al., 2017): is a collection of personal posts from reddit.com. We use a single abstractive summary per post. The same data split from Kedzie et al. (2018) is used.
- AMI (Carletta et al., 2005): is documented meeting minutes from a hundred hours of recordings and their abstractive summaries.

Table 3.29 summarizes the characteristics of each dataset. We note that the Gigaword (Graff et al., 2003), New York Times[65], and Document Understanding Conference (DUC)[66] are also popular datasets commonly used in summarization analyses, though here we exclude them as they represent only additional collections of news articles, showing similar tendencies to the other news datasets such as CNNDM.

[62] https://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/
[63] http://www.gradesaver.com
[64] http://www.cliffsnotes.com/
[65] https://catalog.ldc.upenn.edu/LDC2008T19
[66] http://duc.nist.gov

3.9.5 Analysis on Corpus Bias

We conduct different analyses of how each corpus is biased with respect to the sub-aspects. We highlight some key findings for each sub-section.

		CNNDM			NewsRoom			XSum			PeerRead			PubMed			Reddit			AMI			BookSum			MScript		
		R	VO	SO	R	VO	SO	R	VO	SO	R	VO	SO	R	VO	SO	R	VO	SO	R	VO	SO	R	VO	SO	R	VO	SO
	RANDOM	19.1	18.6	14.6	10.1	2.1	9.0	9.3	-	8.4	27.9	42.5	26.2	30.1	46.9	13.0	11.8	-	11.3	12.0	39.3	2.4	29.4	85.8	4.9	8.1	25.2	0.1
	ORACLE	42.8	-	-	48.1	-	-	19.6	-	-	46.3	-	-	47.0	-	-	30.0	-	-	32.0	-	-	38.9	-	-	24.2	-	-
POSITION	First	**30.7**	13.1	**30.7**	**32.2**	**4.4**	**37.8**	9.1	-	8.7	**32.0**	40.7	**30.3**	27.6	44.3	13.8	**15.3**	-	**19.9**	11.4	48.0	**3.8**	**29.1**	85.1	**7.4**	6.9	12.4	**0.7**
	Last	16.4	18.6	8.2	7.7	1.9	4.4	8.3	-	7.0	28.9	38.5	27.0	28.9	45.2	14.0	11.2	-	10.7	7.8	42.1	2.0	26.5	85.3	3.3	**8.8**	19.5	0.2
	Middle	21.5	18.7	11.8	12.4	1.9	5.6	9.1	-	9.1	29.7	40.7	22.8	28.9	45.9	12.3	11.5	-	7.1	11.1	36.4	2.3	27.9	83.0	4.9	8.0	23.9	0.1
DIVERS.	ConvFall	21.6	**57.7**	15.0	10.6	4.2	7.3	8.4	-	8.0	29.8	**77.5**	25.9	28.2	**93.5**	11.2	11.6	-	7.5	**14.0**	**98.6**	2.4	16.9	**99.7**	2.2	8.5	**59.2**	0.2
	Heuris.	21.4	19.8	14.6	10.5	2.4	7.6	8.4	-	8.1	29.2	36.6	24.8	27.5	59.7	10.5	11.5	-	7.1	10.7	66.0	2.4	26.9	99.7	4.5	6.4	5.7	0.2
IMPORT.	NNear.	22.0	3.3	16.6	13.5	0.5	10.0	**9.8**	-	**10.1**	30.6	8.4	26.7	**31.8**	9.3	**15.5**	13.8	-	12.2	1.3	0.2	0.1	27.9	1.5	5.1	8.7	0.9	0.3
	KNear.	23.0	3.9	17.7	14.0	0.7	10.9	9.3	-	9.1	30.6	9.9	27.0	29.6	10.5	15.0	10.4	-	8.5	0.0	0.1	0.0	21.8	1.4	3.7	0.6	0.0	0.1

Table 3.30: Comparison of different corpora w.r.t the three sub-aspects: POSITION, DIVERSITY, and IMPORTANCE. We averaged R1, R2, and RL as R (See Appendix for full scores). Note that volume overlap (VO) doesn't exist when target summary has a single sentence. (i.e., XSum, Reddit)

Multi-aspect analysis Table 3.30 shows a comparison of the three aspects for each corpus where we include random selection and the oracle set. For each dataset metrics are calculated on a test set except for BookSum and AMI where we use train+test due to the smaller sample size.

Earlier isn't always better. Sentences selected early in the source show high ROUGE and SO on CNNDM, Newsroom, Reddit, and BookSum, but not in other domains such as medial journals and meeting minutes, and the condensed news summaries (XSum). For summarization of movie scripts in particular, the last sentences seem to provide more important summaries.

XSum requires much IMPORTANCE than other corpora. Interestingly, the most powerful algorithm for XSum is N-Nearest. This shows that summaries in XSum are indeed collected by abstracting multiple important contents into single sentence, avoiding the position bias.

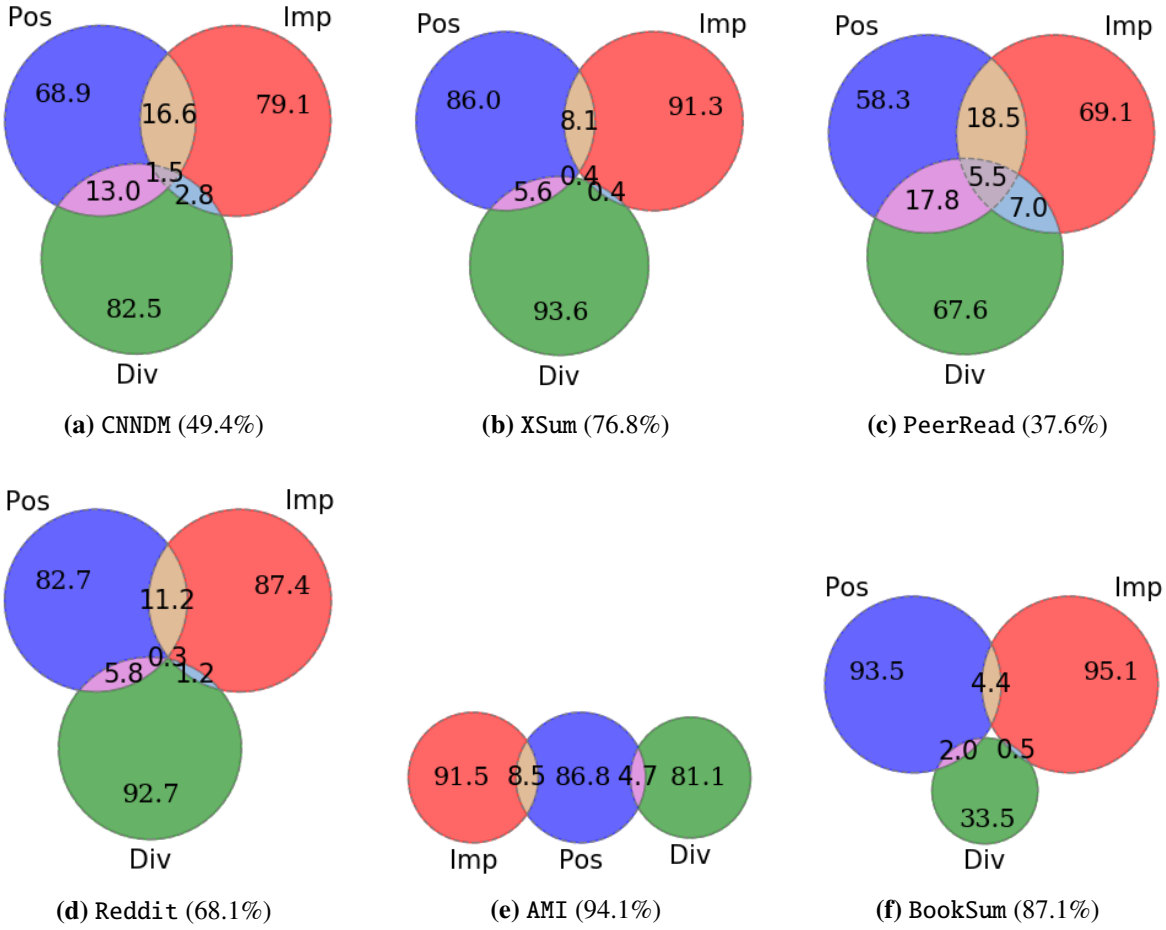
First, ConvexFall, and N-Nearest tend to work better than the other algorithms for each aspect. First is better than Last or Middle in new articles except for XSum and personal posts, while not in academic papers (i.e., PeerRead, PubMed) and meeting minutes. ConvexFall finds the set of sentences that maximize the semantic volume overlap with the target sentences better than the heuristic one.

ROUGE and SO show similar behavior, while VO does not. In most evaluations, ROUGE scores are linear to SO ratios as expected. However, VO has high variance across algorithms and aspects. This is mainly because the semantic volume assumption maximizes the semantic diversity, but sacrifices other aspects like importance by choosing the outlier sentences over the convex hull.

Social posts and news articles are biased to the position aspect while the other two aspects appear less relevant. (Figure 3.20 (a)) However, XSum requires all aspects equally but with relatively less relevant to any of aspects than the other news corpora.

Paper summarization is a well-balanced task. The variance of SO across the three aspects in PeerRead and PubMed is relatively smaller than other corpora. This indicates that abstract summary of the input paper requires the three aspects at the same time. PeerRead has relatively higher SO then PubMed because it only summarize text in Introduction section, while PubMed summarize whole paper text, which is much difficult (almost random performance).

Conversation, movie script and book summarization are very challenging. Conversation of spoken meeting minutes includes a lot of witty replies repeatedly (e.g., 'okay.' , 'mm -hmm.' , 'yeah.'), causing importance and diversity measures to suffer. MScript and BookSum which include very long input document seem to be extremely difficult task, showing almost random performance.



(a) CNNDM (49.4%) **(b)** XSum (76.8%) **(c)** PeerRead (37.6%)

(d) Reddit (68.1%) **(e)** AMI (94.1%) **(f)** BookSum (87.1%)

Figure 3.22: Intersection of averaged summary sentence overlaps across the sub-aspects. We use First for Position, ConvexFall for Diversity, and N-Nearest for Importance. The number in the parenthesis called *Oracle Recall* is the averaged ratio of how many the oracle sentences are **NOT** chosen by union set of the three sub-aspect algorithms. Other corpora are in Appendix with their Oracle Recalls: Newsroom(54.4%), PubMed (64.0%) and MScript (99.1%).

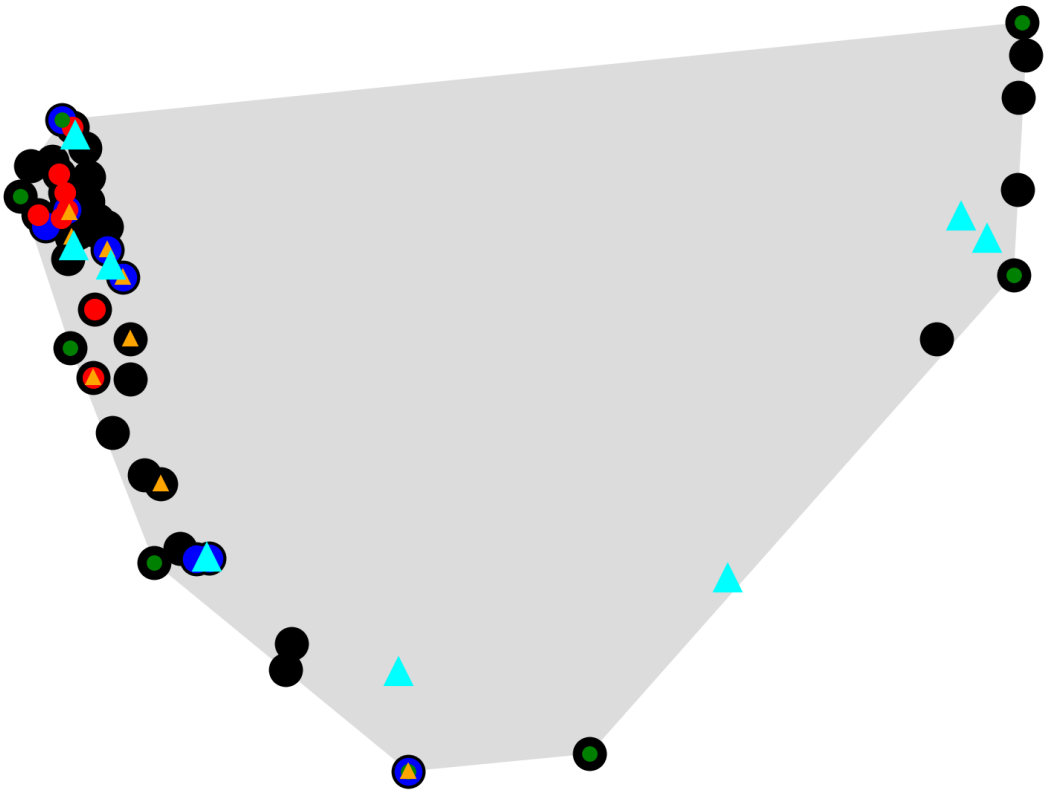


Figure 3.23: PCA projection of extractive summaries chosen by multiple aspects of algorithms (CNNDM). Source and target sentences are black circles (●) and cyan triangles, respectively. The blue, green, red circles are summary sentences chosen by `First`, `ConvexFall`, `NN`, respectively. The yellow triangles are the oracle sentences. Shaded polygon represents a ConvexHull volume of sample source document. Best viewed in color. Please find more examples in Appendix.

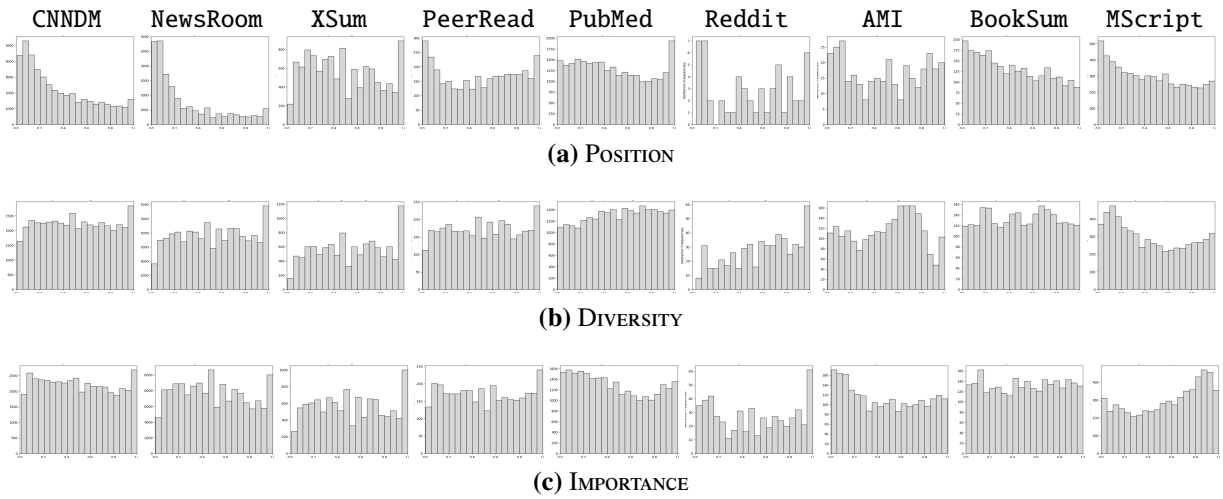
Intersection between the sub-aspects Averaged ratios across the sub-aspects do not capture how the actual summaries overlap with each other. Figure 3.22 shows Venn diagrams of how sets of summary sentences chosen by different sub-aspects are overlapped each other on average.

`XSum`, `BookSum`, and `AMI` have high Oracle Recall. If we develop a mixture model of the three aspects, the Oracle Recall means its upper bound, meaning that another sub-aspect should be considered regardless of the mixture model. This indicates that existing procedures are not enough to cover the Oracle sentences. For example, `AMI` and `BookSum` have a lot of repeated noisy sentences, some of which could likely be removed without a significant loss of pertinent information.

Importance and Diversity are less overlapped with each other. This means that important sentences are not always diverse sentences, indicating that they should be considered together.

Summaries in a embedding space Figure 3.23 shows two dimensional PCA projections of a document in `CNNDM` on the embedding space.

Source sentences are clustered on the convexhull border, not in the middle. Target summaries reflect different sub-aspects according to the sample and corpora. For example, many



(a) POSITION

(b) DIVERSITY

(c) IMPORTANCE

Figure 3.24: Sentence overlap proportion of each sub-aspect (row) with the oracle summary across corpora (column). y-axis is the frequency of overlapped sentences with the oracle summary. X-axis is the normalized RANK of individual sentences in the input document where size of bin is 0.05. E.g., the first / the most diverse / the most important sentence is in the first bin. If earlier bars are frequent, the aspect is positively relevant to the corpus.

target sentences in `CNNDM` are near by `First-k` sentences.

Single-aspect analysis We calculate the frequency of source sentences overlapped with the oracle summary where the source sentences are ranked differently according to the algorithm of each aspect (See Figure 3.24). Heavily skewed histograms indicate that oracle sentences are positively (right-skewed) or negatively (left-skewed) related to the sub-aspect.

In most cases, some oracle sentences are overlapped to the first part of the source sentences. Even though their degrees are different, oracle summaries from many corpora (i.e, `CNNDM`, `NewsRoom`, `PeerRead`, `BookSum`, `MScript`) are highly related to the POSITION. Compared to the other corpora, `PubMed` and `AMI` contain more top-ranked important sentences in their oracle summaries. News articles and papers tend to find oracle sentences without DIVERSITY (i.e., right-skewed), meaning that non-diverse sentences are frequently selected as part of the oracle.

We also measure how many *new* words occur in abstractive target summaries, by comparing overlap between oracle summaries and document sentences (Table 3.31). One thing to note is that `XSum` and `AMI` have less *new* words in their target summaries. On the other hand, paper datasets (i.e., `PeerRead` and `PubMed`) include a lot, indicating that abstract text in academic paper is indeed "abstract".

3.9.6 Analysis on System Bias

We study how current summarization systems are biased with respect to three sub-aspects. In addition, we show that a simple ensemble of systems shows comparable performance to the single-aspect systems.

	R(O,T)	O∩T		T\S	
		Unigram	Bigram	Unigram	Bigram
CNNDM	42.8	66.0	36.4	14.7	5.7
Newsroom	48.1	60.7	43.4	7.8	3.4
XSum	19.6	30.4	6.9	8.4	1.2
PeerRead	46.3	48.5	27.2	20.1	8.8
PubMed	47.0	52.1	27.7	16.7	6.7
Reddit	30.0	41.0	16.4	13.8	3.8
AMI	32.0	28.1	8.5	10.6	1.5
BookSum	38.9	25.6	8.9	6.7	1.7
MScript	38.9	13.9	4.0	0.3	0.1

Table 3.31: ROUGE of oracle summaries and averaged N-gram overlap ratios. O, T and S are a set of N-grams from ORACLE, TARGET and SOURCE document, respectively. **R(O,T)** is the averaged ROUGE between oracle and target summaries, showing how similar they are. **O∩T** shows N-gram overlap between oracle and target summaries. The higher the more overlapped words in between. **T\S** is a proportion of N-grams in target summaries not occurred in source document. The lower the more abstractive (i.e., new words) target summaries.

Existing systems. We compare various extractive and abstractive systems: For extractive systems, we use *K-Means* (Lin and Bilmes, 2010), Maximal Marginal Relevance (*MMR*) (Carbonell and Goldstein, 1998), *cILP* (Gillick and Favre, 2009; Boudin et al., 2015), *TexRank* (Mihalcea and Tarau, 2004), *LexRank* (Erkan and Radev, 2004) and three recent neural systems; *CL* (Cheng and Lapata, 2016), *SumRun* (Nallapati et al., 2017), and *S2SExt* (Kedzie et al., 2018). For abstractive systems, we use *WordILP* (Banerjee et al., 2015) and four neural systems; *S2SAbs* (Rush et al., 2015), *Pointer* (See et al., 2017), *Teacher* (Bengio et al., 2015), and *RL* (Paulus et al., 2017). The detailed description and experimental setup for each algorithm are in Appendix.

Proposed ensemble systems. Motivated by the sub-aspect theory (Lin and Bilmes, 2012, 2011), we combine different types of systems together from two different pools of extractive systems: ASP from the three best algorithm from each aspect and EXT from all extractive systems. For each combination, we choose the sumary sentences randomly among the union set of the predicted sentences (rand) or the most frequent unique sentences (topk).

Results. Table 3.32 shows a comparison of existing and proposed summarization systems on the set of corpora in §3.9.4 except for Newsroom[67]. Neural extractive systems such as *CL*, *SumRun* and *S2SExt* outperform the others in general. *LexRank* is highly biased toward the position aspect. On the other hand, *MMR* is extremely biased to the importance aspect on XSum and Reddit. Interestingly, neural extractive systems are somewhat balanced compared to the others. Ensemble systems seem to have the three sub-aspects in balance, compared to the neural extractive systems. They also outperform the others (either ROUGE or SO) on five out of eight

[67]We exclude it because of its similar behavior as CNNDM.

		CNNDM			XSum			PeerRead			PubMed			Reddit			AMI			BookSum			MScript		
		R	SO	R(P/D/I)	R	SO	R(P/D/I)	R	SO	R(P/D/I)	R	SO	R(P/D/I)	R	SO	R(P/D/I)	R	SO	R(P/D/I)	R	SO	R(P/D/I)	R	SO	R(P/D/I)
extractive	**KMeans**	22.2	16.3	14/22/34	9.8	10.0	14/8/90	30.9	28.3	24/28/38	30.6	14.2	31/40/46	14.0	12.5	10/2/82	12.3	2.5	9/6/7	27.2	4.6	5/2/14	9.1	0.3	0/0/9
	MMR	21.6	15.2	12/24/30	9.8	10.0	14/8/**97**	29.6	24.9	26/29/35	30.2	12.9	33/35/42	13.6	11.5	10/3/**88**	12.3	2.5	9/6/7	29.1	6.1	4/0/13	9.5	0.2	0/0/28
	TexRank	19.6	10.3	34/27/27	9.9	8.5	19/11/16	23.9	12.4	32/32/32	18.0	1.7	19/21/20	17.7	16.7	13/9/15	11.1	0.0	17/20/6	6.7	0.0	8/14/8	8.2	0.2	5/9/8
	LexRank	29.3	29.5	71/29/32	11.2	11.9	**61**/15/19	29.0	24.6	**66**/35/38	26.3	7.7	**56**/27/28	18.7	18.8	46/11/19	8.0	0.2	36/21/12	10.5	0.8	20/20/13	12.7	**0.5**	20/9/9
	wILP	23.1	15.6	27/28/29	11.1	2.1	28/19/21	20.2	16.0	23/27/26	15.6	6.0	14/20/18	17.4	13.5	42/16/20	5.1	0.6	17/18/17	4.3	1.3	5/12/7	6.8	0.1	6/8/6
	CL	**31.2**	30.0	**86**/29/31	**11.8**	**14.3**	25/13/19	31.3	21.8	55/35/38	26.3	9.2	41/26/26	19.4	24.0	23/14/23	23.1	10.3	19/23/5	-	-	-/-/-	14.0	0.2	6/8/7
	SumRun	30.5	27.1	68/29/31	11.6	13.1	14/13/19	**34.0**	20.5	38/36/37	29.4	10.8	27/28/27	**20.2**	19.8	23/12/21	**23.8**	11.4	21/23/6	-	-	-/-/-	14.4	0.0	5/9/9
	S2SExt	30.4	28.3	74/28/31	12.0	14.2	17/13/19	33.9	21.1	43/35/37	29.6	10.8	26/28/28	21.5	**34.4**	27/12/26	23.4	11.9	21/24/6	-	-	-/-/-	14.3	0.0	7/9/8
abstractive	**cILP**	27.8	x	43/31/32	10.9	x	49/15/18	28.2	x	35/36/38	27.8	x	23/29/30	17.7	x	**53**/15/17	12.5	x	22/33/10	7.9	x	9/19/12	10.6	x	5/7/7
	S2SAbs	16.3	x	4/4/4	10.4	x	8/7/8	9.9	x	9/9/9	10.2	x	10/10/10	11.9	x	11/7/8	20.3	x	9/12/1	-	-x	-/-/-	14.0	x	6/8/8
	+Pointer	23.9	x	20/13/14	15.6	x	12/11/12	13.6	x	13/13/13	11.2	x	11/12/11	14.3	x	14/10/12	23.0	x	11/13/1	-	-x	-/-/-	10.0	x	6/7/7
	+Teacher	29.7	x	33/21/22	17.0	x	12/10/12	8.7	x	8/8/8	11.3	x	12/12/11	15.3	x	15/10/11	20.2	x	9/13/1	-	-x	-/-/-	16.0	x	7/10/8
	+RL	30.2	x	34/23/24	18.1	x	12/11/12	30.1	x	30/29/28	12.9	x	13/14/13	16.7	x	1/1/14	23.6	x	11/13/2	-	-x	-/-/-	**16.2**	x	7/10/8
ensemble	ASP(rand)	23.3	19.5	40/**38**/**38**	9.0	9.0	40/**39**/38	29.6	25.5	54/49/52	29.5	13.5	49/**47**/**51**	12.5	5.2	21/11/22	8.9	0.9	44/**50**/**20**	**29.8**	**6.4**	57/33/55	8.4	0.4	**32**/36/37
	ASP(topk)	29.1	**30.4**	71/31/31	9.0	8.8	43/**39**/38	30.5	28.2	63/**54**/**57**	29.7	14.0	55/48/52	12.3	15.6	41/**41**/38	9.9	1.5	**99**/24/11	29.6	6.2	**58**/**34**/**56**	8.3	**0.5**	30/**37**/**38**
	EXT(rand)	24.2	20.2	39/25/27	10.2	10.9	17/13/23	29.4	23.5	42/37/39	31.7	16.0	37/34/38	14.2	17.7	22/12/13	18.7	5.1	21/28/8	28.6	5.4	37/24/42	6.7	0.0	5/9/13
	EXT(topk)	29.4	30.3	58/25/28	11.0	11.8	18/10/37	33.0	**33.0**	54/39/44	**34.1**	**20.5**	41/35/40	16.4	20.8	21/11/52	**23.8**	**13.4**	23/27/6	28.5	5.2	37/24/43	7.4	0.0	6/8/11

Table 3.32: Comparison of different systems using the averaged ROUGE scores (1/2/L) with target summaries (R) and averaged oracle overlap ratios (SO, only for extractive systems). We calculate R between systems and selected summary sentences from each sub-aspect (R(P/D/I)) where each aspect uses the best algorithm: First, ConvexFall and NNearest. R(P/D/I) is rounded by the decimal point. - indicates the system has too few samples to train the neural systems. x indicates SO is not applicable because abstractive systems have no sentence indices. The best score for each corpora is shown in bold with different colors.

datasets.

3.9.7 Conclusion and Future Directions

We define three sub-aspects of text summarization: position, diversity, and importance. We analyze how different domains of summarization dataset are biased to these aspects. We observe that news articles strongly reflect the position aspect, while the others do not. In addition, we investigate how current summarization systems reflect these three sub-aspects in balance. Each type of approach has its own bias, while neural systems rarely do. Simple ensembling of the systems shows more balanced and comparable performance than single ones.

We summarize actionable messages for future summarization research:

- Different domains of datasets except for news articles pose new challenges to the appropriate design of summarization systems. For example, summarization of conversations (e.g., `AMI`) or dialogues (`MSCript`) need to filter out repeated, rhetorical utterances. Book summarization (e.g., `BookSum`) is very challenging due to its extremely large document size. Here current neural encoders suffer from computation limits.
- Summarization systems to be developed should clearly state their computational limits as well as effectiveness in each aspect and in each corpus domain. A good summarization system should reflect different kinds of the sub-aspects harmoniously, regardless of corpus bias. Developing such bias-free or robust models can be very important for future directions.
- Nobody has clearly defined the deeper nature of meaning abstraction yet. A more theoretical study of summarization, and the various aspects, is required. A recent notable example is

Peyrard (2019a)'s attempt to theoretically define different quantities of `importance` aspect, and demonstrate the potential of the framework on an existing summarization system. Similar studies can be applied to other aspects and their combinations in various systems and different domains of corpora.

- One can repeat our bias study on evaluation metrics. Peyrard (2019b) showed that widely used evaluation metrics (e.g., ROUGE, Jensen-Shannon divergence) are strongly mismatched in scoring summary results. One can compare different measures (e.g., n-gram recall, sentence overlaps, embedding similarities, word connectedness, centrality, importance reflected by discourse structures), and study bias of each with respect to systems and corpora.

3.10 Conclusion

Our proposed text planning models achieved significant improvements over a variety of multi-sentence generation tasks. The structural guidance on horizontal planning helps multiple sentences (or phrases) be connected more coherently via causal relations (Kang et al., 2017b), discourse relations (Kang et al., 2019c), content goals (Kang and Hovy, 2020), and latent policies (Kang et al., 2019a). For vertical planning, we created a new dataset (Kang et al., 2018b) for aspect-specific review generation and conducted researches on modeling the multi-intents in email response generation (Kang et al., 2017a; Lin et al., 2018). For hierarchical planning, we provided a comprehensive sub-aspect bias analysis on current summarization systems and corpora (Kang et al., 2019d).

Text planning a cognitive process of structuring multi-sentence text, as humans are naturally able to do. The hierarchical, horizontal and vertical expansion of meaning could be applied to many practical applications such as SmartReply, teaching dialogue, explanation systems, news generation, and more. The structured text from the planning could make our communication more coherent, diverse, and abstractive, and thus effective.

Chapter 4

Stylistically-Appropriate Generation

In this chapter, we will incorporate the *style* facet into language generation. Again, all other facets of language generation were held constant, allowing us to focus on studying the effect of the style facet individually.

Every natural text is written in some style. To generate natural-sounding language, one must represent the additional kinds of information which guide the interpersonal formulation of communication. We call this large heterogeneous group the *style* facet. Due to the relatedness of interpersonal dialogue, style is the most rigorous facet. In language generation, appropriately varying the style of a text often conveys more information than is contained in the literal meaning of the words (Hovy, 1987).

Target style: *the relationship with the listener* Alice to her boss: "I'll complete my project by noon." Bob to a friend: "Dude, give me a few more minutes!!"
Target style: *persona of the speaker (age / education)* Alice (a 60-year old with a Ph.D.): "Pleased to meet you." Bob (a 14-year old in high-school): "Whatsup bro!"
Target style: *geography of the speaker* Alice (location: California US): "Hella good!" Bob (location: Southern US): "That's the bee's knees."
Target style: *sarcastic usage* Clinton: "'Great' year Trump, 'great' year." Bob: "It was a great year, Trump!"
Target style: speaker has a *romantic interest in*, or *relationship with the listener* Alice to Bob (crush or spouse): "I love you more every day."

Table 4.1: Some example textual variation triggered by various style types.

Table 4.1 shows different choices of text depending on various interpersonal factors: the relationship with the listener, the speaker's personas or geography, figurative use of language, or feelings toward the listener. Sometimes, multiple styles are used simultaneously. For example, the

last utterance "I love you more every day." is generated if the speaker has romantic feelings for the listener and/or a romantic relationship with the listener, such as being married or dating. While preserving the original meaning of given text, the stylistic variations in the text primarily manifest themselves at different levels such lexical, like word choice or pronoun dropping, syntactic, like a preference for the passive voice, and even pragmatic, like interpersonal distance with hearer (DiMarco and Hirst, 1990).

Due to its relation to social idiosyncrasies and linguistic variations, style is the most complex facet to define, understand its textual variation, and empirically validate. Social aspects of style in language have been studied for many decades in various fields such as linguistics, sociology, philosophy, and sociolinguistics. We begin by summarizing the prior linguistic theories on style in the next section.

4.1 Literature Survey

The philosopher Bakhtin developed the concept of dialogism (Bakhtin, 1981), which describes the generation of meaning through the primacy of context over text, the hybrid nature of language and the relation between utterances. One of his theories claims that culture and communication are inextricably linked, because one's understanding of a given utterance, text, or message, is contingent upon one's cultural background and experience (Danow, 1991). Bakhtin's believed that the meaning of language is socially determined, in that utterances reflect social values and that their meaning depended upon their relationship with other utterances.

In sociology and sociolinguistics, indexicality is the phenomenon where a sign points to some object, but only in the context in which it occurs. Words and expressions in language often derive some part of their referential or nonreferential meaning from indexicality (Silverstein, 2003). Nonreferential indexicalities include the speaker's gender, affect (Besnier, 1990), power, solidarity (Brown et al., 1960), social class, and identity (Ochs, 1990). The more general sociolinguistic views of style and identity are well summarized by Coupland (2007); Johnstone (2010).

First-order indexicality can be defined as the first level of pragmatic meaning that is drawn from an utterance. Second-order indexicality is concerned with the connection between linguistic variables and the metapragmatic meanings that they encode Silverstein (2003). For example, imagine a woman is walking down the street in New York City and stops to ask somebody where to find a McDonald is[1]. He responds to her in a heavy Brooklyn accent, causing her to consider possible personal characteristics that might be indexed by it, such as the man's intelligence, economic situation, and other non-linguistic aspects. The power of language to encode these preconceived stereotypes based only on an accent is an example of second-order indexicality, representing of a more complex system of indexical form than first-order indexicality.

Silverstein (2003) argued that indexical order can transcend levels such as second-order indexicality, going on to discuss higher-order indexicality. Building on Silverstein's notion of indexical order, Eckert (2008) built the notion that linguistic variables index a social group, which leads to the indexing of certain traits stereotypically associated with members of that group. For

[1]This example appears in the Wikipedia entry for "Indexicality," `https://en.wikipedia.org/wiki/Indexicality`, accessed March 14, 2020.

example, in New York in the 1960s, Labov (1966) showed that the clear articulation of postvocalic [r] in words like "fourth" and "floor" indexed a higher class (in New York), whereas its absence of it indexed a lower class. Eckert (2000) argued that style change creates a new persona, impacting a social landscape. More recently, Eckert (2019) presented the expression of social meaning as a continuum of decreasing reference and increasing performativity, with sociolinguistic variation at the performative extreme. The meaning of sociolinguistic variables is based in their form and their social source, constituting a cline of 'interiority' from public social facts about the speaker to more internal, personal affective states.

Language is situational (Goffman et al., 1978). Goffman's analysis of face-to-face interactions showed that all of the actions we take in our daily life including speaking, and the interpretations and meanings we assign those actions, are fundamentally social in nature. For instance, our impressions of people rely upon which region we think they are from, in what situation we encounter them, and how they behave, which is related to whether the speaker perceives the listener as an audience, as with front stage actions.

More recently, Jaffe et al. (2009) built the notion that stance indexicalities become "short-circuited", so that ways of speaking become associated with situations and the opinion of the speaker. Styles of speaking are thus shorthand for bundles of habitually taken stances. Yoder (2019) suggested empirical observations on how the identity of language users affects language, and how language positions the identity of the speaker to others.

Despite the extensive theoretical research about style, very little work has been done on different styles co-varying in a single text and how such might be dependent on each other.

4.2 Proposed Approach: *Cross-Stylization*

4.2.1 Scope

Style is formed by a complex combination of different stylistic factors, including formality markers, emotions, metaphors, etc. Some factors implicitly reflect the author's personality, while others are explicitly controlled by the author's choices in order to achieve a personal or social goal. One cannot form a complete understanding of a text and its author without considering these factors. The factors combine and co-vary in complex ways to form styles. Studying the nature of the co-varying combinations sheds light on stylistic language in general, sometimes called *cross-style language understanding*.

Stylistically appropriate NLG systems should act like an orchestra conductor. An orchestra is a large ensemble of instruments that jointly produces a single, integrated piece of music. What we hear as music is a complex interaction between individual instruments coordinated by the conductor who controls variables like score and tempo. Some instruments are in the same category, such as bowed strings for the violin and cello. Similarly, text is an output that reflects a complex combination of different style factors where each has its own lexical choices, but even though factors are dependent on each other. We believe modeling this complex combination and finding the dependencies between styles or content and style is an crucial step toward being a maestro of cross-style language generation.

Instead of selecting a specific style type and conducting an in-depth study on its textual vari-

ation, this thesis focuses on conducting a more comprehensive study on the co-varying patterns of multiple styles and their inter-dependencies.

4.2.2 Categorization

It is impossible to develop an overarching categorization of every style in every language, although there are a few attempts in prior papers. Hovy (1987) defined different types of pragmatic aspects, such as the relation between listener and speaker and how the interpersonal goals affect the listener's opinions of topic, and rhetorical goals, like formality and force. He also provided practical examples of conversations to show how those aspects are tightly coupled. These are often called pragmatics constraints. Examples include a strong or neutral stance about the topic of a text or a different level of formality.

Biber (1991) categorized components of specific linguistic situations by participant roles and characteristics and the relationships among participants. These could be also divided into roles, such as speaker or listener; personal characteristics, like personality, interests, and mood; and group characteristics, like social class.

Compared to the prior categorizations, our categorization of interpersonal facets is much broader and more comprehensive, from personal and inter-personal to affective and figurative. We describe what types of styles we studied and provide our theoretical categorization by clustering them into two orthogonal dimensions: *social participation* (from personal to interpersonal) and *content coupledness* (from loosely coupled to tightly coupled).

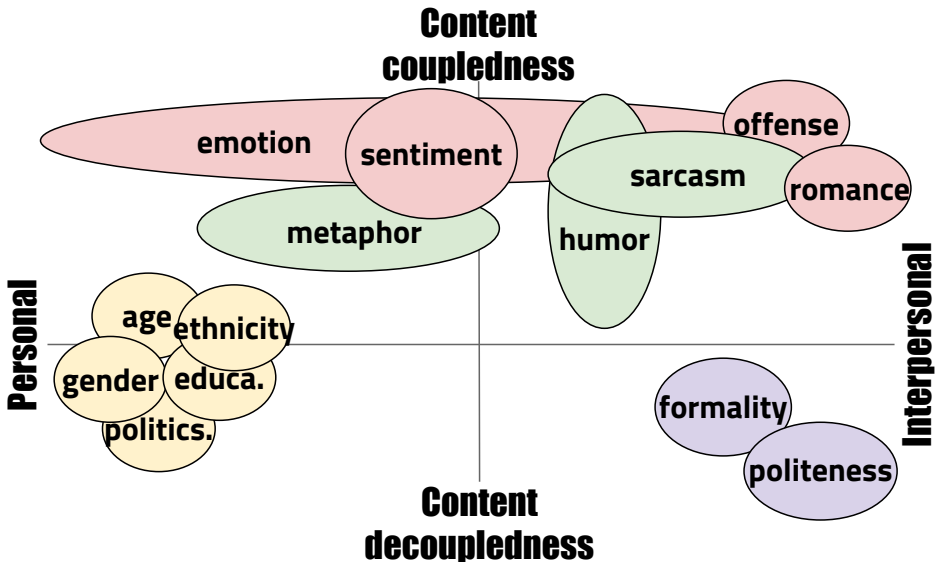


Figure 4.1: A conceptual grouping of styles where x-axis represents a style's social participation, while the y-axis represents the coupledness of the content.

We chose the 13 types of styles based on the xSLUE benchmark corpora (Kang and Hovy, 2019). Then, we hypothesized two orthogonal aspects of style: *social participation* and *content coupledness* (Figure 4.1).

Social participation measures whether a style is related to the speaker or the listener in a conversation. This dimension was studied in Biber (1991); personal style looks at the char-

acteristics of the speaker, such as their personality, while interpersonal style focuses on their relationship with the hearer, such as whether they are friends. In addition to Biber's definition, our view focuses on whether the style affects textual variation implicitly or explicitly. Personal styles, like age and gender, are originally given to a person; her or his text implicitly contains the combination of her or his personal styles (Kang et al., 2019b). Interpersonal styles, like friend, enemy, or boss, are given by their social interactions so the text can be explicitly controlled by the speaker with respect to the hearer. For instance, one can explicitly control the formality of words depending on who one is talking with, while personal characteristics are implicitly contained in your own words without any explicit control. Recently, Calvo and Mac Kim (2013) distinguished how emotion was attributed by comparing the writer's and reader's perspectives.

Content coupledness examines how much the style is influenced by the content of the original text. Ficler and Goldberg (2017) controlled different styles, specifically descriptive and professional, in variations of the same text, regardless of its coupledness to the semantics of the text. However, it is often observed that content words are tightly coupled with their styles (Kang et al., 2019b; Preoţiuc-Pietro and Ungar, 2018). For instance, one can increase or decrease the formality of text regardless of the topic, irrespective of whether the speaker or writer feels emotion about the topic or person in the text.

We then projected the 13 styles over the two dimensions and stretched each style so that they accurately aligned with a broad spectrum of each dimension (Figure 4.1). Personal styles are not biased based on the content because they are implicitly reflected in the text. On the other hand, formality and politeness are interpersonal but loosely coupled on content, because they can vary independently. Emotion can be either personal or interpersonal, while offense and romance are related to an other person, and sentiment is tightly coupled with content. The other three styles, metaphor, sarcasm, and humor, are more complex phenomena which are layered a top others, so are stretched across dimensions.

Inter-personal styles are used to intentionally or unintentionally change the relationship with the listener. E.g., *formality*, *politeness*
Personal styles are used to unconsciously or consciously express the speaker's persona by using a dialect or otherwise identifying his or her group characteristics to the listener. E.g., *age*, *ethnicity*, *gender*, *education level*, *political view*
Figurative styles are used to amplify the message of text by using various figures of speech, making communication more effective, persuasive, or impactful. E.g., *humor*, *sarcasm*, *metaphor*
Affective styles are used to synchronize emotional status across individuals and promote social interaction (Nummenmaa et al., 2012). E.g., *emotion*, *offense*, *romance*, *sentiment*

Table 4.2: Our categorization of styles with their social goals.

Based on these groupings, Table 4.2 categorize styles into four groups: figurative, affective, personal, and inter-personal. In addition, we also added each style group's social goals as extracted from §1.2.5. This figure is driven by our own conjectures, so we anticipate better projects and categorizations to be developed in the future.

4.2.3 Challenges and Proposed Methods

Due to the difficulty of modeling the exact functional variation of stylistic text, we tackle two practical challenges in our style research:

#1: Lack of a parallel dataset for a controlled experiment

One of the challenges in the cross-style study is the lack of an appropriate dataset, leading to invalid model development and uncontrolled experiments. Despite the recent attempts to computationally model stylistic variations, the lack of parallel corpora for style makes it difficult to systematically control the stylistic change as well as evaluate such models.

Kang et al. (2019b) proposed the first parallel dataset for stylistic language, particularly focusing on persona styles. The dataset includes a parallel, annotated stylistic language dataset, which contains approximately 41K parallel sentences in 8.3K parallel stories annotated across different personas. Each persona has several different styles: gender, age, country, political view, education, ethnicity, and time-of-writing. The dataset was collected by human annotators who controlled input denotation, not only preserving the original meaning between texts, but promoting stylistic diversity in the annotations.

We tested the dataset on two interesting applications of style, where PASTEL helps design appropriate experiments and evaluations. First, the goal is to predict a target style, like male or female. Given a text, multiple styles of PASTEL make other external style variables controlled, allowing for a more accurate experimental design. In this situation, a simple supervised model with our parallel text outperforms the unsupervised models using non-parallel text in style transfer. Further details are described in §4.4.

#2: Lack of a multi-style benchmark for a style dependency study

Style is not a single variable, but a combination of variables simultaneously shifting in complex ways. Only a few papers have studied dependency between styles only on the particular group of styles. They include Preoţiuc-Pietro and Ungar (2018)'s study on demographics, Warriner et al. (2013)'s work on emotions, and (Dankers et al., 2019; Mohammad et al., 2016)'s papers on metaphor and emotion. Studying the nature of co-varying combinations and the general understanding of *cross-style language* is the key to better understand how stylistically appropriate language is produced.

Kang and Hovy (2019) provide a benchmark corpus, called **x**SLUE, for cross-style language understanding and evaluation. The benchmark contains text in 15 different styles and 23 classification tasks. For each task, we fine-tuned classifiers using BERT (Devlin et al., 2019) for further analysis. Our analysis shows that some styles are highly dependent on each other, such as

impoliteness and offense, while some domains, such as tweets and political debates, are stylistically more diverse than others, like academic manuscripts. In §4.5, we discuss the technical challenges of cross-style understanding and potential directions for future research. Specifically, we look at cross-style modeling which shares the internal representation for low-resource or low-performance styles and other applications such as cross-style generation.

4.3 Related Work

We survey prior work on parallel style dataset and cross-style study as follows:

Parallel style dataset. Most of recent works transfer styles between sentiment (Fu et al., 2018; Shen et al., 2017; Hu et al., 2017), gender (Reddy and Knight, 2016), two conflicting corpora (e.g, paper and news (Han et al., 2017) or real and synthetic reviews (Lipton et al., 2015)). However, they suffer from semantic drift and limited evaluation for meaning preservation.

Few recent works use parallel text for style transfer between modern and Shakespearean text (Jhamtani et al., 2017), sarcastic and literal tweets (Peled and Reichart, 2017), and formal and informal text (Heylighen and Dewaele, 1999; Rao and Tetreault, 2018). Compared to these, we aim to understand and transfer for variation owing to multiple demographic attributes in conjunction. The multiple styles in conjunction with PASTEL enable an appropriate experiment setting for controlled style classification tasks.

Cross-style language understanding. Some recent works attempted to provide empirical evidence of style dependencies but in a very limited range: Warriner et al. (2013) analyzed emotional norms and their correlation in lexical features of text. Chhaya et al. (2018) studied a correlation of formality, frustration, and politeness but on small samples (i.e., 960 emails). Preoţiuc-Pietro and Ungar (2018) focused on correlation across demographic information (e.g., age, race) with some other factors such as emotions. Dankers et al. (2019); Mohammad et al. (2016) studied the interplay of metaphor and emotion in text. Liu et al. (2010) studied sarcasm detection using sentiment as a sub-problem.

Instead of finding the dependencies, some prior works controlled the confounding style variables to identify the target style: For example, different demographic attributes (e.g., gender, age) are collected in conjunction and controlled on each other for style identification (Bamman et al., 2014; Nguyen et al., 2016), personalized machine translation (Rabinovich et al., 2016), and style transfer (Kang et al., 2019b).

4.4 Parallel Style Language Dataset

4.4.1 Introduction

Hovy (1987) claims that appropriately varying the style of text often conveys more information than is contained in the literal meaning of the words. He defines the roles of styles in text variation by pragmatics aspects (e.g., relationship between them) and rhetorical goals (e.g., formality),

and provides example texts of how they are tightly coupled in practice. Similarly, Biber (1991) categorizes components of conversational situation by participants' characteristics such as their roles, personal characteristics, and group characteristics (e.g., social class). Despite the broad definition of style, this work mainly focuses on one specific aspect of style, *pragmatics aspects in group characteristics of speakers*, which is also called *persona*. Particularly, we look at multiple types of group characteristics in conjunction, such as gender, age, education level, and more.

Stylistic variation in text primarily manifest themselves at the different levels of textual features: lexical features (e.g., word choice), syntactic features (e.g., preference for the passive voice) and even pragmatics, while preserving the original meaning of given text (DiMarco and Hirst, 1990). Connecting such textual features to someone's persona is an important study to understand stylistic variation of language. For example, do highly educated people write longer sentences (Bloomfield, 1927)? Are Hispanic and East Asian people more likely to drop pronouns (White, 1985)? Are elder people likely to use lesser anaphora (Ulatowska et al., 1986)?

To computationally model a meaning-preserved variance of text across styles, many recent works have developed systems that transfer styles (Reddy and Knight, 2016; Hu et al., 2017; Prabhumoye et al., 2018) or profiles authorships from text (Verhoeven and Daelemans, 2014; Koppel et al., 2009; Stamatatos et al., 2018) without parallel corpus of stylistic text. However, the absence of such a parallel dataset makes it difficult both to systematically learn the textual variation of multiple styles as well as properly evaluate the models.

In this paper, we propose a *large scale, human-annotated, parallel stylistic dataset* called PASTEL, with focus on *multiple types of personas in conjunction*. Ideally, annotations for a parallel style dataset should preserve the original meaning (i.e., *denotation*) between reference text and stylistically transformed text, while promoting diversity for annotators to allow their own styles of persona (i.e., *connotation*). However, if annotators are asked to write their own text given a reference sentence, they may simply produce arbitrarily paraphrased output which does not exhibit a stylistic diversity. To find such a proper input setting for data collection, we conduct a denotation experiment in §4.4.2. PASTEL is then collected by crowd workers based on the most effective input setting that balances both meaning preservation and diversity metrics (§4.4.3).

PASTEL includes stylistic variation of text at two levels of parallelism: ≈8.3K annotated, parallel stories and ≈41K annotated, parallel sentences, where each story has five sentences and has 2.63 annotators on average. Each sentence or story has the seven types of persona styles in conjunction: gender, age, ethnics, countries to live, education level, political view, and time of the day.

We introduce two interesting applications of style language using PASTEL: controlled style classification and supervised style transfer. The former application predicts a category (e.g., male or female) of target style (i.e., gender) given a text. Multiplicity of persona styles in PASTEL makes other style variables controlled (or fixed) except the target, which is a more accurate experimental design. In the latter, contrast to the unsupervised style transfer using non-parallel corpus, simple supervised models with our parallel text in PASTEL achieve better performance, being evaluated with the parallel, annotated text.

We hope PASTEL sheds light on the study of stylistic language variation in developing a solid model as well as evaluating the system properly.

4.4.2 Denotation Experiment

Denotation:	**Produced sentences**:
single ref. sentence	the old door with wood was the only direction to the courtyard
story(imgs)	The old wooden door in the stonewall looks like a portal to a fairy tale.
story(imgs.+keyw words)	Equally so, he is intrigued by the heavy wooden door in the courtyard.
Reference sentence: the old wooden door was only one way into the courtyard.	

Table 4.3: Textual variation across different denotation settings. Each sentence is produced by a same annotator. Note that providing reference sentence increases fidelity to the reference while decreases diversity.

We first provide a preliminary study to find the best input setting (or denotation) for data collection to balance between two trade-off metrics: meaning preservation and style diversity.

Preliminary Study

Table 4.3 shows output texts produced by annotators given different input denotation settings. The basic task is to provide an input denotation (e.g., a sentence only, a sequence of images) and then ask them to reproduce text maintaining the meaning of the input but with their own persona.

For instance, if we provide a *single reference sentence*, annotators mostly repeat the input text with a little changes of the lexical terms. This setup mostly preserves the meaning by simply paraphrasing the sentence, but annotators' personal style does not reflect the variation. With a *single image*, on the other hand, the outputs produced by annotators tend to be diverse. However, the image can be explained with a variety of contents, so the output meaning can drift away from the reference sentence.

If a series of consistent images (i.e., a story) is given, we expect a stylistic diversity can be more narrowed down, by grounding it to a specific event or a story. In addition to that, some keywords added to each image of a story help deliver more concrete meaning of content as well as the style diversity.

Experimental Setup

In order to find the best input setting that preserves meaning as well as promotes a stylistic diversity, we conduct a denotation experiment as described in Figure 4.2. The experiment is a subset of our original dataset, which have only 100 samples of annotations.

A basic idea behind this setup is to provide (1) a perceptually common denotation via sentences or images so people share the same context (i.e., denotation) given, (2) a series of them as

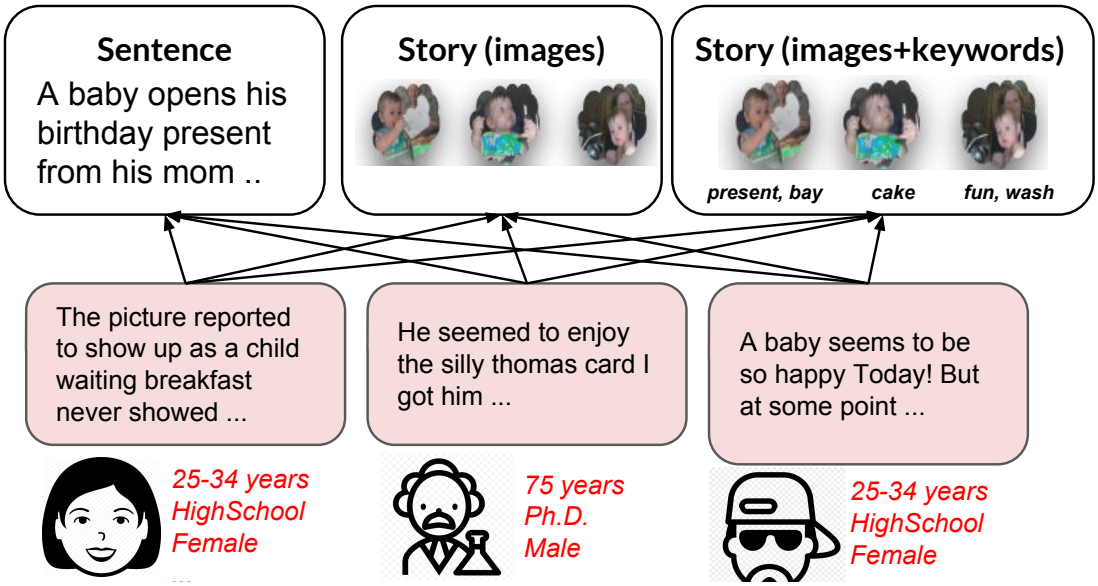


Figure 4.2: Denotation experiment finds the best input setting for data collection, that preserves meaning but diversifies styles among annotators with different personas.

a "story" to limit them into a specific event context, and (3) two modalities (i.e., text and image) for better disambiguation of the context by grounding them to each other.

We test five different input settings[2]: *Single reference sentence*, ***Story*** *(images)*, ***Story*** *(images) + global keywords*, ***Story*** *(images + local keywords)*, and ***Story*** *(images + local keywords + ref. sentence)*.

For the keyword selection, we use RAKE algorithm (Rose et al., 2010) to extract keywords and rank them for each sentence by the output score. Top five uni/bigram keywords are chosen at each story, which are called *global keywords*. On the other hand, another top three uni/bigram keywords are chosen at each image/sentence in a story, which are called *local keywords*. Local keywords for each image/sentence help annotators not deviate too much. For example, *local keywords* look like *(restaurant, hearing, friends)* → *(pictures, menu, difficult)* → *(salad, corn, chose)* for three sentences/images, while *global keywords* look like *(wait, salad, restaurant)* for a story of the three sentences/images.

We use Visual Story Telling (ViST) (Huang et al., 2016) dataset as our input source. The dataset contains stories, and each story has five pairs of images and sentences. We filter out stories that are not temporally ordered using the timestamps of images. The final number of stories after filtering the non-temporally-ordered stories is 28,130. For the denotation experiment, we only use randomly chosen 100 stories. The detailed pre-processing steps are described in Appendix.

Measuring Meaning Preservation & Style Diversity across Different Denotations

For each denotation setting, we conduct a quantitative experiment to measure the two metrics: meaning preservation and style diversity. The two metrics pose a trade-off to each other. The

[2]Other settings like *Single reference image* are tested as well, but they didn't preserve the meaning well.

Table 4.4: Denotation experiment to find the best input setting (i.e., meaning preserved but stylistically diverse). **story-level** measures the metrics for five sentences as a story, and **sentence-level** per individual sentence. Note that *single reference sentence* setting only has sentence level. For every metrics in both meaning preservation and style diversity, the higher the better. The **bold** number is the highest, and the underlined is the second highest.

	denotation settings	**Style Diversity**	**Meaning Preservation**	
		E(GM)	METEOR	VectorExtrema
sentence-level	*single ref. sentence*	2.98	**0.37**	**0.70**
	***story**(images)*	2.86	0.07	0.38
	***story**(images) + global keywords*	2.85	0.07	0.39
	***story**(images + local keywords)*	**3.07**	0.17	0.53
	***story**(images + local keywords + ref. sentence)*	2.91	0.21	0.43
story-level	***story**(images)*	4.43	0.1	0.4
	***story**(images) + global keywords*	4.43	0.1	0.42
	***story**(images + local keywords)*	**4.58**	0.19	**0.55**
	***story**(images + local keywords + ref. sentence)*	4.48	**0.22**	0.44

best input setting then is one that can capture both in appropriate amounts. For example, we want meaning of the input preserved, while lexical or syntactic features (e.g., POS tags) can vary depending on annotator's persona. We use the following automatic measures for the two metrics:

Style Diversity measures how much produced sentences (or stories) differ amongst themselves. Higher the diversity, better the stylistic variation in language it contains. We use an entropy measure to capture the variance of n-gram features between annotated sentences: Entropy (Gaussian-Mixture) that combines the N-Gram entropies (Shannon, 1951) using Gaussian mixture model (N=3).

Meaning Preservation measures semantic similarity of the produced sentence (or story) with the reference sentence (or story). Higher the similarity, better the meaning preserved. We use a hard-measure, METEOR (Banerjee and Lavie, 2005), that calculates F-score of word overlaps between the output and reference sentences[3]. Since the hard measures do not take into account all semantic similarities [4], we also use a soft measure, VectorExtrema (VecExt) (Liu et al., 2016). It computes cosine similarity of averaged word embeddings (i.e., GloVe (Pennington et al., 2014)) between the output and reference sentences.

Table 4.4 shows results of the two metrics across different input settings we define. For the sentence level, as expected, *single reference sentence* has the highest meaning preservation across all the metrics because it is basically paraphrasing the reference sentence. In general, ***Story*** (*im-*

[3]Other measures (e.g., BLEU (Papineni et al., 2002), ROUGE (Lin and Hovy, 2003)) show relatively similar performance.

[4]METEOR does consider synonymy and paraphrasing but is limited by its predefined model/dictionaries/resources for the respective language, such as Wordnet

Figure 4.3: Final denotation setting for data collection: an event that consists of a series of five images with a handful number of keywords. We ask annotators to produce text about the event for each image.

ages + local keywords) shows a great performance with the highest diversity regardless of the levels, as well as the highest preservation at the soft measure on the story-level. Thus, we use ***Story***(*images+local keywords*) as the input setting for our final data collection, which has the most balanced performance on both metrics. Figure 4.3 shows an example of our input setting for crowd workers.

4.4.3 PASTEL: A Parallelly Annotated Dataset for Stylistic Language Dataset

We describe how we collect the dataset with human annotations and provide some analysis on it.

Annotation Schemes

Our crowd workers are recruited from the Amazon Mechanical Turk (AMT) platform. Our annotation scheme consists of two steps: (1) ask annotator's demographic information (e.g., gender, age) and (2) given an input denotation like Figure 4.3, ask them to produce text about the denotation with their own style of persona (i.e., connotation).

In the first step, we use seven different types of persona styles; *gender*, *age*, *ethnic*, *country*, *education level*, and *political orientation*, and one additional context style *time-of-day* (tod). For each type of persona, we provide several categories for annotators to choose. For example, *political orientation* has three categories: Centrist, Left Wing, and Right Wing. Categories in other styles are described in the next sub-section.

In the second step, we ask annotators to produce text that describes the given input of denotation. We again use the pre-processed ViST (Huang et al., 2016) data in §4.4.2 for our input denotations. To reflect annotators' persona, we explicitly ask annotators to reflect their own persona in the stylistic writing, instead of pretending others' persona.

To amortize both costs and annotators' effort at answering questions, each HIT requires the participants to annotate three stories after answering demographic questions. One annotator was paid $0.11 per HIT. For English proficiency, the annotators were restricted to be from USA or UK. A total 501 unique annotators participated in the study. The average number of HIT per annotator was 9.97.

Once we complete our annotations, we filter out noisy responses such as stories with missing images and overtly short sentences (i.e., minimum sentence length is 5). The dataset is then

Table 4.5: Data statistics of the PASTEL.

	Number of Sentences	Number of Stories
Train	33,240	6,648
Valid	4,155	831
Test	4,155	831
total	41,550	8,310

randomly split into train, valid, and test set by 0.8, 0.1, and 0.1 ratios, respectively. Table 4.5 shows the final number of stories and sentences in our dataset.

Reference Sentence: went to an art museum with a group of friends.	
edu:HighSchoolOrNoDiploma	My friends and I went to a art museum yesterday .
edu:Bachelor	I went to the museum with a bunch of friends.

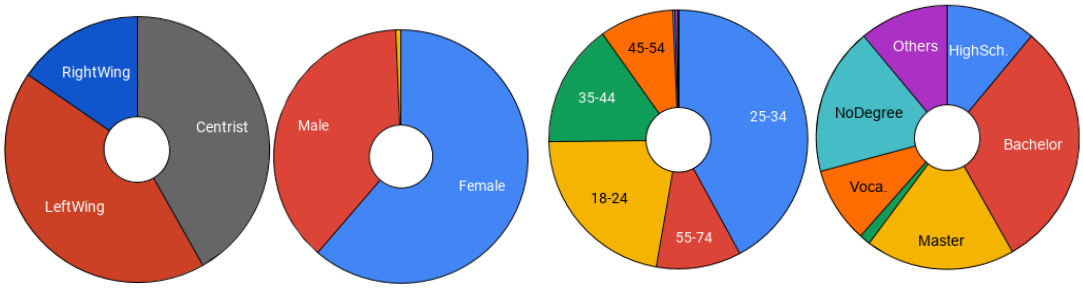
Reference Sentence: the living room of our new home is nice and bright with natural light.	
edu:NoDegree, *gender*:Male	The natural lightning made the apartment look quite nice for the upcoming tour .
edu:Graduate, *gender*:Female	The house tour began in the living room which had a sufficient amount of natural lighting.

Reference Story: Went to an art museum with a group of friends . We were looking for some artwork to purchase, as sometimes artist allow the sales of their items . There were pictures of all sorts , but in front of them were sculptures or arrangements of some sort . Some were far out there or just far fetched . then there were others that were more down to earth and stylish. this set was by far my favorite.very beautiful to me .

edu:HighSchool, *ethnic*:Caucasian, *gender*:Female	My friends and I went to a art museum yesterday . There were lots of puchases and sales of items going on all day . I loved the way the glass sort of brightened the art so much that I got all sorts of excited . After a few we fetched some grub . My favorite set was all the art that was made out of stylish trash .
edu:Bachelor, *ethnic*:Caucasian, *gender*:Female	I went to the museum with a bunch of friends . There was some cool art for sale . We spent a lot of time looking at the sculptures . This was one of my favorite pieces that I saw . We looked at some very stylish pieces of artwork .

Table 4.6: Two sentence-level (top, middle) and one story-level (bottom) annotations in PASTEL. Each text produced by an annotator has their own persona values (underline) for different types of styles (italic). Note that the reference sentence (or story) is given for comparison with the annotated text. Note that misspellings of the text are made by annotators.

Analysis and Examples



(a) Political orientation **(b)** Gender **(c)** Age **(d)** Education level

Figure 4.4: Distribution of annotators for each personal style in PASTEL. Best viewed in color.

Figure 4.4 shows demographic distributions of the annotators. Education-level of annotators is well-balanced, while gender and political view are somewhat biased (e.g., 68% of annotators are Female, only 18.6% represent themselves as right-wing).

Table 4.6 shows few examples randomly chosen from our dataset: two at sentence level (top, middle) and one at story level (bottom). Due to paucity of space, we only show a few types of persona styles. For example, we observe that Education level (e.g., NoDegree vs. Graduate) actually reflects a certain degree of formality in their writing at both sentence and story levels. In §4.4.4, we conduct an in-depth analysis of textual variation with respect to the persona styles in PASTEL.

4.4.4 Applications

PASTEL can be used in many style related applications including style classification, stylometry (Verhoeven and Daelemans, 2014), style transfer (Fu et al., 2018), visually-grounded style transfer, and more. Particularly, we chose two applications, where PASTEL helps design appropriate experiment and evaluation: controlled style classification (§4.4.4) and supervised style transfer (§4.4.4).

Controlled Style Classification

A common mistake in style classification datasets is not controlling external style variables when predicting the category of the target style. For example, when predicting a gender type given a text $P(gender{=}Male|\texttt{text})$, the training data is only labeled by the target style *gender*. However, the `text` is actually produced by a person with not only *gender*=*Male* but also other persona styles such as *age*=55-74 or *education*=HighSchool. Without controlling the other external styles, the classifier is easily biased against the training data.

We define a task called *controlled style classification* where all other style variables are fixed[5], except one to classify. Here we evaluate (1) which style variables are relatively difficult or easy

[5]The distribution of number of training instances per variable is given in Appendix

to predict from the text given, and (2) what types of textual features are salient for each type of style classification.

Features.

Stylistic language has a variety of features at different levels such as lexical choices, syntactic structure and more. Thus, we use following features:

- **lexical** features: ngram's frequency (n=3), number of named entities, number of stop-words
- **syntax** features: sentence length, number of each Part-of-Speech (POS) tag, number of out-of-vocabulary, number of named entities
- **deep** features: pre-trained sentence encoder using BERT (Devlin et al., 2019)
- **semantic** feature: sentiment score

where named entities, POS tags, and sentiment scores are obtained using the off-the-shelf tools such as Spacy[6] library. We use $70K$ n-gram lexical features, 300 dimensional embeddings, and 14 hand-written features.

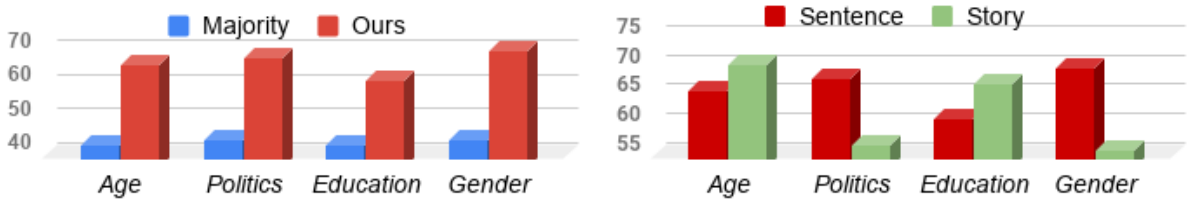
Models.

We train a binary classifier for each personal style with different models: logistic regression, SVM with linear/RBF kernels, Random Forest, Nearest Neighbors, Multi-layer Perceptron, AdaBoost, and Naive Bayes. For each style, we choose the best classifiers on the validation. Their F-scores are reported in Figure 4.5. We use sklearn's implementation of all models (Pedregosa et al., 2011b).[7] We consider various regularization parameters for SVM and logistic regression (e.g., c=[0.01, 0.1, 0.25, 0.5, 0.75, 1.0].

We use neural network based baseline models: deep averaging networks (DAN, Iyyer et al., 2015) of GloVe word embeddings (Pennington et al., 2014)[8]. We also compare with the non-controlled model (Combined) which uses a combined set of samples across all other variables except for one to classify using the same features we used.

Setup.

We tune hyperparameters using 5-fold cross validation. If a style has more than two categories, we choose the most conflicting two: *gender*:{Male, Female}, *age*: {18-24, 35-44}, *education*: {Bachelor, No Degree}, and *politics*: {LeftWing, RightWing}. To classify one style, all possible combinations of other styles ($2 * 2 * 2$=8) are separately trained by different models. We use the macro-averaged F-scores among the separately trained models on the same test set for every models.



(a) different style types **(b)** sentences vs stories

Figure 4.5: Controlled style classification: F-scores on (a) different types of styles on sentences and on (b) our best models between sentences and stories. Best viewed in color.

Results.

Figure 4.5 shows F-scores (a) among different styles and (b) between sentences and stories. In most cases, multilayer perceptron (MLP) outperforms the majority classifier and other models by large margins. Compared to the neural baselines and the combined classifier, our models show better performance. In comparison between controlled and combined settings, controlled setting achieves higher improvements, indicating that fixing external variables helps control irrelevant features that come from other variables. Among different styles, gender is easier to predict from the text than ages or education levels. Interestingly, a longer context (i.e., story) is helpful in predicting age or education, whereas not for political view and gender.

In our ablation test among the feature types, the combination of different features (e.g., lexical, syntax, deep, semantic) is very complementary and effective. Lexical and deep features are two most significant features across all style classifiers, while syntactic features are not.

Table 4.7 shows the most salient features for classification of each style. Since we can't interpret deep features, we only show lexical and syntactic features. The salience of features are ranked by coefficients of a logistic regression classifier. Interestingly, female annotators likely write more nouns and lexicons like 'happy', while male annotators likely use pronouns, adjectives, and named entities. Annotators on left wing prefer to use 'female', nouns and adposition, while annotators on right wing prefer shorter sentences and negative verbs like 'n't'. Not many syntactic features are observed from annotators without degrees compared to with bachelor degree.

Supervised Style Transfer

The style transfer is defined as $(S, \alpha) \rightarrow \hat{S}$: We attempt to alter a given source sentence S to a given target style α. The model generates a candidate target sentence $\hat{S}$ which preserves the meaning of S but is more faithful to the target style α so being similar to the target annotated sentence $\bar{S}_\alpha$. We evaluate the model by comparing the predicted sentence $\hat{S}$ and target annotated sentence $\bar{S}_\alpha$. The sources are from the original reference sentences, while the targets are from

[6] `https://spacy.io/`

[7] `http://scikit-learn.org/stable/`

[8] Other architectures such as convolutional neural networks (CNN, Zhang et al., 2015) and recurrent neural networks (LSTM, Hochreiter and Schmidhuber, 1997b) show comparable performance as DAN.

Table 4.7: Most salient lexical (lower cased) and syntactic (upper cased) features on story-level classification. Each feature is chosen by the highest coefficients in the logistic regression classifier.

Gender:Male	*Gender*:Female
PROPN, ADJ, #_ENTITY, went, party, SENT_LEN	happy, day, end, group, just, snow, NOUN
Politics:LeftWing	*Politics*:RightWing
female, time, NOUN, ADP, VERB, porch, day, loved	SENT_LENGTH, PROPN, #_ENTITY, n't, ADJ, NUM
Education:Bachelor	*Education*:NoDegree
food, went, #_STOPWORDS, race, ADP	!, just, came, love, lots, male, fun, n't, friends, happy
Age:18-24	*Age*:35-44
ADP, come, PROPN, day, ride, playing, sunset	ADV, did, town, went, NOUN, #_STOPWORDS

our annotations.

Models.

We compare five different models:

- AsItIs: copies over the source sentence to the target, without any alterations.
- WordDistRetrieve: retrieves a training source-target pair that has the same target style as the test pair and is closest to the test source in terms of word edit distance (Navarro, 2001). It then returns the target of that pair.
- EmbDistRetrieve: Similar to WordDistRetrieve, except that a continuous bag-of-words (CBOW) is used to retrieve closest source sentence instead of edit distance.
- Unsupervised: use unsupervised style transfer models using Variational Autoencoder (Shen et al., 2017) and using additional objectives such as cross-domain and adversarial losses (Lample et al., 2017)[9]. Since unsupervised models can't train multiple styles at the same time, we train separate models for each style and macro-average their scores at the end. In order not to use the parallel text in PASTEL, we shuffle the training text of each style.
- Supervised: uses a simple attentional sequence-to-sequence (S2S) model (Bahdanau et al., 2014) extracting the parallel text from PASTEL. The model jointly trains different styles in conjunction by concatenating them to the source sentence at the beginning.

We avoid more complex architectural choices for Supervised models like adding a pointer component or an adversarial loss, since we seek to establish a minimum level of performance on this dataset.

[9]We can't directly compare with Hu et al. (2017); Prabhumoye et al. (2018) since their performance highly depends on the pre-trained classifier that often shows poor performance.

Setup.

We experiment with both SOFTMAX and SIGMOID non-linearities to normalize attention scores in the sequence-to-sequence attention. Adam (Kingma and Ba, 2014) is used as the optimizer. Word-level cross entropy of the target is used as the loss. The batch size is set to 32. We pick the model with lowest validation loss after 15 training epochs. All models are implemented in PyTorch (Paszke et al., 2017).

For an evaluation, in addition to the same hard and soft metrics used for measuring the meaning preservation in §4.4.2, we also use BLEU_2 (Papineni et al., 2002) for unigrams and bigrams, and ROUGE (Lin and Hovy, 2003) for hard metric and Embedding Averaging (EA) similarity (Liu et al., 2016) for soft metric.

Table 4.8: Supervised style transfer. GLOVE initializes with pre-trained word embeddings. PRETR. denotes pre-training on YAFC. Hard measures are **B**LEU$_2$, **M**ETEOR, and **R**OUGE, and soft measures are **E**mbeding**A**veraging and **V**ector**E**xtrema.

	Hard $(\hat{S}, \bar{S}_\alpha)$			Soft $(\hat{S}, \bar{S}_\alpha)$	
Models: $(S, \alpha) \rightarrow \hat{S}$	$\mathbf{B}_2$	**M**	**R**	**EA**	**VE**
ASITIS	**35.41**	**12.38**	21.08	0.649	0.393
WORDDISTRETRIEVE	30.64	7.27	22.52	0.771	0.433
EMBDISTRETRIEVE	33.00	8.29	24.11	0.792	0.461
UNSUPERVISED					
· Shen et al. (2017)	23.78	7.23	21.22	0.795	0.353
· Lample et al. (2017)	24.52	6.27	19.79	0.702	0.369
SUPERVISED					
· S2S	26.78	7.36	25.57	0.773	0.455
· S2S+GLOVE	31.80	10.18	<u>29.18</u>	<u>0.797</u>	<u>0.524</u>
· S2S+GLOVE+PRETR.	<u>31.21</u>	<u>10.29</u>	**29.52**	**0.804**	**0.529**

Results.

Table 4.8 shows our results on style tranfer. We observe that initializing both en/decoder's word embeddings with GLOVE (Pennington et al., 2014) improves model performance on most metrics. Pretraining (PRETR.) on the formality style transfer data YAFC (Rao and Tetreault, 2018) further helps performance. All supervised S2S approaches outperform both retrieval-based baselines on all measures. This illustrates that the performance scores achieved are not simply a result of memorizing the training set. S2S methods surpass ASITIS on both soft measures and ROUGE. The significant gap that remains on BLEU remains a point of exploration for future work. The significant improvement against the unsupervised methods (Shen et al., 2017; Lample et al., 2017) indicates the usefulness of the parallel text in PASTEL.

Table 4.9 shows output text $\hat{S}$ produced by our model given a source text S and a style α. We observe that the output text changes according to the set of styles.

Source (S): I'd never seen so many beautiful flowers.
Style (α): (Morning, HighSchool) $S + \alpha \rightarrow \hat{S}$: the beautiful flowers were beautiful. $\bar{S}_\alpha$: the flowers were in full bloom. **Style** (α): (Afternoon, NoDegree) $S + \alpha \rightarrow \hat{S}$: The flowers were very beautiful. $\bar{S}_\alpha$: Tulips are one of the magnificent varieties of flowers.
Source (S): she changed dresses for the reception and shared food with her new husband.
Style (α): (Master, Centrist) $S + \alpha \rightarrow \hat{S}$: The woman had a great time with her husband $\bar{S}_\alpha$: Her husband shared a cake with her during reception
Style (α): (Vocational, Right) $S + \alpha \rightarrow \hat{S}$: The food is ready for the reception $\bar{S}_\alpha$: The new husband shared the cake at the reception

Table 4.9: Examples of style transferred text by our supervised model (S2S+GloVe+PreTr.) on PASTEL. Given source text (S) and style (α), the model predicts a target sentence $\hat{S}$ compared to annotated target sentence $\bar{S}_\alpha$.

Additional experiment: Proof of stylistic transfer claim

In order to precisely show how styles are transferred between two texts, we propose a claim for stylistic transfer and provide empirical proof on our current transfer model.

The style transfer model takes an input text S and style factor α, then generate style-transferred text S^α:

$$T(S, \alpha^+) \longrightarrow S^{\alpha^+} \tag{4.1}$$

$$T(S, \alpha^-) \longrightarrow S^{\alpha^-} \tag{4.2}$$

where T is the style transfer model trained, α^+ and α^- are style factors with two opposing values; positive (+) and negative (−). For example, in sentiment style, the style factors can be positive and negative sentiment, while in gender style, the style factors can be male and female.

Since our dataset has the reference text with the style factor given; $\hat{S}^{\alpha^+}$ and $\hat{S}^{\alpha^-}$, we can evaluate their similarities.

$$F^m(S^{\alpha^+}, \hat{S}^{\alpha^+}) \quad , \quad F^m(S^{\alpha^-}, \hat{S}^{\alpha^-}) \tag{4.3}$$

where m is an evaluation metric such as BLEU, METEOR, or VectorExtrema.

Then, we propose a claim for proof of stylistic transfer of the model T as follows:

Theorem 1 *If for N sentences S and style factors α with values + (positive) and − (negative), it is true that*

$$F^m(S^{\alpha^+}, \hat{S}^{\alpha^+}) > F^m(S^{\alpha^-}, \hat{S}^{\alpha^+})$$

and

$$F^m(S^{\alpha^-}, \hat{S}^{\alpha^-}) > F^m(S^{\alpha^+}, \hat{S}^{\alpha^-})$$

then it is demonstrated that the transfer model T is able to reliably convert each sentence S into the appropriate style $+$ and $-$ for α.

	Gender	**Political view**	**Education level**	**Ethnicity**
+/-	Male, Female	Left-wing, Right-wing	No-degree, Bachelor	Caucasian, Hispanic
#	191	63	47	18
*m*etric used in F	METEOR / VE	METEOR / VE	METEOR / VE	METEOR / VE
$\Delta^+ = F(S^{\alpha^+}, \hat{S}^{\alpha^+}) - F(S^{\alpha^-}, \hat{S}^{\alpha^+})$	0.22 / 0.7	0.55 / 0.22	0.12 / 0.55	-0.32 / -2.27
$\Delta^- = F(S^{\alpha^-}, \hat{S}^{\alpha^-}) - F(S^{\alpha^+}, \hat{S}^{\alpha^-})$	-0.02 / -0.76	0.63 / 0.88	0.15 / 0.46	1.32 / 1.9

Table 4.10: Stylistic transfer inequalities in Eq 1 in PASTEL. # means the number of aligned test pairs with positive and negative style factors used in our experiment. Blue -shaded numbers show **valid** inequalities ($\Delta^+ > 0$, $\Delta^- > 0$), while red -shared numbers show **invalid** inequalities ($\Delta^+ < 0$, $\Delta^- < 0$).

Table 4.10 shows the inequalities over the four styles with the number of aligned pairs used in the experiment[10]. We check the inequalities in Eq 1 for four style factors in PASTEL: **gender** for male and female, **political view** for left wing and right wing, **education level** for no-degree and bachelor, and **ethnicity** for Caucasian and Hispanic[11].

Some styles, such as education level and political view, show stable increases of transfer similarity between the positive output and positive reference (or negative output and negative reference), compared to the negative output and positive reference (or vice versa), and prove our contention. Unfortunately, this is not true for all styles. Transferring to female in gender and to Caucasian in ethnicity does not reliably convert style toward the appropriate reference text: for ethnicity the problem occurs toward Caucasian and for gender the problem occurs toward female, though the effect (considering just METEOR) is still in the appropriate direction. Why exactly is unclear; it may be that the dataset does not strongly enough encode Caucasian and female stylistic features, or simply the the model is inadequately trained. Extending this experiment on other style transfer models or other style-parallel datasets might be an interesting direction for future work.

4.4.5 Conclusion

We present PASTEL, a parallelly annotated stylistic language dataset. Our dataset is collected by human annotation using a proper denotation setting that preserves the meaning as well as maximizes the diversity of styles. Multiplicity of persona styles in PASTEL makes other style variables controlled (or fixed) except the target style for classification, which is a more accurate experimental design. Our simple supervised models with our parallel text in PASTEL outperforms the unsupervised style transfer models using non-parallel corpus. We hope PASTEL can

[10]The pairs are extracted from the test set only.

[11]We choose the two most frequent style values if there are multiple values.

be a useful benchmark to both train and evaluate models for style transfer and other related problems in text generation field.

4.5 Cross-style Language Understanding

4.5.1 Introduction

Every text is written in some style. People often use style as a strategic choice for their personal or social goals. The strategic use of style is mainly because style often conveys more information (e.g., respect) than is contained in the literal meaning of the text (Hovy, 1987).

Imagine an orchestra performed by a large group of instrumental ensemble. What we only hear at the end is the harmonized sound of complex interacting combinations of individual instruments, where the conductor controls their combinatory choices (e.g., score, tempo, correctness) on top of it. Some instruments are in the same category such as violin and cello for bowed string type, and horn and trumpet for brass type. Similarly, text is an output that reflects a complex combination of different types of styles (e.g., metaphor, formality markers) where each has its own lexical choices and some are dependent on each other. The consistent combination of the choices by the speaker (like a conductor in an orchestra) will produce stylistically appropriate text.

Some stylistic choices implicitly reflect the author's characteristics (e.g., personality, demographic traits (Kang et al., 2019b), emotion (Rashkin et al., 2019)), while others are explicitly controlled by the author's choices for some social goals (e.g., polite language for better relationship with the elder (Hovy, 1987), humorous language for smoother communication (Dobrovol'skij and Piirainen, 2005; Glucksberg et al., 2001; Loenneker-Rodman and Narayanan, 2010)). Broadly, we call each individual as one specific type of *style* of language. Each style is then compressed into a single numerical variable (e.g., positive for 1 and negative for 0 in sentiment) to measure the amount of style contained in the text.

However, style is not a single variable, but a combination of multiple variables co-vary in complex ways. Only a few works studied dependency between styles but on the particular group of styles such as demographics (Preoţiuc-Pietro and Ungar, 2018), emotions (Warriner et al., 2013), or between metaphor and emotion (Dankers et al., 2019; Mohammad et al., 2016). This work focuses on a general understanding of *cross-style language*; how different styles (e.g., formality, politeness) co-vary together in text, which styles are dependent on each other, and how they are systematically composed to produce the final text.

To accelerate more research along this line, we present a benchmark (**x**SLUE) for understanding and evaluating cross-style language which includes following features:

- collect a benchmark of 15 style types (e.g., formality, emotion, humor, politeness, sarcasm, offense, romance, personal traits) and 23 classification tasks, and categorize them into four groups (Table 4.11): figurative, affective, personal and interpersonal styles.
- build an online platform (`http://anonymized`) for comparing systems and easily downloading the dataset with the fine-tuned BERT classifier (Devlin et al., 2019).
- collect an extra diagnostic set (i.e., 400 samples of text) which has labels of multiple styles in conjunction annotated by human workers in order to investigate cross-style behavior of the

Figurative styles
HUMOR (ShortHumor, ShortJoke), SARCASM (SarcGhosh, SARC), METAPHOR (TroFi, VUA)
Affective styles
EMOTION (EmoBank, DailyDialog, CrowdFlower), OFFENSE (HateOffensive), ROMANCE (ShortRomance), SENTIMENT (SentiTreeBank)
Personal styles
AGE (PASTEL), ETHNICITY (PASTEL), GENDER (PASTEL), EDUCATION LEVEL (PASTEL), POLITICAL VIEW (PASTEL)
Interpersonal styles
FORMALITY (GYAFC), POLITENESS (StanfordPolite)

Table 4.11: Our categorization of styles with their benchmark dataset (under parenthesis) used in **x**SLUE.

model.

- provide interesting analyses on cross-style language: correlations between two styles (e.g., impoliteness is related to offense) and a comparison of style diversity of text in different domains (e.g., academic papers are stylistically less diverse than tweets).

4.5.2 xSLUE: A Benchmark for Cross-Style Language Understanding

The term *style* is often used in a mixed manner, but no one actually defines a formal categorization of style types and their dependencies. we survey the recent works which described their work as a style language, and then collect the 14 widely-used types of style language: *emotion*, *sentiment*, *metaphor*, *humor*, *sarcasm*, *offensiveness*, *romance*, *formality*, *politeness*, *age*, *ethnicity*, *gender*, *political orientation*, and *education level*. We then categorize them into four groups (Table 4.11): figurative, affective, personal and interpersonal styles. Please find the Appendix for detailed metrics used in our categorization.

xSLUE includes one to three datasets for each style language and a diagnostic set of 400 samples for further understanding of cross-style language.

Dataset for Single Style Language

We choose existing datasets of each style language or collect our own if there is no dataset available. Table 4.12 summarizes the style types, datasets, and data statistics. Due to the label imbalance of some datasets, we measure f-score besides accuracy for classification tasks. We measure Pearson-Spearman correlation for regression tasks. For multi-labels, all scores are macro-averaged. We include the detailed rules of thumbs in our dataset selection in the Appendix. Here we describe the datasets used in our benchmark with the pre-processing procedures.

Formality.

Appropriately choosing the right formality in the situation (e.g., a person to talk to) is the key aspect for effective communication (Heylighen and Dewaele, 1999). We use GYAFC dataset (Rao and Tetreault, 2018) which includes both formal and informal text collected from the web. However, the dataset requires individual authorization from the authors, so we only include a script for preprocessing it to make the same format as other datasets.

Humor.

Humor (or joke) is a social style to make the conversation more smooth or make a break. Detecting humor (Rodrigo and de Oliveira, 2015; Yang et al., 2015; Chandrasekaran et al., 2016) and entendre (Kiddon and Brun, 2011) has been broadly studied using various linguistic features. We use the two well-known datasets used in humor detection: ShortHumor (CrowdTruth, 2016) which contains 22K humorous sentences collected from several websites and ShortJoke (Moudgil, 2017) which contains 231K jokes scraped from several websites[12]. We randomly sample negative (i.e., non-humorous) sentences from the two sources: random text from Reddit summarization corpus (Kang et al., 2019d) and literal text from Reddit corpus (Khodak et al., 2017).

Politeness.

Encoding (im)politeness in conversation often plays different roles of social interactions such as for power dynamics at workplaces, decisive factor, and strategic use of it in social context (Chilton, 1990; Holmes and Stubbe, 2015; Clark and Schunk, 1980). We use Stanford's politeness dataset StanfPolite (Danescu et al., 2013) which collects request types of polite and impolite text from the web such as Stack Exchange question-answer community.

Sarcasm.

Sarcasm acts by using words that mean something other than what you want to say, to insult someone, show irritation, or simply be funny. Therefore, it is often used interchangeably with irony. The figurative nature of sarcasm leads to more challenges to identify it in text (Tepperman et al., 2006; Wallace et al., 2014; Wallace, 2015; Joshi et al., 2017). Sarcasm datasets are collected from different domains: books (Joshi et al., 2016), tweets (González-Ibánez et al., 2011; Peled and Reichart, 2017; Ghosh and Veale, 2016), reviews (Filatova, 2012), forums (Walker et al., 2012), and Reddit posts (Khodak et al., 2017). We choose two of them for **x**SLUE: SarcGhosh (Ghosh and Veale, 2016) and SARC v2.0 (Khodak et al., 2017)[13]. For SARC, we use the same preprocessing scheme in Ilić et al. (2018).

[12]We do not use other joke datasets (Pungas, 2017; Potash et al., 2017; Rodrigo and de Oliveira, 2015; Mihalcea and Strapparava, 2006), because of the limited domain or low recall issue.

[13]SARC_{pol} is a sub-task for the text from politics subreddit.

Metaphor.

Metaphor is a figurative language that describes an object or an action by applying it to which is not applicable. Detecting metaphoric text has been studied in different ways: rule-based (Russell, 1976; Martin, 1992), dictionary-based, and recently more computation-based with different factors (e.g., discourse, emotion) (Nissim and Markert, 2003; Jang et al., 2017; Mohler et al., 2013). We use two benchmark datasets[14]: Trope Finder (`TroFi`) (Birke and Sarkar, 2006) and VU Amsterdam `VUA` Corpus (Steen, 2010) where metaphoric text is annotated by human annotators.

Offense.

Hate speech is a speech that targets disadvantaged social groups based on group characteristics (e.g., race, gender, sexual orientation) in a manner that is potentially harmful to them (Jacobs et al., 1998; Walker, 1994). We use the `HateOffenssive` dataset (Davidson et al., 2017) which includes hate text (7%), offensive text (76%), and none of them (17%).

Romance.

To the best of our survey, we could not find any dataset which includes romantic text. Thus, we crawl romantic text from eleven different web sites (See Appendix), pre-process them by filtering out some noisy, too-long, and duplicate text, and then make a new dataset called `ShortRomance`. Similar to the humor datasets, we make the same number of negative samples from the literal Reddit sentences (Khodak et al., 2017) as the romantic text.

Sentiment.

Identifying sentiment polarity of opinion is challenging because of its implicit and explicit presence in text (Kim and Hovy, 2004; Pang et al., 2008). We use the well-known, large scale of annotated sentiment corpus on movie reviews; Sentiment Tree Bank (Socher et al., 2013) (`SentiBank`).

Emotion.

Emotion is more fine-grained modeling of sentiment. It can be either *categorical* or *dimensional*: Ekman (1992) categorized six discrete types of emotion states: anger, joy, surprise, disgust, fear, and sadness, while Warriner et al. (2013) described the states as independent dimensions called VAD model: Valence (polarity), Arousal (calmness or excitement), and Dominance (degree of control). We use two datasets: `DailyDialog` (Li et al., 2017) from the Ekman's categorical model and `EmoBank` (Buechel and Hahn, 2017) from the VAD's model[15]. We also include a large but noisy emotion-annotated corpus `CrowdFlower` (CrowdFlower, 2016), which contains Ekman's categories as well as additional 7 categories: enthusiasm, worry, love, fun, hate, relief, and boredom.

[14]we did not include Mohler et al. (2016)'s dataset because the labels are not obtained from human annotators.

[15]The range for original `EmoBank` was $[0, 5]$ but we normalize it in $[0, 1]$ in our benchmark.

Persona Styles: age, gender, political view, ethnicity, country, and education.

Persona is a pragmatics style in group characteristics of the speaker (Kang et al., 2019b). Certain groups of persona use specific textual features of language styles. We use the stylistic language dataset written in parallel called PASTEL (Kang et al., 2019b) where multiple types of the author's personas are given in conjunction. Similar to the emotion datasets, PASTEL has six different persona styles (i.e., age, gender, political view, ethnicity, country, education) where each has multiple attributes.

Style Type & Dataset	#S	Split	#L	Label (proportion)	B	Domain	Public	Task
Formality								
GYAFC (Rao and Tetreault, 2018)	224k	given	2	formal (50%), informal (50%)	Y	web	N	clsf.
Politeness								
StanfPolite (Danescu et al., 2013)	10k	given	2	polite (49.6%), impolite (50.3%)	Y	web	Y	clsf.
Humor								
ShortHumor (CrowdTruth, 2016)	44k	random	2	humor (50%), non-humor (50%)	Y	web	Y	clsf.
ShortJoke (Moudgil, 2017)	463k	random	2	humor (50%), non-humor (50%)	Y	web	Y	clsf.
Sarcasm								
SarcGhosh (Ghosh and Veale, 2016)	43k	given	2	sarcastic (45%), non-sarcastic (55%)	Y	tweet	Y	clsf.
SARC (Khodak et al., 2017)	321k	given	2	sarcastic (50%), non-sarcastic (50%)	Y	reddit	Y	clsf.
SARC_pol (Khodak et al., 2017)	17k	given	2	sarcastic (50%), non-sarcastic (50%)	Y	reddit	Y	clsf.
Metaphor								
VUA (Steen, 2010)	23k	given	2	metaphor (28.3%), non-metaphor (71.6%)	N	misc.	Y	clsf.
TroFi (Birke and Sarkar, 2006)	3k	random	2	metaphor (43.5%), non-metaphor (54.5%)	N	news	Y	clsf.
Emotion								
EmoBank$_{valence}$ (Buechel and Hahn, 2017)	10k	random	1	negative, positive	-	misc.	Y	rgrs.
EmoBank$_{arousal}$ (Buechel and Hahn, 2017)	10k	random	1	calm, excited	-	misc.	Y	rgrs.
EmoBank$_{dominance}$ (Buechel and Hahn, 2017)	10k	random	1	being_controlled, being_in_control	-	misc.	Y	rgrs.
CrowdFlower (CrowdFlower, 2016)	40k	random	14	neutral (21%), worry (21%), happy (13%)..	N	tweet	Y	clsf.
DailyDialog (Li et al., 2017)	102k	given	7	noemotion(83%),happy(12%),surprise(1%)..	N	dialogue	Y	clsf.
Offense								
HateOffensive (Davidson et al., 2017)	24k	given	3	hate(6.8%), offensive(76.3%), neither(16.8%)	N	tweet	Y	clsf.
Romance								
ShortRomance	2k	random	2	romantic (50%), non-romantic (50%)	Y	web	Y	clsf.
Sentiment								
SentiBank (Socher et al., 2013)	239k	given	2	positive (54.6%), negative (45.4%)	Y	web	Y	clsf.
Persona								
PASTEL_gender (Kang et al., 2019b)	41k	given	3	Female (61.2%), Male (38.0%), Others (.6%)	N	caption	Y	clsf.
PASTEL_age (Kang et al., 2019b)	41k	given	8	35-44 (15.3%), 25-34 (42.1%), 18-24 (21.9%)..	N	caption	Y	clsf.
PASTEL_country (Kang et al., 2019b)	41k	given	2	USA (97.9%), UK (2.1%)	N	caption	Y	clsf.
PASTEL_politics (Kang et al., 2019b)	41k	given	3	Left (42.7%), Center (41.7%), Right (15.5%)	N	caption	Y	clsf.
PASTEL_education (Kang et al., 2019b)	41k	given	10	Bachelor(30.6%),Master(18.4%),NoDegree(18.2	N	caption	Y	clsf.
PASTEL_ethnicity (Kang et al., 2019b)	41k	given	10	Caucasian(75.6%),NativeAmerican(8.6%),Africa	N	caption	Y	clsf.

Table 4.12: Style datasets in **x**SLUE. Every label ranges in [0, 1]. #S and #L mean the number of total samples and labels, respectively. **B** means whether the labels are balanced or not. '_' in the dataset means its sub-task. **Public** means whether the dataset is publicly available or not: GYAFC needs special permission from the authors. *clsf.* and *rgrs.* denotes classification and regression, respectively. We use accuracy and f1 measures for classification, and Pearson-Spearman correlation for regression.

Diagnostic Set for Cross-Style Language

With a single type of style language, we train an individual classifier on each style type and measure its performance on the independent test set. Without a shared test set across different styles, however, we can not measure how different styles are identified at the same time (i.e., *cross-style classification*) and whether the model captures the underlying structure of an inter-style variation of text. In order to help researchers test their models in the cross-style setting, we collect a diagnostic set by annotating appropriate labels of multiple styles (i.e., total 15 style types) at the same time from crowd workers. Our diagnostic set has a total 400 samples: 200 from the *cross-test* set and 200 from the *tweet-diverse* set:

- *cross-test* set is 200 samples randomly chosen from the test set among the 15 datasets in balance.
- *tweet-diverse* set is another 200 samples chosen from random tweets. We collect 100 tweets containing the high stylistic diversity and another 100 tweets containing less stylistic diversity (See §4.5.4 for our measurement of style diversity).

We ask human workers to predict the stylistic attribute of the text for 15 different style types, so making them appropriately adjusting the multi-style attributes at the same time. For the confidence of annotation, each sample is annotated by three different workers. The final label for each style is decided via the majority voting over the three annotators. For personal styles (e.g., age, gender), we also add Don't Know option to choose in case that its prediction is too difficult. In case three votes are all different from each other, we did not use the sample in our evaluation. We will be releasing these ambiguous or controversy cases including the Don't Know answer as a separate evaluation set in the future. The detailed instructions and annotation schemes are in the Appendix.

4.5.3 Single and Cross Style Classification

Setup.

In a single-style classification, we individually train a classifier (or regression for `EmoBank`) on each dataset and predict the label. We use the state-of-the-art classifier; the fine-tuning with the (uncased) pre-trained language model; Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al., 2019)[16]. For evaluation, we report both accuracy and f1-score by macro-averaging due to the label imbalance.

As a baseline, we provide a simple majority classifier (i.e., taking the majority label from the training set as prediction). Besides, we compare another baseline using Bidirectional LSTM (BiLSTM) (Hochreiter and Schmidhuber, 1997a) initialized with GLoVe word embeddings (Pennington et al., 2014). We report model performances from the original paper if given. If the experimental setup from the original paper is not directly applicable (e.g., the difference in evaluation metrics), we mark them as *na*. The details of the hyper-parameters are in the Appendix. All experimental code and data are publicly available upon acceptance[17].

[16]Other variants of the BERT models such as 'large-uncased' showed comparable performance.
[17]`http://annonymized`

		Single-style classification				Cross-style classification	
						cross-test	*tweet-diverse*
STYLE	**Dataset**	**Majority**	**Original**	**BiLSTM**	**BERT**	**BERT**	**BERT**
FORMALITY	GYAFC	43.3 (30.2)	*na*	76.5 (76.4)	**88.3** (**88.3**)	62.8 (62.6)	78.5 (57.5)
POLITENESS	StanfPolite	56.7 (36.2)	**83.2**	62.1 (61.8)	66.8 (**65.8**)	78.7 (78.7)	90.0 (47.3)
HUMOR	ShortHumor	50.0 (33.3)	*na*	88.6 (88.6)	**97.0** (**97.0**)	-	-
	ShortJoke	50.0 (33.3)	*na*	89.1 (89.1)	**98.3** (**98.3**)	57.3 (46.6)	60.8 (52.5)
SARCASM	SarcGhosh	50.0 (33.3)	*na*	**73.0** (**72.6**)	54.4 (42.4)	-	-
	SARC	50.0 (33.3)	**75.8**	63.0 (63.0)	70.2 (**70.1**)	61.9 (49.6)	51.7 (41.6)
	SARC_pol	50.0 (33.3)	**76.0**	61.3 (61.3)	71.8 (**71.7**)	-	-
METAPHOR	VUA	70.0 (41.1)	*na*	77.1 (68.9)	**84.5** (**89.1**)	28.5 (22.2)	33.3 (25.0)
	TroFi	57.2 (36.4)	46.3	74.5 (73.9)	**75.7** (**78.9**)	-	-
EMOTION	$\text{EmoBank}_{V/A/D}$	-/-/-	*na*	78.5/49.4/39.5	**81.2**/**58.7**/**43.6**	66.4/26.1/82.9	77.8/31.6/81.9
	CrowdFlower	22.4 (2.8)	*na*	31.1 (12.3)	**36.5** (**21.9**)	-	-
	DailyDialog	81.6 (12.8)	*na*	**84.2** (27.61)	**84.2** (**49.6**)	50.0 (20.4)	65.8 (21.4)
OFFENSE	HateOffens	75.0 (28.5)	91.0	86.6 (68.2)	**96.6** (**93.4**)	84.0 (50.1)	81.4 (33.6)
ROMACE	ShortRomance	50.0 (33.3)	*na*	90.6 (90.6)	**99.0** (**98.9**)	93.1 (75.0)	73.3 (55.5)
SENTIMENT	SentiBank	50.0 (33.3)	87.6	82.8 (82.8)	**96.6** (**96.6**)	89.8 (89.4)	88.1 (77.2)
PERSONA	PASTEL_gender	62.8 (25.7)	*na*	**73.2** (45.5)	73.0 (**48.7**)	40.4 (21.1)	44.1 (29.6)
	PASTEL_age	41.5 (7.3)	*na*	41.9 (15.2)	**46.3** (**23.9**)	40.0 (25.5)	60.2 (39.1)
	PASTEL_country	97.2 (49.2)	*na*	**97.2** (49.3)	97.1 (**55.2**)	94.9 (48.7)	97.2 (49.3)
	PASTEL_politics	42.9 (20.0)	*na*	48.5 (33.5)	**50.9** (**46.1**)	11.7 (9.5)	36.3 (29.6)
	PASTEL_education	31.4 (4.7)	*na*	42.4 (15.0)	**42.5** (**25.4**)	24.6 (12.4)	24.0 (11.5)
	PASTEL_ethnicity	75.4 (8.5)	*na*	**82.3** (17.6)	81.1 (**25.6**)	63.3 (20.4)	32.5 (16.3)
	total	55.4(26.8)		69.3(55.7)	73.7(64.3)	58.7 (41.8)	61.6 (39.5)

Table 4.13: Single and cross style classification. We use accuracy and macro-averaged f1-score (under parenthesis) for classification tasks. *na* means not applicable. For cross-style classification, we choose a classifier train on one dataset per style, which has larger training data.

Results.

Table 4.13 shows performance on single-style classification (left) and cross-style classification (right). The fine-tuned BERT classifier outperforms the majority and BiLSTM baselines on the f1 score by the large margins except for SarcGhosh. Especially, BERT shows significant f1 improvements on humor and personal styles. For sarcasm and politeness tasks, our classifiers do not outperform the performance in the original papers, which use additional hand-written syntactic features.

When classifying multiple styles at the same time, single-style classifiers do not show comparable performance as done in a single-style classification. This is mainly because the single-style classifier trained on a specific domain of dataset is biased to the domain and the dataset itself may include some annotation artifacts which are not scalable to the held-out samples. More importantly, there is a fundamental difference between the cross-style and the single-style classification: when predicting multiple styles together, one may consider how different styles are dependent on each other, indicating the necessity of a unified model where multiple styles are jointly trained and their underlying dependency structures across multi-styles are modeled. In

the same manner, generating multi-style text can be also applicable as if the harmonized sound of complex combinations across individual instruments.

4.5.4 Cross-Style Language Understanding

We provide interesting analyses of cross-style language using **x**SLUE: correlation analysis across style types and stylistic diversity of text w.r.t domains.

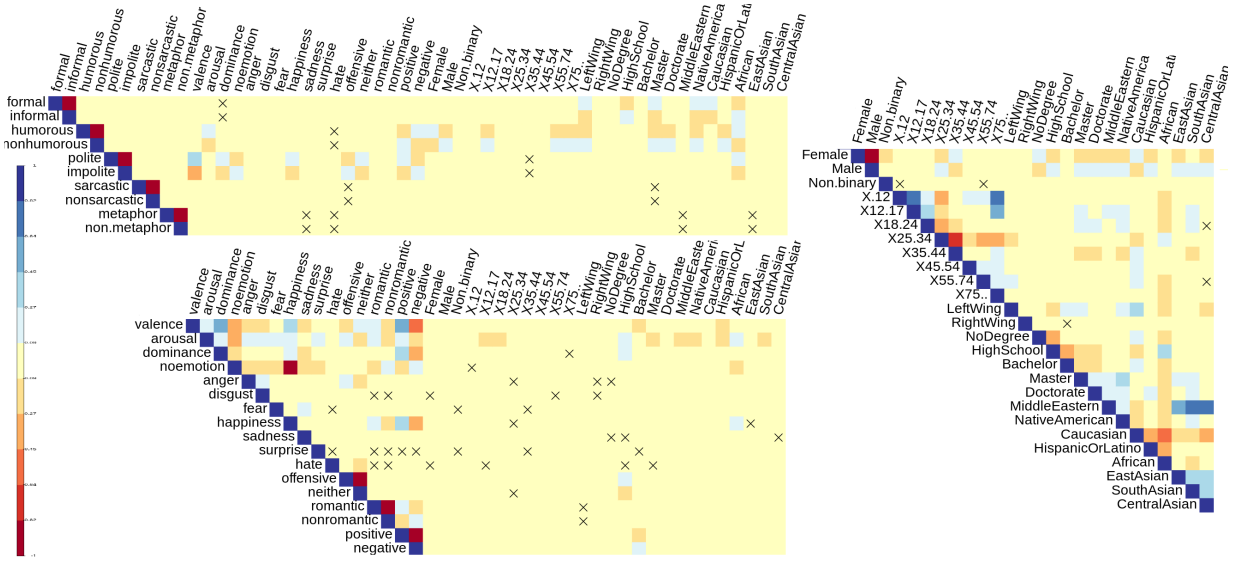


Figure 4.6: Cross-style correlation. The degree of correlation gradually increases from red (negative) to blue (positive), where the color intensity is proportional to the correlation coefficients. Correlations with $p < 0.05$ (confidence interval: 0.95) are only considered as statistically significant. Otherwise, crossed. **NOTE: Please be careful not to make any unethical or misleading claims based on these results which include potential weakness (see text below).** Best viewed in color.

Cross-Style Correlation

Can we measure how different styles are independent on each other?

Setup. We first sample 1,000,000 tweets from the CSpike dataset (Kang et al., 2017b) crawled using Twitter's Gardenhose API. We choose tweets as a target domain of our analysis due to its stylistic diversity compared to other domains such as news articles (See §4.5.4 for stylistic diversity across domains). Using the fine-tuned style classifiers in §4.5.3, we predict the probability of 53 style attributes over the 1M tweets. We then produce a correlation matrix across them using Pearson correlation coefficients with Euclidean distance measure and finally output the 53×53 correlation matrix (Figure 4.6): we split it into three pieces based on the sub-categories in Table 4.11: interpersonal and figurative (top, left), affective (bottom, left), and personal (right) styles.

We assume that certain textual features (e.g., lexical choices) which could be detected by the classifiers, co-occur across multiple styles, giving the co-occurrence of the predicted probabilities. Compared to the prior analyses (Hovy, 1987; Preoţiuc-Pietro and Ungar, 2018; Kang et al., 2019b) based on specific lexical features, our analysis uses classification predictions and their co-occurrence patterns.

Analysis and Weakness. We observe some reasonable positive correlations: (humorous text, negative text), (non-humorous text, text by Master / Doctorate education), (polite text, text with no-emotion), (text with dominance in control, positive text), (text with anger emotion, offensive text), and more.

However, we should not blindly trust the correlations. For example, there are unreasonable cases like the positive correlation between Age(<12) and Age(>=75), which is not expected. More than that, we should be VERY CAREFUL not to make any misleading interpretation based on them, especially for some styles related to personal traits. This is not only due to the ethical issues but the weakness of our experimental design:

- Our correlation analysis is not causal. In order to find causal relation between styles, more sophisticated causal analyses (e.g., analysis of covariance (ANCOVA) (Keppel, 1991) or propensity score (Austin, 2011)) need to be applied for controlling the confounding variables.
- Do not trust the classifiers. Our results on style classification (§4.5.3) show that some styles (e.g., sarcasm, persona styles) are very difficult to predict, leading unreliable results in our analysis.
- Each dataset has its own issues. Some are only collected from specific domains (e.g., news articles), making the classifier biased to it. Some have a very imbalanced distribution over labels. Each data collection may or may not have annotation artifacts.

Stylistic Diversity of Text

Why are some text easier to change its style than others? Can we predict whether a text can be stylistically changeable or not? Different domains or genres of text may have different degrees of style diversity. For example, text from academic manuscripts may be more literal (stylistically less-diverse) than tweets.

Setup. We propose an ad-hoc measure to rank stylistic diversity of text using the fine-tuned style classifiers used in §4.5.3. Given a text, we first calculate the mean and the standard deviation (std) over the style probabilities $S_{1..53}$ predicted by the classifiers. We sort samples by the mean and take the top (or bottom) 10% samples first. Then, we sort the sampled tweets again by the std and take the top (or bottom) 10% samples. The final top or bottom ranked tweets are called *stylistically diverse or less diverse text*, indicating that the total amounts of style prediction scores and their variations are high (or less).

Stylistically diverse and less-diverse text. Table 4.14 shows the stylistically diverse (top) and less-diverse (bottom) tweets. We observe that stylistically diverse text uses more emotions and social expressions (e.g., complaining, greeting), while stylistically less diverse text is more literal and factual. Again, some predicted scores are not accurate due to the aforementioned weaknesses. We observe that the classifiers often predict very extreme scores (e.g., 0.99, 0.01) even though its true posterior (i.e., accuracy) does not correspond to certain amounts, where its posterior probabilities need to be calibrated accordingly.

	mean	std	formal	humor.	polite	sarcas.	metaph.	offens.
Stylistically diverse text	△	△						
i'm glad i can add hilarity into your life	.32	.45	.98	.99	0	.99	.99	0
it was really cool speaking with you today i look forward to working for you	.32	.45	.99	.99	.99	.99	0	0
i'm *ucking proud of you baby you've come a long way	.31	.45	0	.99	.99	0	.99	.99
Stylistically less diverse text	▽	▽						
lip/tongue tingling	.15	.28	.01	0	.02	0	0	0
satellite server is a mess cleaning up	.15	.28	0	0	.04	.01	.68	0
having beer with and some latin americans	.14	.28	0	0	.28	0	0	0

Table 4.14: Stylistically diverse (top, △) and less-diverse (bottom, ▽) text. Offensive words are replaced by *. More examples with full attributes are in the Appendix.

Analysis of style diversity across domains. We sample 100,000 sentences from different domains; tweets, academic papers, news articles, novel books, dialogues, movie scripts, and political debate (See Appendix for details). For each domain, we again predict probability scores of each style using the classifiers and then average the scores across sentences.

Figure 4.7 shows a proportion of the averaged prediction scores of each style over different domains. Text in academic papers has the least affective styles followed by news articles, while text in social media (e.g., tweets, Reddit) has a lot of style diversity, showing the correlation with freedom of speech in the domains. Interestingly, text in political debate has two conflicting styles pair in balance; high hate and happy, but less offensive and anger. More analyses on interpersonal, figurative, and personal styles are in the Appendix.

4.5.5 Conclusion

We build a benchmark **x**SLUE for studying cross-style language understanding and evaluation, where it includes 15 different styles and 23 classification tasks. Using the state-of-the-art classifiers trained on each style dataset, we provide interesting observations (e.g., cross-style classification/correlation, style diversity across domains). We believe **x**SLUE helps other researchers develop more solid systems on various applications of cross-style language.

4.6 Conclusion and Future Directions

Style is a still under-studied field where much more attention needs to be paid to data collection and in-depth analysis than to model-focused efforts. We present PASTEL, a parallelly annotated stylistic language dataset, with a careful design of denotation that preserves the meaning as well as maximizes the diversity of styles. The multiplicity of persona styles in PASTEL makes other style variables controlled (or fixed) except the target style for classification, which is a more accurate experimental design. Our benchmark corpus **x**SLUE is collected for studying the

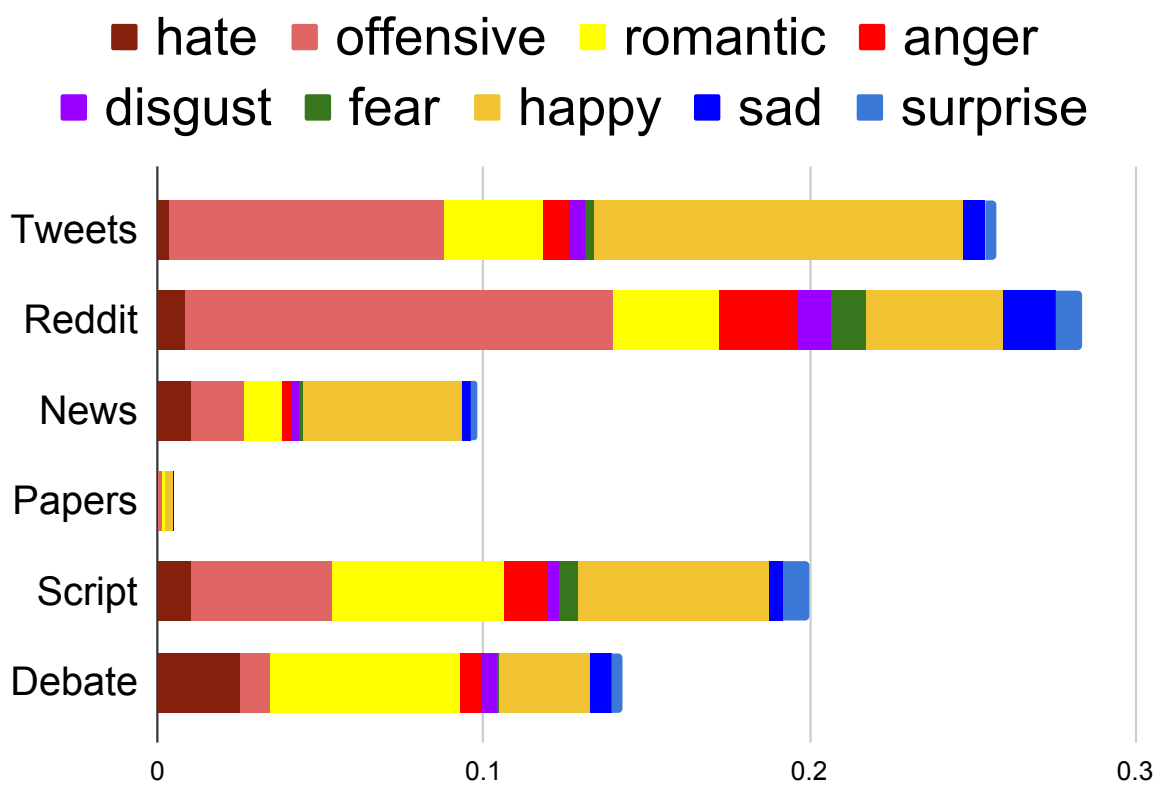


Figure 4.7: Diversity of affective styles on different domains: tweets, Reddit posts, news, papers, movie scripts, and political debates. Best viewed in color.

cross-style language understanding and evaluation. With the benchmark, we provide interesting observations such as cross-style dependency in written language and style diversity across domains. We believe PASTEL and **x**SLUE help other researchers develop more solid systems on various applications of cross-style language.

We summarize some directions for future style researches:

- **Content-style dependency**: in our ablation study, salient features for style classification are not only syntactic or lexical features but also content words (e.g., love, food). This is a counterexample to the hypothesis implicit in much of recent style research: *style* needs to be *separately modeled* from the *content*. We also observe that some texts remain similar across different annotator personas or across outputs from our transfer models, indicating that some content is stylistically invariant. Studying these and other aspects of the content-style relationship in PASTEL could be an interesting direction.
- **Cross-style modeling**: We have not yet explored any models which can learn the inter-style dependency structure. Developing such cross-style models would help find the complex combination of different styles. For example, instead of multiple classifiers for each style, developing a universal classifier on multiple styles where their internal representations are shared across styles might be an interesting direction.

- **Low-resource/performance styles**: Moreover, the cross-style models will be useful for some styles which have fewer annotation data (*low-resource* style) or some with very low performance due to the difficulty of the style language (*low-performance* style). For example, our study shows that detecting sarcasm and metaphor from text is still very difficult, which might be helped by other styles.
- **Cross-stylization on other applications**: Besides the style classification task, **x**SLUE can be applied to other applications such as style transfer or dialogues. For example, transferring one style without changing other styles or developing a more sympathetic dialogue system by controlling multiple styles might be interesting future directions.
- **Semantic and style drift in cross-stylization**: The biggest challenge in collecting cross-style datasets (Kang et al., 2019b) or controlling multiple styles in the generation (Ficler and Goldberg, 2017) is to diversify the style of text but at the same time preserve the meaning, in order to avoid *semantic drift*. It can be addressed by collecting text in parallel or preserving the meaning using various techniques. In the cross-style setting, multiple styles change at the same time in different parts of the text in a complicated way, leading to more server semantic drift. Moreover, we face a new challenge; *style drift*, where different styles are coupled together with text so changing one type may affect the others. For example, if we change it to the more impolite text given a text, such change tends to make the text more offensive and negative. In the cross-style setting, we first need to understand the underlying dependencies across style types, then develop a model that can handle the implicit dependencies.
- **Ethical concerns on model interpretation**: **A more careful interpretation is required.** In a cross-style language, some style types (e.g., personal styles) are very sensitive so require more careful interpretation of their result. We made three weak points about our analysis in §4.5.4, in order not to make any misleading points from our analysis. Any follow-up research on this direction needs to consider such ethical issues as well as provide potential weaknesses of their proposed methods.

Chapter 5

Conclusion

NLG is an extraordinarily rich and understudied topic in NLP. It exposes many as-yet unaddressed phenomena and aspects, especially in the realm of styles. NLG also requires complex, multi-faceted planning across all the facets discussed herein. An improved understanding of these issues will not only produce better NLG systems but enrich and inform other areas of NLP, like machine translation, summarization, question-answering, and dialogues. However, there are some fundamental advances needed in the field. This thesis has identified and addressed some of the facets at a certain degree. Without solving these problems, we will never be able to build human-like NLG systems.

Based on Halliday's SFL theory, there are three central themes in my work: making NLG systems more knowledgeable, coherently structured, and stylistically appropriate. For each facet, we developed prototypical yet effective, cognitive systems, specifically neural-symbolic integration, text planning, and cross-stylization. We believe these linguistically informed NLG systems will produce more human-like outputs, which may be some of the next steps toward artificial general intelligence (AGI) systems. Further attempts to address the remaining issues along with the aforementioned faceted systems should be considered.

Imagine an NLG system that can debate with a human, compose a narrative story for a novel, or write a critical review about your manuscript. Such human-like systems cannot be achieved by simply developing end-to-end neural systems nor a simple neural network trained on a large quantity of data. Rather, the technical contribution of my work lies in careful consideration of the entire development pipeline, from intentional task design to appropriate data collection, cognitive model development, and reproducible evaluation. Although many other parts of the pipeline are still missing or far from a human-like performance, the techniques outlined in this paper bring us closer to natural-sounding, yet artificially generated, text.

We suggest three main directions for future research:

- **Toward multi-faceted NLG applications.** My previous works studied individual facets separately. However, our daily conversations often require a combination of multiple facets. For instance, one can imagine a chatbot that can develop an empathetic relationship with users and help them manage symptoms of anxiety and depression (Weizenbaum, 1966). Such systems require modeling the patient's mental status, appropriately utilizing stylistic features to generate utterances, and strategically structuring the dialogue, with the goal of reducing the patient's depression. The recent advances of sociable conversational sys-

tems, like persuasive dialogue (Zhou et al., 2019), recommendation dialogue (Kang et al., 2019a), or negotiation dialogue (Lewis et al., 2017), are good examples of multifaceted systems. The development of such systems needs more cognitive architecture such that multiple facets can dynamically interact with each other. We would like to further explore more cognitive architectures for multi-faceted NLG systems based on multidisciplinary studies in cognitive science, psychology, and sociolinguistics.

- **Toward standardized NLG evaluation.** The lack of standardized evaluations and appropriate NLG tasks hampers the growth of NLG research and makes the progress unreliable and unreproducible. For example, automatic evaluation metrics (e.g., ROUGE, BLUE) are not appropriate for evaluating generation outputs, especially when the references can be open-ended or diverse (Gupta et al., 2019). On the other hand, human evaluations are very subjective, and their measurement of qualities like naturalness and consistency are not standardized (van der Lee et al., 2019). For a fair comparison of developed models and reproducible NLG research, we would like to develop an evaluation platform where researchers can plug in their systems and obtain benchmark results from standardized metrics. Such a platform needs to have following features: (1) scalable human metrics including the well-known ones (van der Lee et al., 2019) as well as new facets such as open-endedness, logicalness, level of abstraction, coherency, styles, and more, (2) multiple references with unbiased human annotators, and (3) goal-specific NLG tasks, like paragraph unmasking (Kang and Hovy, 2020), for more quantitative evaluations.
- **Toward interactive NLG systems.** The standard way of training and evaluating an NLG system is a *static* machine learning pipeline: the training data is collected or annotated by users and output from the system is measured automatically or using human judgement. However, the ML pipeline is not feasible for training and testing language generation, due to linguistic diversity. Can we make the pipeline more *dynamic* by adding a human to the training and evaluation loop? Imagine a human and a NLG machine collaborating to write a plot. The human writes one sentence, then the machine writes the next. Beyond the simple tasks required of the machine, such as grammar and spell checking, the machine can suggest the next sentence, re-write the text to be more formal, summarize the story, retrieve external articles to add references, check the coherence of text, and aid in other complex tasks. Creating human-machine NLG collaboration in a mixed-initiative way would be an interesting future direction.

References

Saurav Acharya. 2014. Causal modeling and prediction over event streams. 66

ACL. 2019. Fourth Conference on Machine Translation (WMT 2019). `http://www.statmt.org/wmt19/`. [Online; accessed 1-Feb-2020]. 13

Miltiadis Allamanis, Hao Peng, and Charles Sutton. 2016. A convolutional attention network for extreme summarization of source code. *arXiv preprint arXiv:1602.03001* . 70

Mehndiratta Surya Pratap Singh Tanwar Anand. 2014. Web Metric Summarization using Causal Relationship Graph. 66

Gabor Angeli and Christopher D Manning. 2014. Naturalli: Natural logic inference for common sense reasoning. In *EMNLP*. pages 534–545. 29

Douglas E Appelt. 1982. Planning natural-language utterances to satisfy multiple goals. Technical report, SRI INTERNATIONAL MENLO PARK CA ARTIFICIAL INTELLIGENCE CENTER. 66, 81, 88

Vikas Ganjigunte Ashok, Song Feng, and Yejin Choi. 2013. Success with style: Using writing style to predict the success of novels. In *Proc. of EMNLP*. pages 1753–1764. 123

Peter C Austin. 2011. An introduction to propensity score methods for reducing the effects of confounding in observational studies. *Multivariate behavioral research* 46(3):399–424. 165

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* . 76, 85, 153

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural machine translation by jointly learning to align and translate. In *ICLR*. 106, 108

Collin F Baker, Charles J Fillmore, and John B Lowe. 1998. The berkeley framenet project. In *Proceedings of the 17th international conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, pages 86–90. 48, 71, 74

Rose D Baker. 1995. Two permutation tests of equality of variances. *Statistics and Computing* 5(4):289–296. 47

Mikhail M. Bakhtin. 1981. The dialogic imagination. 138

Vipin Balachandran. 2013. Reducing human effort and improving quality in peer code reviews using automatic static analysis and reviewer recommendation. In *Proc. of ICSE*. 123

David Bamman, Jacob Eisenstein, and Tyler Schnoebelen. 2014. Gender identity and lexical variation in social media. *Journal of Sociolinguistics* 18(2):135–160. 143

Satanjeev Banerjee and Alon Lavie. 2005. Meteor: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the acl workshop on intrinsic and extrinsic evaluation measures for machine translation and/or summarization*. pages 65–72. 85, 96, 147

Siddhartha Banerjee, Prasenjit Mitra, and Kazunari Sugiyama. 2015. Multi-document abstractive summarization using ilp based multi-sentence compression. In *Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (IJCAI 2015)*. 69,

134

Michele Banko, Michael J Cafarella, Stephen Soderland, Matthew Broadhead, and Oren Etzioni. 2007. Open information extraction from the web. In *IJCAI*. volume 7, pages 2670–2676. 54

C Bradford Barber, David P Dobkin, David P Dobkin, and Hannu Huhdanpaa. 1996. The quickhull algorithm for convex hulls. *ACM Transactions on Mathematical Software (TOMS)* 22(4):469–483. 127

Regina Barzilay and Mirella Lapata. 2008. Modeling local coherence: An entity-based approach. *Computational Linguistics* 34(1):1–34. 4, 88

Islam Beltagy, Stephen Roller, Pengxiang Cheng, Katrin Erk, and Raymond J. Mooney. 2016. Representing meaning with a combination of logical and distributional models. *Computational Linguistics* 42:763–808. 29

Samy Bengio, Oriol Vinyals, Navdeep Jaitly, and Noam Shazeer. 2015. Scheduled sampling for sequence prediction with recurrent neural networks. In *Advances in Neural Information Processing Systems*. pages 1171–1179. 70, 134

Taylor Berg-Kirkpatrick, Dan Gillick, and Dan Klein. 2011. Jointly learning to extract and compress. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, pages 481–490. 69

Niko Besnier. 1990. Language and affect. *Annual review of anthropology* 19(1):419–451. 138

Chandra Bhagavatula, Sergey Feldman, Russell Power, and Waleed Ammar. 2018. Content-based citation recommendation. In *Proc. of NAACL*. 124

Douglas Biber. 1991. *Variation across speech and writing*. Cambridge University Press. 10, 11, 140, 144

Steffen Bickel and Tobias Scheffer. 2004. Learning from message pairs for automatic email answering. In *European Conference on Machine Learning*. Springer, pages 87–98. 68

Julia Birke and Anoop Sarkar. 2006. A clustering approach for nearly unsupervised recognition of nonliteral language. In *EACL*. 160, 161

Aliaksandr Birukou, Joseph R. Wakeling, Claudio Bartolini, Fabio Casati, Maurizio Marchese, Katsiaryna Mirylenka, Nardine Osman, Azzurra Ragone, Carles Sierra, and Aalam Wassef. 2011. Alternatives to peer review: Novel approaches for research evaluation. In *Front. Comput. Neurosci.*. 123

Eduardo Blanco, Nuria Castell, and Dan I Moldovan. 2008. Causal relation extraction. In *LREC*. 66, 74

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent dirichlet allocation. *JMLR* . 72

Leonard Bloomfield. 1927. Literate and illiterate speech. *American speech* 2(10):432–439. 144

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*. AcM, pages

1247–1250. 41

Andrea Bonaccorsi, Antonio Ferrara, and Marco Malgarini. 2018. Journal ratings as predictors of article quality in arts, humanities, and social sciences: An analysis based on the italian research evaluation exercise. In *The Evaluation of Research in Social Sciences and Humanities*, Springer, pages 253–267. 123

Antoine Bordes and Jason Weston. 2016. Learning end-to-end goal-oriented dialog. *arXiv preprint arXiv:1605.07683* . 68

Florian Boudin. 2016. pke: an open source python-based keyphrase extraction toolkit. In *Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: System Demonstrations*. Osaka, Japan, pages 69–73. http://aclweb.org/anthology/C16-2015. 91

Florian Boudin and Emmanuel Morin. 2013. Keyphrase extraction for n-best reranking in multi-sentence compression. In *North American Chapter of the Association for Computational Linguistics (NAACL)*. 69

Florian Boudin, Hugo Mougard, and Benoit Favre. 2015. Concept-based summarization using integer linear programming: From concept pruning to multiple optimal solutions. In *Conference on Empirical Methods in Natural Language Processing (EMNLP) 2015*. 69, 134

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*. 26, 48, 54

Roger Brown, Albert Gilman, et al. 1960. The pronouns of power and solidarity . 138

Sven Buechel and Udo Hahn. 2017. Emobank: Studying the impact of annotation perspective and representation format on dimensional emotion analysis. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. pages 578–585. 160, 161

Michael Byrd. 1986. The logic of natural language. 29

Donn Byrne. 1979. *Teaching writing skills*. Longman. 81, 88

Rafael A Calvo and Sunghwan Mac Kim. 2013. Emotions in text: dimensional and categorical models. *Computational Intelligence* 29(3):527–543. 141

Ricardo Campos, Vítor Mangaravite, Arian Pasquali, Alípio Jorge, Célia Nunes, and Adam Jatowt. 2020. Yake! keyword extraction from single documents using multiple local features. *Information Sciences* 509:257–289. 91

Jaime Carbonell and Jade Goldstein. 1998. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, pages 335–336. 69, 125, 134

Jean Carletta, Simone Ashby, Sebastien Bourban, Mike Flynn, Mael Guillemot, Thomas Hain, Jaroslav Kadlec, Vasilis Karaiskos, Wessel Kraaij, Melissa Kronenthal, et al. 2005. The ami meeting corpus: A pre-announcement. In *International Workshop on Machine Learning for Multimodal Interaction*. Springer, pages 28–39. 69, 125, 129

Nathanael Chambers and Dan Jurafsky. 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08: HLT*. pages 789–797. 88

Arjun Chandrasekaran, Ashwin K Vijayakumar, Stanislaw Antol, Mohit Bansal, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. 2016. We are humor beings: Understanding and predicting visual humor. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. pages 4603–4612. 159

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051* . 21

Desai Chen, Nathan Schneider, Dipanjan Das, and Noah A Smith. 2010. Semafor: Frame argument resolution with log-linear models. In *Proceedings of the 5th international workshop on semantic evaluation*. Association for Computational Linguistics, pages 264–267. 74

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, and Diana Inkpen. 2018. Natural language inference with external knowledge. In *ACL*. 26, 27, 29

Jianpeng Cheng and Mirella Lapata. 2016. Neural summarization by extracting sentences and words. *arXiv preprint arXiv:1603.07252* . 70, 134

Niyati Chhaya, Kushal Chawla, Tanya Goyal, Projjal Chanda, and Jaya Singh. 2018. Frustrated, polite, or formal: Quantifying feelings and tone in email. In *Proceedings of the Second Workshop on Computational Modeling of People's Opinions, Personality, and Emotions in Social Media*. Association for Computational Linguistics, New Orleans, Louisiana, USA, pages 76–86. https://doi.org/10.18653/v1/W18-1111. 143

Paul Chilton. 1990. Politeness, politics and diplomacy. *Discourse & Society* 1(2):201–224. 159

LI Chongxuan, Taufik Xu, Jun Zhu, and Bo Zhang. 2017. Triple generative adversarial nets. In *NIPS*. pages 4091–4101. 25

Junyoung Chung, Sungjin Ahn, and Yoshua Bengio. 2016. Hierarchical multiscale recurrent neural networks. *CoRR* abs/1609.01704. 67

Herbert H Clark and Dale H Schunk. 1980. Polite responses to polite requests. *Cognition* 8(2):111–143. 159

Peter Clark, Oyvind Tafjord, and Kyle Richardson. 2020. Transformers as soft reasoners over language. *arXiv preprint arXiv:2002.05867* . xix, 4, 5

Trevor Cohn and Mirella Lapata. 2008. Sentence compression beyond word deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*. Association for Computational Linguistics, pages 137–144. 69

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *Journal of Machine Learning Research* 12(Aug):2493–2537. 83

Simon Corston-Oliver, Eric Ringger, Michael Gamon, and Richard Campbell. 2004. Task-focused summarization of email. In *ACL-04 Workshop: Text Summarization Branches Out*. pages 43–50. 68

Nikolas Coupland. 2007. *Style: Language variation and identity*. Cambridge University Press.

138

CrowdFlower. 2016. text Emotion. http://www.crowdflower.com/wp-content/uploads/2016/07/text_emotion.csv. [Online; accessed 1-Oct-2019]. 160, 161

CrowdTruth. 2016. Short Text Corpus For Humor Detection. http://github.com/CrowdTruth/Short-Text-Corpus-For-Humor-Detection. [Online; accessed 1-Oct-2019]. 159, 161

Ido Dagan, Oren Glickman, and Bernardo Magnini. 2005. The PASCAL recognising textual entailment challenge. In *MLCW*. 51

Hercules Dalianis and Eduard Hovy. 1996. Aggregation in natural language generation. In *Trends in Natural Language Generation An Artificial Intelligence Perspective*, Springer, pages 88–105. 66

Bhavana Dalvi, Niket Tandon, and Peter Clark. 2017. Domain-targeted, high precision knowledge extraction. 54, 57

Niculescu-Mizil Cristian Danescu, Moritz Sudhof, Dan Jurafsky, Jure Leskovec, and Christopher Potts. 2013. A computational approach to politeness with application to social factors. *arXiv preprint arXiv:1306.6078* . 159, 161

Verna Dankers, Marek Rei, Martha Lewis, and Ekaterina Shutova. 2019. Modelling the interplay of metaphor and emotion through multitask learning. In *IJCNLP 2019*. 142, 143, 157

David K Danow. 1991. The thought of mikhail bakhtin from word to culture . 138

Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. 2017. Automated hate speech detection and the problem of offensive language. In *Eleventh international aaai conference on web and social media*. 160, 161

Pali UK De Silva and Candace K Vance. 2017. Preserving the quality of scientific research: Peer review of research articles. In *Scientific Scholarly Communication*, Springer, pages 73–99. 113

Luciano Del Corro and Rainer Gemulla. 2013. Clausie: clause-based open information extraction. In *Proceedings of the 22nd international conference on World Wide Web*. ACM, pages 355–366. 54

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*. 4, 21, 38, 88, 89, 91, 92, 96, 109, 126, 142, 151, 157, 162

Chrysanne DiMarco and Graeme Hirst. 1990. Accounting for style in machine translation. In *Third International Conference on Theoretical Issues in Machine Translation, Austin*. 138, 144

Quang Xuan Do, Yee Seng Chan, and Dan Roth. 2011. Minimally supervised event causality identification. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 294–303. 66

Dmitrij Dobrovol'skij and Elisabeth Piirainen. 2005. *Figurative language: Cross-cultural and cross-linguistic perspectives*. Brill. 157

William B. Dolan, Chris Quirk, and Chris Brockett. 2004. Unsupervised construction of large paraphrase corpora: Exploiting massively parallel news sources. In *COLING*. 48

John Duchi, Elad Hazan, and Yoram Singer. 2011. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of Machine Learning Research* 12(Jul):2121–2159. 85

Greg Durrett, Taylor Berg-Kirkpatrick, and Dan Klein. 2016. Learning-based single-document summarization with compression and anaphoricity constraints. *arXiv preprint arXiv:1603.08887* . 69

Penelope Eckert. 2000. *Language variation as social practice: The linguistic construction of identity in Belten High*. Wiley-Blackwell. 139

Penelope Eckert. 2008. Variation and the indexical field 1. *Journal of sociolinguistics* 12(4):453–476. 138

Penelope Eckert. 2019. The limits of meaning: Social indexicality, variation, and the cline of interiority. *Language* 95(4):751–776. 139

Harold P Edmundson. 1969. New methods in automatic extracting. *Journal of the ACM (JACM)* 16(2):264–285. 124

Paul Ekman. 1992. An argument for basic emotions. *Cognition & emotion* 6(3-4):169–200. 160

Günes Erkan and Dragomir R Radev. 2004. Lexrank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research* pages 457–479. 69, 134

Angela Fan, Mike Lewis, and Yann Dauphin. 2019. Strategies for structuring story generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, pages 2650–2660. https://doi.org/10.18653/v1/P19-1254. 67

Ferric C Fang, Anthony Bowen, and Arturo Casadevall. 2016. NIH peer review percentile scores are poorly predictive of grant productivity. *eLife* . 123

Manaal Faruqui, Jesse Dodge, Sujay K. Jauhar, Chris Dyer, Eduard Hovy, and Noah A. Smith. 2015. Retrofitting word vectors to semantic lexicons. In *In Proceedings of the 2015 Conference of NAACL*. 25, 27, 29, 35, 39, 45, 48

Jessica Ficler and Yoav Goldberg. 2017. Controlling linguistic style aspects in neural language generation. *arXiv preprint arXiv:1707.02633* . 141, 168

Elena Filatova. 2012. Irony and sarcasm: Corpus generation and analysis using crowdsourcing. In *LREC*. pages 392–398. 159

Katja Filippova. 2010. Multi-sentence compression: finding shortest paths in word graphs. In *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, pages 322–330. 69

Corina Florescu and Cornelia Caragea. 2017. Positionrank: An unsupervised approach to keyphrase extraction from scholarly documents. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. pages 1105–1115. 91

Yao Fu, Yansong Feng, and John P Cunningham. 2019. Paraphrase generation with latent bag of words. In *Advances in Neural Information Processing Systems*. pages 13623–13634. 67, 91, 93

Zhenxin Fu, Xiaoye Tan, Nanyun Peng, Dongyan Zhao, and Rui Yan. 2018. Style transfer in text: Exploration and evaluation. *CoRR* abs/1711.06861. 143, 150

Kavita Ganesan, ChengXiang Zhai, and Jiawei Han. 2010. Opinosis: a graph-based approach to abstractive summarization of highly redundant opinions. In *Proceedings of the 23rd international conference on computational linguistics*. Association for Computational Linguistics, pages 340–348. 69

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *NAACL-HLT*. pages 758–764. 27, 29, 48

A d'Avila Garcez, Tarek R Besold, Luc De Raedt, Peter Földiak, Pascal Hitzler, Thomas Icard, Kai-Uwe Kühnberger, Luis C Lamb, Risto Miikkulainen, and Daniel L Silver. 2015. Neural-symbolic learning and reasoning: contributions and challenges. In *Proceedings of the AAAI Spring Symposium on Knowledge Representation and Reasoning: Integrating Symbolic and Neural Approaches, Stanford*. 26

Claire Gardent, Anastasia Shimorina, Shashi Narayan, and Laura Perez-Beltrachini. 2017. Creating training corpora for nlg micro-planning. In *55th annual meeting of the Association for Computational Linguistics (ACL)*. 66

Albert Gatt and Emiel Krahmer. 2018. Survey of the state of the art in natural language generation: Core tasks, applications and evaluation. *Journal of Artificial Intelligence Research* 61:65–170. 66

Sebastian Gehrmann, Yuntian Deng, and Alexander M Rush. 2018. Bottom-up abstractive summarization. *arXiv preprint arXiv:1808.10792* . 124

Sebastian Gehrmann, Hendrik Strobelt, and Alexander M Rush. 2019. Gltr: Statistical detection and visualization of generated text. *arXiv preprint arXiv:1906.04043* . 4

Shima Gerani, Yashar Mehdad, Giuseppe Carenini, Raymond T Ng, and Bita Nejat. 2014. Abstractive summarization of product reviews using discourse structure. In *Proceedings of EMNLP*. 69

Aniruddha Ghosh and Tony Veale. 2016. Fracking sarcasm using neural network. In *Proceedings of the 7th workshop on computational approaches to subjectivity, sentiment and social media analysis*. pages 161–169. 159, 161

Dan Gillick and Benoit Favre. 2009. A scalable global model for summarization. In *Proceedings of the Workshop on Integer Linear Programming for Natural Langauge Processing*. Association for Computational Linguistics, pages 10–18. 69, 134

Roxana Girju. 2003. Automatic detection of causal relations for question answering. In *Proceedings of the ACL 2003 workshop on Multilingual summarization and question answering-Volume 12*. Association for Computational Linguistics, pages 76–83. 66, 74

Max Glockner, Vered Shwartz, and Yoav Goldberg. 2018. Breaking nli systems with sentences that require simple lexical inferences. In *ACL*. 29

Sam Glucksberg, Matthew S McGlone, Yosef Grodzinsky, and Katrin Amunts. 2001. *Understanding figurative language: From metaphor to idioms*. 36. Oxford University Press on Demand. 157

Erving Goffman et al. 1978. *The presentation of self in everyday life*. Harmondsworth London. 139

Yichen Gong, Heng Luo, and Jian Zhang. 2017. Natural language inference over interaction space. *arXiv preprint arXiv:1709.04348* . 26

Roberto González-Ibánez, Smaranda Muresan, and Nina Wacholder. 2011. Identifying sarcasm in twitter: a closer look. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: Short Papers-Volume 2*. Association for Computational Linguistics, pages 581–586. 159

Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. 2014. Generative adversarial nets. In *Advances in neural information processing systems*. pages 2672–2680. 27, 34

Google. 2016. Freebase Data Dumps. `https://developers.google.com/freebase/data`. 75

Philip John Gorinski and Mirella Lapata. 2015. Movie script summarization as graph-based scene extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. pages 1066–1076. 69, 129

Andreas Graefe. 2016. Guide to automated journalism . 1

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2003. English gigaword. *Linguistic Data Consortium, Philadelphia* 4(1):34. 129

Clive WJ Granger. 1988. Some recent development in a concept of causality. *Journal of econometrics* 39(1):199–211. 66, 70, 73

Sarah Greaves, Joanna Scott, Maxine Clarke, Linda Miller, Timo Hannay, Annette Thomas, and Philip Campbell. 2006. Nature's trial of open peer review. *Nature* 444(971):10–1038. 113

Cécile Grivaz. 2010. Human judgements on causation in french texts. In *LREC*. 66

Barbara J Grosz, Scott Weinstein, and Aravind K Joshi. 1995. Centering: A framework for modeling the local coherence of discourse. *Computational linguistics* 21(2):203–225. 4

Max Grusky, Mor Naaman, and Yoav Artzi. 2018. Newsroom: A dataset of 1.3 million summaries with diverse extractive strategies. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Association for Computational Linguistics, New Orleans, Louisiana, pages 708–719. http://aclweb.org/anthology/N18-1065. 69, 129

Prakhar Gupta, Shikib Mehri, Tiancheng Zhao, Amy Pavel, Maxine Eskenazi, and Jeffrey P Bigham. 2019. Investigating evaluation of open-domain dialogue systems with human generated multiple references. *arXiv preprint arXiv:1907.10568* . 170

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel Bowman, and

Noah A. Smith. 2018a. Annotation artifacts in natural language inference data. In *NAACL (Short Papers)*. 56

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018b. Annotation artifacts in natural language inference data. In *NAACL*. 27

Suchin Gururangan, Swabha Swayamdipta, Omer Levy, Roy Schwartz, Samuel R Bowman, and Noah A Smith. 2018c. Annotation artifacts in natural language inference data. *arXiv preprint arXiv:1803.02324* . 49

Jung-Woo Ha, Dongyeop Kang, Hyuna Pyo, and Jeonghee Kim. 2015. News2images: Automatically summarizing news articles into image-based contents via deep learning . 124

Aria Haghighi, Andrew Ng, and Christopher Manning. 2005. Robust textual inference via graph matching. In *EMNLP*. 25

Michael Alexander Kirkwood Halliday. 1976. *An interpretation of the functional relationship between language and social structure*. MAK Halliday. 8

Michael Alexander Kirkwood Halliday. 2003a. *On language and linguistics*, volume 3. A&C Black. xiii, 8

Michael Alexander Kirkwood Halliday. 2003b. On the "architecture" of human language. *On language and linguistics* 3:1–29. 8

James Douglas Hamilton. 1994. *Time series analysis*, volume 2. Princeton university press Princeton. 73, 79

Mengqiao Han, Ou Wu, and Zhendong Niu. 2017. Unsupervised Automatic Text Style Transfer using LSTM. In *National CCF Conference on Natural Language Processing and Chinese Computing*. Springer, pages 281–292. 143

He He, Anusha Balakrishnan, Mihail Eric, and Percy Liang. 2017a. Learning symmetric collaborative dialogue agents with dynamic knowledge graph embeddings. In *ACL*. 68

He He, Derek Chen, Anusha Balakrishnan, and Percy Liang. 2018. Decoupling strategy and generation in negotiation dialogues. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Brussels, Belgium, pages 2333–2343. https://www.aclweb.org/anthology/D18-1256. 68

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. pages 770–778. 55

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. 2017b. Neural collaborative filtering. In *WWW*. 68, 99

Bodil Helder. 2011. *Textual analysis: An approach to analysing professional texts*. Samfundslitteratur. 2

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*. pages 1693–1701. 21

Francis Heylighen and Jean-Marc Dewaele. 1999. Formality of language: definition, measurement and behavioral determinants. *Interner Bericht, Center "Leo Apostel", Vrije Universiteit Brüssel* . 143, 159

Katerina Hlaváčková-Schindler, Milan Paluš, Martin Vejmelka, and Joydeep Bhattacharya. 2007. Causality detection based on information-theoretic approaches in time series analysis. *Physics Reports* 441(1):1–46. 66

Sepp Hochreiter and Jürgen Schmidhuber. 1997a. Long short-term memory. *Neural computation* 9(8):1735–1780. 69, 76, 79, 85, 95, 105, 108, 162

Sepp Hochreiter and Jürgen Schmidhuber. 1997b. Long short-term memory. *Neural computation* 9(8):1735–1780. 121, 152

Janet Holmes and Maria Stubbe. 2015. *Power and politeness in the workplace: A sociolinguistic analysis of talk at work*. Routledge. 159

Ari Holtzman, Jan Buys, Maxwell Forbes, and Yejin Choi. 2019. The curious case of neural text degeneration. *ArXiv* abs/1904.09751. 95

Kai Hong and Ani Nenkova. 2014. Improving the estimation of word importance for news multi-document summarization. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*. pages 712–721. 69, 124, 126

Eduard Hovy. 1987. Generating natural language under pragmatic constraints. *Journal of Pragmatics* 11(6):689–719. 10, 11, 14, 15, 137, 140, 143, 157, 165

Eduard Hovy, Chin-Yew Lin, et al. 1999. Automated text summarization in summarist. *Advances in automatic text summarization* 14. 1

Eduard H. Hovy. 1985. Integrating text planning and production in generation. In *IJCAI*. 66

Eduard H Hovy. 1990. Pragmatics and natural language generation. *Artificial Intelligence* 43(2):153–197. 66

Eduard H Hovy. 1991. Approaches to the planning of coherent text. In *Natural language generation in artificial intelligence and computational linguistics*, Springer, pages 83–102. 81, 88

Patrik O Hoyer. 2004. Non-negative matrix factorization with sparseness constraints. *Journal of machine learning research* 5(Nov):1457–1469. 73

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *KDD*. 48

Zhiting Hu, Xuezhe Ma, Zhengzhong Liu, Eduard Hovy, and Eric Xing. 2016. Harnessing deep neural networks with logic rules. *ACL* . 25

Zhiting Hu, Zichao Yang, Xiaodan Liang, Ruslan Salakhutdinov, and Eric P Xing. 2017. Toward controlled generation of text. In *International Conference on Machine Learning*. pages 1587–1596. 143, 144, 153

Xinyu Hua and Lu Wang. 2019. Sentence-level content planning and style specification for neural text generation. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, Hong Kong, China,

pages 591–602. https://doi.org/10.18653/v1/D19-1055. 67, 89, 91

Ting-Hao (Kenneth) Huang, Francis Ferraro, Nasrin Mostafazadeh, Ishan Misra, Aishwarya Agrawal, Jacob Devlin, Ross Girshick, Xiaodong He, Pushmeet Kohli, Dhruv Batra, C. Lawrence Zitnick, Devi Parikh, Lucy Vanderwende, Michel Galley, and Margaret Mitchell. 2016. Visual storytelling. In *NAACL*. https://www.microsoft.com/en-us/research/publication/visual-storytelling/. 146, 148

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *CoRR* abs/1508.01991. 83

Thomas Icard III and Lawrence Moss. 2014. Recent progress in monotonicity. *LiLT (Linguistic Issues in Language Technology)* 9. 31

Suzana Ilić, Edison Marrese-Taylor, Jorge A Balazs, and Yutaka Matsuo. 2018. Deep contextualized word representations for detecting sarcasm and irony. *arXiv preprint arXiv:1809.09795* . 159

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. 2015. Deep unordered composition rivals syntactic methods for text classification. In *Proc. of ACL-IJCNLP*. volume 1, pages 1681–1691. 121, 151

James B Jacobs, Kimberly Potter, et al. 1998. *Hate crimes: Criminal law & identity politics*. Oxford University Press on Demand. 160

Alexandra Jaffe et al. 2009. *Stance: sociolinguistic perspectives*. OUP USA. 139

Hyeju Jang, Keith Maki, Eduard Hovy, and Carolyn Rose. 2017. Finding structure in figurative language: Metaphor detection with topic-based frames. In *Proceedings of the 18th Annual SIGdial Meeting on Discourse and Dialogue*. pages 320–330. 160

Sujay Kumar Jauhar, Chris Dyer, and Eduard Hovy. 2015. Ontologically grounded multi-sense representation learning for semantic vector space models. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. pages 683–693. 25

Harsh Jhamtani, Varun Gangal, Eduard H. Hovy, and Eric Nyberg. 2017. Shakespearizing modern language using copy-enriched sequence-to-sequence models. *CoRR* abs/1707.01161. 143

Yangfeng Ji and Jacob Eisenstein. 2014. Representation learning for text-level discourse parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. volume 1, pages 13–24. 67, 83, 85

R. Jia and P. Liang. 2017. Adversarial examples for evaluating reading comprehension systems. In *Empirical Methods in Natural Language Processing (EMNLP)*. 26

José Jiménez. 2015. Neural network architecture for time series forecasting. `https://github.com/josejimenezluna/nnet-ts`. 79

Barbara Johnstone. 2010. Locating language in identity. *Language and identities* 31:29–36. 138

Aditya Joshi, Pushpak Bhattacharyya, and Mark J Carman. 2017. Automatic sarcasm detection: A survey. *ACM Computing Surveys (CSUR)* 50(5):73. 159

Aditya Joshi, Vaibhav Tripathi, Kevin Patel, Pushpak Bhattacharyya, and Mark Carman. 2016. Are word embedding-based features useful for sarcasm detection? *arXiv preprint arXiv:1610.00883* . 159

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2016. Bag of tricks for efficient text classification. *arXiv preprint arXiv:1607.01759* . 108

David A Jurgens, Peter D Turney, Saif M Mohammad, and Keith J Holyoak. 2012. Semeval-2012 task 2: Measuring degrees of relational similarity. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics-Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*. Association for Computational Linguistics, pages 356–364. 41

Dongyeop Kang, Waleed Ammar, Bhavana Dalvi, Madeleine van Zuylen, Sebastian Kohlmeier, Eduard Hovy, and Roy Schwartz. 2018a. A dataset of peer reviews (peerread): Collection, insights and nlp applications. In *Meeting of the North American Chapter of the Association for Computational Linguistics (NAACL)*. New Orleans, USA. 19, 65, 69, 129

Dongyeop Kang, Waleed Ammar, Bhavana Dalvi, Madeleine van Zuylen, Sebastian Kohlmeier, Eduard Hovy, and Roy Schwartz. 2018b. A dataset of peer reviews (peerread): Collection, insights and nlp applications. In *Proceedings of NAACL-HLT*. xiv, 63, 64, 84, 136

Dongyeop Kang, Anusha Balakrishnan, Pararth Shah, Paul Crook, Y-Lan Boureau, and Jason Weston. 2019a. Recommendation as a communication game: Self-supervised bot-play for goal-oriented dialogue. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Hong Kong. http://arxiv.org/. xiv, 19, 63, 64, 65, 136, 170

Dongyeop Kang, Michael Gamon, Patrick Pantel, Ahmed Hassan Awadallah, Madian Khabsa, Mark Encarnacion, Chetan Bansal, and Eduard Hovy. 2017a. Email generation with diversity and variational prior. *preprint* . xiv, 64, 65, 136

Dongyeop Kang, Varun Gangal, and Eduard Hovy. 2019b. (male, bachelor) and (female, ph.d) have different connotations: Parallelly annotated stylistic language dataset with multiple personas. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Hong Kong. https://arxiv.org/abs/1909.00098. 19, 141, 142, 143, 157, 161, 165, 168

Dongyeop Kang, Varun Gangal, Ang Lu, Zheng Chen, and Eduard Hovy. 2017b. Detecting and explaining causes from text for a time series event. In *Conference on Empirical Methods on Natural Language Processing*. xiv, 18, 26, 64, 65, 67, 136, 164

Dongyeop Kang, DongGyun Han, NaHea Park, Sangtae Kim, U Kang, and Soobin Lee. 2014. Eventera: Real-time event recommendation system from massive heterogeneous online media. In *IEEE International Conference on Data Mining (ICDM)*. 66

Dongyeop Kang, Hiroaki Hayashi, Alan W Black, and Eduard Hovy. 2019c. Linguistic versus latent relations for modeling coherent flow in paragraphs. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Hong Kong. https://arxiv.org/abs/1908.11790. xiv, 18, 63, 64, 65, 68, 88, 89, 90, 93, 95, 96, 136

Dongyeop Kang and Eduard Hovy. 2019. xslue: A benchmark and analysis platform for cross-

style language understanding and evaluation. 19, 140, 142

Dongyeop Kang and Eduard Hovy. 2020. Plan ahead: Content-guided text planning for partially, masked paragraph generation. In *Preprint*. xiv, 18, 63, 64, 65, 136, 170

Dongyeop Kang, Taehee Jung, and Eduard Hovy. 2020. Don't treat every relation the same! constraining word vectors to geometric regularities of inter-word relations. In *Preprint*. xiii, 17, 24

Dongyeop Kang, Taehee Jung, Lucas Mentch, and Eduard Hovy. 2019d. Earlier isn't always better: Sub-aspect analysis on corpus and system biases in summarization. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Hong Kong. https://arxiv.org/abs/1908.11723. xiv, 19, 63, 64, 95, 136, 159

Dongyeop Kang, Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018c. Bridging knowledge gaps in neural entailment via symbolic models. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Brussels, Belgium. xiii, 17, 24

Dongyeop Kang, Tushar Khot, Ashish Sabharwal, and Eduard Hovy. 2018d. Adventure: Adversarial training for textual entailment with knowledge-guided examples. In *The 56th Annual Meeting of the Association for Computational Linguistics (ACL)*. Melbourne, Australia. xiii, 17, 24, 25, 26, 67

Anjuli Kannan, Karol Kurach, Sujith Ravi, Tobias Kaufmann, Andrew Tomkins, Balint Miklos, Greg Corrado, László Lukács, Marina Ganea, Peter Young, et al. 2016. Smart reply: Automated response suggestion for email. In *Proceedings of the ACM SIGKDD Conference on Knowledge Discovery and Data Mining (KDD)*. volume 36, pages 495–503. 1, 19, 63, 65, 69

Chris Kedzie, Kathleen McKeown, and Hal Daume III. 2018. Content selection in deep learning models of summarization. *arXiv preprint arXiv:1810.12343* . 69, 70, 124, 125, 129, 134

Geoffrey Keppel. 1991. *Design and analysis: A researcher's handbook*. Prentice-Hall, Inc. 165

Daniel Khashabi, Tushar Khot, Ashish Sabharwal, Peter Clark, Oren Etzioni, and Dan Roth. 2016. Question answering via integer programming over semi-structured knowledge. *arXiv preprint arXiv:1604.06076* . 26, 55

Mikhail Khodak, Nikunj Saunshi, and Kiran Vodrahalli. 2017. A large self-annotated corpus for sarcasm. *arXiv preprint arXiv:1704.05579* . 159, 160, 161

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2017. Answering complex questions using open information extraction. *arXiv preprint arXiv:1704.05572* . 26, 54, 55

Tushar Khot, Ashish Sabharwal, and Peter Clark. 2018. SciTail: A textual entailment dataset from science question answering. *AAAI* . 13, 27, 35, 48, 52, 53, 56, 57

Chloe Kiddon and Yuriy Brun. 2011. That's what she said: Double entendre identification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*. Association for Computational Linguistics, pages 89–94. 159

Soo-Min Kim and Eduard Hovy. 2004. Determining the sentiment of opinions. In *Proceedings of the 20th international conference on Computational Linguistics*. Association for Compu-

tational Linguistics, page 1367. 160

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR* abs/1412.6980. 154

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*. 108

Moshe Koppel, Jonathan Schler, and Shlomo Argamon. 2009. Computational methods in authorship attribution. *Journal of the American Society for information Science and Technology* 60(1):9–26. 144

Yehuda Koren, Robert M. Bell, and Chris Volinsky. 2009. Matrix factorization techniques for recommender systems. *Computer* 42. 68, 99

Zornitsa Kozareva. 2012. Cause-effect relation learning. In *Workshop Proceedings of TextGraphs-7 on Graph-based Methods for Natural Language Processing*. Association for Computational Linguistics, pages 39–43. 66, 71

William Labov. 1966. The social stratification of english in new york city. . 139

John D. Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*. 83

George Lakoff. 1970. Linguistics and Natural Logic. *Synthese* 22(1-2):151–271. 28, 29

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2017. Unsupervised machine translation using monolingual corpora only. *arXiv preprint arXiv:1711.00043* . 153, 154

John Langford and Mark Guzdial. 2015. The arbitrariness of reviews, and advice for school administrators. *Communications of the ACM Blog* 58(4):12–13. 113, 123

Mirella Lapata and Regina Barzilay. 2005. Automatic evaluation of text coherence: Models and representations. In *IJCAI*. volume 5, pages 1085–1090. 4

Claire Le Goues, Yuriy Brun, Sven Apel, Emery Berger, Sarfraz Khurshid, and Yannis Smaragdakis. 2017. Effectiveness of anonymization in double-blind review. ArXiv:1709.01609. 123

Benjamin J. Lengerich, Andrew L. Maas, and Christopher Potts. 2018. Retrofitting distributional embeddings to knowledge graphs with functional relations. In *COLING*. 25

Vladimir I. Levenshtein. 1966. Binary codes capable of correcting deletions, insertions, and reversals. *Soviet physics doklady* 10(8):707–710. 115

Omer Levy and Yoav Goldberg. 2014. Linguistic regularities in sparse and explicit word representations. In *Proceedings of the eighteenth conference on computational natural language learning*. pages 171–180. 25, 38, 42, 49

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics* 3:211–225. 25

Mike Lewis, Denis Yarats, Yann Dauphin, Devi Parikh, and Dhruv Batra. 2017. Deal or no deal? end-to-end learning for negotiation dialogues. In *EMNLP*. 13, 63, 68, 106, 170

Jiwei Li, Minh-Thang Luong, and Daniel Jurafsky. 2015. A hierarchical neural autoencoder for paragraphs and documents. In *ACL*. 67

Raymond Li, Samira Ebrahimi Kahou, Hannes Schulz, Vincent Michalski, Laurent Charlin, and Chris Pal. 2018. Towards deep conversational recommendations. In *NIPS*. pages 9725–9735. 68, 99

Xin Li and Dan Roth. 2002. Learning question classifiers. In *COLING*. 48

Yanran Li, Hui Su, Xiaoyu Shen, Wenjie Li, Ziqiang Cao, and Shuzi Niu. 2017. Dailydialog: A manually labelled multi-turn dialogue dataset. *arXiv preprint arXiv:1710.03957* . 160, 161

Chen Liang, Jonathan Berant, Quoc Le, Kenneth D Forbus, and Ni Lao. 2016. Neural symbolic machines: Learning semantic parsers on freebase with weak supervision. *arXiv preprint arXiv:1611.00020* . 25, 26

Chin-Yew Lin. 2004. Rouge: A package for automatic evaluation of summaries. In *Text summarization branches out: Proceedings of the ACL-04 workshop*. volume 8. 126

Chin-Yew Lin and Eduard Hovy. 1997. Identifying topics by position. In *Fifth Conference on Applied Natural Language Processing*. 69, 124, 126

Chin-Yew Lin and Eduard Hovy. 2000. The automated acquisition of topic signatures for text summarization. In *Proceedings of the 18th conference on Computational linguistics-Volume 1*. Association for Computational Linguistics, pages 495–501. 69, 128

Chin-Yew Lin and Eduard Hovy. 2003. Automatic evaluation of summaries using n-gram co-occurrence statistics. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*. Association for Computational Linguistics, pages 71–78. 147, 154

Chu-Cheng Lin, Dongyeop Kang, Michael Gamon, Madian Khabsa, Ahmed Hassan Awadallah, and Patrick Pantel. 2018. Actionable email intent modeling with reparametrized rnns. In *arXiv:1712.09185; To Appear in AAAI 2018*. 65, 136

Hui Lin and Jeff Bilmes. 2010. Multi-document summarization via budgeted maximization of submodular functions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 912–920. 69, 134

Hui Lin and Jeff Bilmes. 2011. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*. Association for Computational Linguistics, pages 510–520. 69, 125, 134

Hui Lin and Jeff A Bilmes. 2012. Learning mixtures of submodular shells with application to document summarization. *arXiv preprint arXiv:1210.4871* . 69, 125, 134

Rui Lin, Shujie Liu, Muyun Yang, Mu Li, Ming Zhou, and Sheng Li. 2015a. Hierarchical recurrent neural network for document modeling. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. pages 899–907. 67, 84, 85

Yankai Lin, Zhiyuan Liu, Huan-Bo Luan, Maosong Sun, Siwei Rao, and Song Liu. 2015b. Modeling relation paths for representation learning of knowledge bases. In *EMNLP*. 41

Tal Linzen. 2016. Issues in evaluating semantic spaces using word analogies. In *RepEval@ACL*. 25, 39, 42

Zachary C Lipton, Sharad Vikram, and Julian McAuley. 2015. Generative concatenative nets jointly learn to Write and Classify reviews. *arXiv preprint arXiv:1511.03683* . 143

Bing Liu et al. 2010. Sentiment analysis and subjectivity. *Handbook of natural language processing* 2(2010):627–666. 143

Chia-Wei Liu, Ryan Lowe, Iulian V Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How not to evaluate your dialogue system: An empirical study of unsupervised evaluation metrics for dialogue response generation. *arXiv preprint arXiv:1603.08023* . 85, 96, 147, 154

Fei Liu, Jeffrey Flanigan, Sam Thomson, Norman Sadeh, and Noah A Smith. 2015. Toward abstractive summarization using semantic representations. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. pages 1077–1086. 69

Birte Loenneker-Rodman and Srini Narayanan. 2010. Computational approaches to figurative language. *Cambridge Encyclopedia of Psycholinguistics* . 157

Thang Luong, Hieu Pham, and Christopher D. Manning. 2015. Effective approaches to attention-based neural machine translation. In *EMNLP*. 30

Hao Ma, Dengyong Zhou, Chao Liu, Michael R. Lyu, and Irwin King. 2011. Recommender systems with social regularization. In *WSDM*. 68

Laurens van der Maaten and Geoffrey Hinton. 2008. Visualizing data using t-sne. *Journal of Machine Learning Research* 9(Nov):2579–2605. 42

Bill MacCartney and Christopher D Manning. 2014. Natural logic and natural language inference. In *Computing meaning*, Springer, pages 129–147. 28

Rahul Malik, L Venkata Subramaniam, and Saroj Kaushik. 2007. Automatically selecting answer templates to respond to customer emails. In *IJCAI*. volume 7, pages 1659–1664. 68

William C Mann and Sandra A Thompson. 1988. Rhetorical structure theory: Toward a functional theory of text organization. *Text-Interdisciplinary Journal for the Study of Discourse* 8(3):243–281. 65, 67, 82

Daniel Marcu. 1999. Discourse trees are good indicators of importance in text. *Advances in automatic text summarization* 293:123–136. 127

Marco Marelli, Luisa Bentivogli, Marco Baroni, Raffaella Bernardi, Stefano Menini, and Roberto Zamparelli. 2014a. Semeval-2014 task 1: Evaluation of compositional distributional semantic models on full sentences through semantic relatedness and textual entailment. In *SemEval@COLING*. 48

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014b. A sick cure for the evaluation of compositional distributional semantic models. In *LREC*. pages 216–223. 29

James H Martin. 1992. Computer understanding of conventional metaphoric language. *Cogni-*

tive science 16(2):233–270. 160

Yasuko Matsubara, Yasushi Sakurai, B. Aditya Prakash, Lei Li, and Christos Faloutsos. 2012. Rise and fall patterns of information diffusion: model and implications. In *KDD*. pages 6–14. 78, 79

Bryan McCann, James Bradbury, Caiming Xiong, and Richard Socher. 2017. Learned in translation: Contextualized word vectors. In *NIPS*. 25

Ryan McDonald. 2007. *A study of global inference algorithms in multi-document summarization*. Springer. 69

Stephen McGregor, Elisabetta Jezek, Matthew Purver, and Geraint A. Wiggins. 2017. A geometric method for detecting semantic coercion. In *IWCS*. 39

Kathleen R McKeown. 1985. Discourse strategies for generating natural-language text. *Artificial Intelligence* 27(1):1–41. 81, 88

Yashar Mehdad, Giuseppe Carenini, and Raymond Ng. 2014. Abstractive summarization of spoken and written conversations based on phrasal queries. In *Proc. of ACL*. pages 1220–1230. 69

Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. 2016. Pointer sentinel mixture models. *arXiv preprint arXiv:1609.07843* . 95

Lesly Miculicich, Marc Marone, and Hany Hassan. 2019. Selecting, planning, and rewriting: A modular approach for data-to-document generation and translation. In *Proceedings of the 3rd Workshop on Neural Generation and Translation*. Association for Computational Linguistics, Hong Kong, pages 289–296. https://doi.org/10.18653/v1/D19-5633. 67, 89, 91

Rada Mihalcea and Hakan Ceylan. 2007. Explorations in automatic book summarization. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*. 69, 129

Rada Mihalcea and Carlo Strapparava. 2006. Learning to laugh (automatically): Computational models for humor recognition. *Computational Intelligence* 22(2):126–142. 159

Rada Mihalcea and Paul Tarau. 2004. Textrank: Bringing order into text. In *Proceedings of the 2004 conference on empirical methods in natural language processing*. pages 404–411. 69, 134

T Mikolov and J Dean. 2013. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems* . 38, 41, 76

Tomas Mikolov, Kai Chen, Gregory S. Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR* abs/1301.3781. 38, 41, 42

Alexander H. Miller, Will Feng, Dhruv Batra, Antoine Bordes, Adam Fisch, Jiasen Lu, Devi Parikh, and Jason Weston. 2017. Parlai: A dialog research software platform. In *EMNLP*. 103, 108

Alexander H. Miller, Adam Fisch, Jesse Dodge, Amir-Hossein Karimi, Antoine Bordes, and Jason Weston. 2016. Key-value memory networks for directly reading documents. In *EMNLP*. 102

George A Miller. 1995. WordNet: a lexical database for english. *Communications of the ACM* 38(11):39–41. 27, 29, 48

David M. Mimno and Laure Thompson. 2017. The strange geometry of skip-gram with negative sampling. In *EMNLP*. 39

Saif Mohammad, Ekaterina Shutova, and Peter Turney. 2016. Metaphor as a medium for emotion: An empirical study. In *Proceedings of the Fifth Joint Conference on Lexical and Computational Semantics*. Association for Computational Linguistics, Berlin, Germany, pages 23–33. https://doi.org/10.18653/v1/S16-2003. 142, 143, 157

Michael Mohler, David Bracewell, Marc Tomlinson, and David Hinote. 2013. Semantic signatures for example-based linguistic metaphor detection. In *Proceedings of the First Workshop on Metaphor in NLP*. pages 27–35. 160

Michael Mohler, Mary Brunson, Bryan Rink, and Marc T Tomlinson. 2016. Introducing the lcc metaphor datasets. In *LREC*. 160

Raymond J Mooney and Loriene Roy. 2000. Content-based book recommending using learning for text categorization. In *Proceedings of the fifth ACM conference on Digital libraries*. ACM, pages 195–204. 68

Johanna D Moore and Cécile L Paris. 1993. Planning text for advisory dialogues: Capturing intentional and rhetorical information. *Computational linguistics* 19(4):651–694. 67

Amit Moryossef, Yoav Goldberg, and Ido Dagan. 2019. Step-by-step: Separating planning from realization in neural data-to-text generation. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. Association for Computational Linguistics, Minneapolis, Minnesota, pages 2267–2277. https://doi.org/10.18653/v1/N19-1236. 67, 89, 91

Nasrin Mostafazadeh, Nathanael Chambers, Xiaodong He, Devi Parikh, Dhruv Batra, Lucy Vanderwende, Pushmeet Kohli, and James Allen. 2016. A corpus and evaluation framework for deeper understanding of commonsense stories. *arXiv preprint arXiv:1604.01696* . 88

Abhinav Moudgil. 2017. short jokes dataset. `https://github.com/amoudgl/short-jokes-dataset`. [Online; accessed 1-Oct-2019]. 159, 161

Nikola Mrksic, Diarmuid Ó Séaghdha, Blaise Thomson, Milica Gasic, Lina Maria Rojas-Barahona, Pei hao Su, David Vandyke, Tsung-Hsien Wen, and Steve J. Young. 2016. Counter-fitting word vectors to linguistic constraints. In *HLT-NAACL*. 25, 27, 29

Ramesh Nallapati, Feifei Zhai, and Bowen Zhou. 2017. Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In *Thirty-First AAAI Conference on Artificial Intelligence*. 70, 134

Ramesh Nallapati, Bowen Zhou, Cicero dos Santos, Caglar Gulcehre, and Bing Xiang. 2016. Abstractive text summarization using sequence-to-sequence rnns and beyond. In *Proceedings of The 20th SIGNLL Conference on Computational Natural Language Learning*. pages 280–290. 69, 124, 129

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018a. Don't give me the details, just

the summary! topic-aware convolutional neural networks for extreme summarization. *arXiv preprint arXiv:1808.08745* . 69, 124, 125, 129

Shashi Narayan, Shay B Cohen, and Mirella Lapata. 2018b. Ranking sentences for extractive summarization with reinforcement learning. *arXiv preprint arXiv:1802.08636* . 70, 124

Gonzalo Navarro. 2001. A guided tour to approximate string matching. *ACM computing surveys (CSUR)* 33(1):31–88. 153

Dong Nguyen, A Seza Doğruöz, Carolyn P Rosé, and Franciska de Jong. 2016. Computational sociolinguistics: A survey. *Computational linguistics* 42(3):537–593. 143

Malvina Nissim and Katja Markert. 2003. Syntactic features and word similarity for supervised metonymy resolution. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*. Association for Computational Linguistics, pages 56–63. 160

Xing Niu, Sudha Rao, and Marine Carpuat. 2018. Multi-task neural models for translating between styles within and across languages. *arXiv preprint arXiv:1806.04357* . 14

Lauri Nummenmaa, Enrico Glerean, Mikko Viinikainen, Iiro P Jääskeläinen, Riitta Hari, and Mikko Sams. 2012. Emotions promote social interaction by synchronizing brain activity across individuals. *Proceedings of the National Academy of Sciences* 109(24):9599–9604. 16, 141

Elinor Ochs. 1990. Indexicality and socialization. *Cultural psychology: Essays on comparative human development* . 138

OpenAI. 2018. Openai five. `https://blog.openai.com/openai-five/`. 106

Jessica Ouyang, Serina Chang, and Kathy McKeown. 2017. Crowd-sourced iterative annotation for narrative summarization corpora. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*. pages 46–51. 69, 129

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *ACL*. 48

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *ACL*. 48

Bo Pang, Lillian Lee, et al. 2008. Opinion mining and sentiment analysis. *Foundations and Trends® in Information Retrieval* 2(1–2):1–135. 160

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, pages 311–318. 80, 96, 109, 147, 154

Ankur P. Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *EMNLP*. xix, 26, 33, 35, 48, 51, 54, 57

Prashant Parikh. 2001. The use of language . 16

Adam Paszke, Sam Gross, Soumith Chintala, and Gregory Chanan. 2017. Pytorch. 154

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304* . 70, 124, 134

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *ACL*. 29

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Édouard Duchesnay. 2011a. Scikit-learn: Machine learning in Python. *JMLR* 12:2825–2830. 120

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, et al. 2011b. Scikit-learn: Machine learning in python. *Journal of machine learning research* 12(Oct):2825–2830. 151

Lotem Peled and Roi Reichart. 2017. Sarcasm sign: Interpreting sarcasm with sentiment based monolingual machine translation. *arXiv preprint arXiv:1704.06836* . 143, 159

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*. pages 1532–1543. 35, 38, 41, 55, 85, 108, 121, 126, 147, 151, 154, 162

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018a. Deep contextualized word representations. In *NAACL*. 25

Matthew E Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018b. Deep contextualized word representations. *arXiv preprint arXiv:1802.05365* . 38

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. 2019. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066* . 21

Maxime Peyrard. 2019a. A simple theoretical model of importance for summarization. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, pages 1059–1073. https://www.aclweb.org/anthology/P19-1101. 69, 136

Maxime Peyrard. 2019b. Studying summarization evaluation metrics in the appropriate scoring range. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Florence, Italy, pages 5093–5100. https://www.aclweb.org/anthology/P19-1502. 136

Peter Potash, Alexey Romanov, and Anna Rumshisky. 2017. Semeval-2017 task 6:# hashtag-wars: Learning a sense of humor. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*. pages 49–57. 159

Shrimai Prabhumoye, Yulia Tsvetkov, Ruslan Salakhutdinov, and Alan W. Black. 2018. Style

transfer through back-translation. In *ACL*. 144, 153

Daniel Preoţiuc-Pietro and Lyle Ungar. 2018. User-level race and ethnicity predictors from twitter text. In *Proceedings of the 27th International Conference on Computational Linguistics*. pages 1534–1545. 141, 142, 143, 157, 165

Ratish Puduppully, Li Dong, and Mirella Lapata. 2019. Data-to-text generation with content selection and planning. In *Proceedings of the AAAI Conference on Artificial Intelligence*. volume 33, pages 6908–6915. 67

Taivo Pungas. 2017. A dataset of english plaintext jokes. https://github.com/taivop/joke-dataset. 159

Ashequl Qadir, Michael Gamon, Patrick Pantel, and Ahmed Hassan Awadallah. 2016. Activity modeling in email. In *Proceedings of NAACL-HLT*. pages 1452–1462. 68

Huida Qiu, Yan Liu, Niranjan A Subrahmanya, and Weichang Li. 2012. Granger causality for time-series anomaly detection. In *Data Mining (ICDM), 2012 IEEE 12th International Conference on*. IEEE, pages 1074–1079. 66

Ella Rabinovich, Shachar Mirkin, Raj Nath Patel, Lucia Specia, and Shuly Wintner. 2016. Personalized machine translation: Preserving original author traits. *arXiv preprint arXiv:1610.05461* . 143

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. *URL https://s3-us-west-2. amazonaws. com/openai-assets/researchcovers/languageunsupervised/language understanding paper. pdf* . 4

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. *OpenAI Blog* 1(8):9. 3, 4, 65, 67, 88, 89, 92, 96

Azzurra Ragone, Katsiaryna Mirylenka, Fabio Casati, and Maurizio Marchese. 2011. A quantitative analysis of peer review. In *Proc. of ISSI*. 113, 123

Rajat Raina, Aria Haghighi, Christopher Cox, Jenny Finkel, Jeff Michels, Kristina Toutanova, Bill MacCartney, Marie-Catherine de Marneffe, Christopher D Manning, and Andrew Y Ng. 2005. Robust textual inference using diverse knowledge sources. In *1st PASCAL Recognition Textual Entailment Challenge Workshop*. 25

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250* . 1, 13, 21, 26

Sudha Rao and Joel Tetreault. 2018. Dear sir or madam, may i introduce the gyafc dataset: Corpus, benchmarks and metrics for formality style transfer. *arXiv preprint arXiv:1803.06535* . 143, 154, 159, 161

Hannah Rashkin, Eric Michael Smith, Margaret Li, and Y-Lan Boureau. 2019. Towards empathetic open-domain conversation models: A new benchmark and dataset. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. pages 5370–5381. 157

Siva Reddy, Danqi Chen, and Christopher D Manning. 2019. Coqa: A conversational question answering challenge. *Transactions of the Association for Computational Linguistics* 7:249–266. 14

Sravana Reddy and Kevin Knight. 2016. Obfuscating gender in social media writing. In *Proceedings of the First Workshop on NLP and Computational Social Science*. pages 17–26. 143, 144

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics. http://arxiv.org/abs/1908.10084. 2

Ehud Reiter and Robert Dale. 1997. Building applied natural language generation systems. *Natural Language Engineering* 3(1):57–87. 1, 14, 15

Ehud Reiter and Robert Dale. 2000. *Building natural language generation systems*. Cambridge university press. 66, 82

Drummond Rennie. 2016. Make peer review scientific: thirty years on from the first congress on peer review, drummond rennie reflects on the improvements brought about by research into the process–and calls for more. *Nature* 535(7610):31–34. 123

Kevin Reschke, Adam Vogel, and Daniel Jurafsky. 2013. Generating recommendation dialogs by extracting information from user reviews. In *ACL*. 68, 99

Mehwish Riaz and Roxana Girju. 2013. Toward a better understanding of causality between verbal events: Extraction and analysis of the causal power of verb-verb associations. In *Proceedings of the annual SIGdial Meeting on Discourse and Dialogue (SIGDIAL)*. Citeseer. 66, 71

Seán G Roberts and Tessa Verhoef. 2016. Double-blind reviewing at evolang 11 reveals gender bias. *Journal of Language Evolution* 1(2):163–167. 123

Alfredo Láinez Rodrigo and Luke de Oliveira. 2015. Sequential convolutional architectures for multi-sentence text classification cs224n-final project report . 159

Stuart Rose, Dave Engel, Nick Cramer, and Wendy Cowley. 2010. Automatic keyword extraction from individual documents. *Text Mining: Applications and Theory* pages 1–20. 91, 146

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A neural attention model for abstractive sentence summarization. *arXiv preprint arXiv:1509.00685* . 69, 134

Sylvia Weber Russell. 1976. Computer understanding of metaphorically used verbs. *American journal of computational linguistics* . 160

Roger C Schank and Robert P Abelson. 2013. *Scripts, plans, goals, and understanding: An inquiry into human knowledge structures*. Psychology Press. 65, 88, 91

Roy Schwartz, Maarten Sap, Ioannis Konstas, Leila Zilles, Yejin Choi, and Noah A Smith. 2017. The effect of different writing tasks on linguistic style: A case study of the roc story cloze task. *arXiv preprint arXiv:1702.01841* . 1

Abigail See, Peter J Liu, and Christopher D Manning. 2017. Get to the point: Summarization

with pointer-generator networks. *arXiv preprint arXiv:1704.04368* . 13, 67, 70, 94, 95, 124, 126, 134

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *ACL*. 108

Iulian Vlad Serban, Alessandro Sordoni, Ryan Lowe, Laurent Charlin, Joelle Pineau, Aaron C Courville, and Yoshua Bengio. 2017. A hierarchical latent variable encoder-decoder model for generating dialogues. In *AAAI*. pages 3295–3301. 67, 85, 95

Lei Sha, Sujian Li, Baobao Chang, and Zhifang Sui. 2016. Recognizing textual entailment via multi-task knowledge assisted lstm. In *Chinese Computational Linguistics and Natural Language Processing Based on Naturally Annotated Big Data*, Springer, pages 285–298. 25, 27, 29

Claude E Shannon. 1951. Prediction and entropy of printed English. *Bell Labs Technical Journal* 30(1):50–64. 147

Aakanksha Sharaff and Naresh Kumar Nagwani. 2016. Email thread identification using latent dirichlet allocation and non-negative matrix factorization based clustering techniques. *Journal of Information Science* 42(2):200–212. 68

Rebecca Sharp, Mihai Surdeanu, Peter Jansen, Peter Clark, and Michael Hammond. 2016. Creating causal embeddings for question answering with minimal supervision. *arXiv preprint arXiv:1609.08097* . 74

Tianxiao Shen, Tao Lei, Regina Barzilay, and Tommi Jaakkola. 2017. Style transfer from non-parallel text by cross-alignment. In *Advances in Neural Information Processing Systems*. pages 6830–6841. 143, 153, 154

Xiaoyu Shen, Jun Suzuki, Kentaro Inui, Hui Su, Dietrich Klakow, and Satoshi Sekine. 2019. Select and attend: Towards controllable content selection in text generation. *arXiv preprint arXiv:1909.04453* . 67, 89

Vivian S Silva, André Freitas, and Siegfried Handschuh. 2018. Recognizing and justifying text entailment through distributional navigation on definition graphs. In *AAAI*. 25

David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, et al. 2017. Mastering the game of go without human knowledge. *Nature* 550(7676):354. 106

Michael Silverstein. 2003. Indexical order and the dialectics of sociolinguistic life. *Language & communication* 23(3-4):193–229. 138

Eriks Sneiders. 2010. Automated email answering by text pattern matching. In *International Conference on Natural Language Processing*. Springer, pages 381–392. 68

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*. pages 1631–1642. 13, 160, 161

Alessandro Sordoni, Yoshua Bengio, Hossein Vahabi, Christina Lioma, Jakob Grue Simonsen, and Jian-Yun Nie. 2015. A hierarchical recurrent encoder-decoder for generative context-

aware query suggestion. In *Proceedings of the 24th ACM International on Conference on Information and Knowledge Management*. ACM, pages 553–562. 67, 85, 95, 96

Rupesh Kumar Srivastava, Klaus Greff, and Jürgen Schmidhuber. 2015. Highway networks. *arXiv preprint arXiv:1505.00387* . 55

Efstathios Stamatatos, Francisco Rangel, Michael Tschuggnall, Benno Stein, Mike Kestemont, Paolo Rosso, and Martin Potthast. 2018. Overview of pan 2018. In *International Conference of the Cross-Language Evaluation Forum for European Languages*. Springer, pages 267–285. 144

Gerard Steen. 2010. *A method for linguistic metaphor identification: From MIP to MIPVU*, volume 14. John Benjamins Publishing. 160, 161

Ilya Sutskever, Oriol Vinyals, and Quoc VV Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. pages 3104–3112. 30

Charles Sutton and Linan Gong. 2017. Popularity of arxiv.org within computer science. ArXiv:1710.05225. 115, 123

Judith A. Swan. 2002. The science of scientific writing. In *Book*. 81, 88

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075* . 82

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2017. A compare-propagate architecture with alignment factorization for natural language inference. *arXiv preprint arXiv:1801.00102* . 57

Joseph Tepperman, David Traum, and Shrikanth Narayanan. 2006. " yeah right": Sarcasm recognition for spoken dialogue systems. In *Ninth International Conference on Spoken Language Processing*. 159

Tijmen Tieleman and Geoffrey Hinton. 2012. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning* 4(2):26–31. 121

Andrew Tomkins, Min Zhang, and William D Heavlin. 2017. Single versus double blind reviewing at wsdm 2017. ArXiv:1702.00502. 123

Silvan S Tomkins. 1978. Script theory: Differential magnification of affects. In *Nebraska symposium on motivation*. University of Nebraska Press. 67, 81, 88, 91

Oren Tsur and Ari Rappoport. 2009. Revrank: A fully unsupervised algorithm for selecting the most helpful book reviews. In *Proc. of ICWSM*. 123

Yulia Tsvetkov, Manaal Faruqui, Wang Ling, Guillaume Lample, and Chris Dyer. 2015. Evaluation of word vector representations by subspace alignment. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. pages 2049–2054. 48

Peter D Turney. 2012. Domain and function: A dual-space model of semantic relations and compositions. *Journal of Artificial Intelligence Research* 44:533–585. 38

Peter D. Turney. 2013. Distributional semantics beyond words: Supervised learning of analogy and paraphrase. *TACL* 1:353–366. 38

Hanna K Ulatowska, Mari M Hayashi, Michael P Cannito, and Susan G Fleming. 1986. Disruption of reference in aging. *Brain and language* 28(1):24–41. 144

Chris van der Lee, Albert Gatt, Emiel van Miltenburg, Sander Wubben, and Emiel Krahmer. 2019. Best practices for the human evaluation of automatically generated text. In *Proceedings of the 12th International Conference on Natural Language Generation*. pages 355–368. 170

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*. pages 5998–6008. 126

Ben Verhoeven and Walter Daelemans. 2014. CLiPS Stylometry Investigation (CSI) corpus: A Dutch corpus for the detection of age, gender, personality, sentiment and deception in text. In *LREC 2014-NINTH INTERNATIONAL CONFERENCE ON LANGUAGE RESOURCES AND EVALUATION*. pages 3081–3085. 144, 150

Jesse Vig. 2019. A multiscale visualization of attention in the transformer model. *arXiv preprint arXiv:1906.05714* https://arxiv.org/abs/1906.05714. 91

Oriol Vinyals, Igor Babuschkin, Junyoung Chung, Michael Mathieu, Max Jaderberg, Wojciech M. Czarnecki, Andrew Dudzik, Aja Huang, Petko Georgiev, Richard Powell, Timo Ewalds, Dan Horgan, Manuel Kroiss, Ivo Danihelka, John Agapiou, Junhyuk Oh, Valentin Dalibard, David Choi, Laurent Sifre, Yury Sulsky, Sasha Vezhnevets, James Molloy, Trevor Cai, David Budden, Tom Paine, Caglar Gulcehre, Ziyu Wang, Tobias Pfaff, Toby Pohlen, Yuhuai Wu, Dani Yogatama, Julia Cohen, Katrina McKinney, Oliver Smith, Tom Schaul, Timothy Lillicrap, Chris Apps, Koray Kavukcuoglu, Demis Hassabis, and David Silver. 2019. AlphaStar: Mastering the Real-Time Strategy Game StarCraft II. https://deepmind.com/blog/alphastar-mastering-real-time-strategy-game-starcraft-ii/. 106

Marilyn A Walker, Jean E Fox Tree, Pranav Anand, Rob Abbott, and Joseph King. 2012. A corpus for research on deliberation and debate. In *LREC*. pages 812–817. 159

Samuel Walker. 1994. *Hate speech: The history of an American controversy*. U of Nebraska Press. 160

Byron C Wallace. 2015. Computational irony: A survey and new perspectives. *Artificial Intelligence Review* 43(4):467–483. 159

Byron C Wallace, Laura Kertz, Eugene Charniak, et al. 2014. Humans require context to infer ironic intent (so computers probably do, too). In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. pages 512–516. 159

E. Walsh, Michael W Rooney, Louis Appleby, and Greg Wilkinson. 2000. Open peer review: a randomised controlled trial. *The British journal of psychiatry : the journal of mental science* 176:47–51. 123

Alex Wang and Kyunghyun Cho. 2019. Bert has a mouth, and it must speak: Bert as a markov

random field language model. *arXiv preprint arXiv:1902.04094* . 92, 95

Qingyun Wang, Xiaoman Pan, Lifu Huang, Boliang Zhang, Zhiying Jiang, Heng Ji, and Kevin Knight. 2018. Describing a knowledge base. In *CoRR*. volume abs/1809.01797. 66

Shanshan Wang and Lei Zhang. 2017. CatGAN: Coupled adversarial transfer for domain generation. *CoRR* abs/1711.08904. 25

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017. Bilateral multi-perspective matching for natural language sentences. *arXiv preprint arXiv:1702.03814* . 26

Pontus Wärnestål. 2005. Modeling a dialogue strategy for personalized movie recommendations. In *Beyond Personalization Workshop*. pages 77–82. 68, 99

Amy Beth Warriner, Victor Kuperman, and Marc Brysbaert. 2013. Norms of valence, arousal, and dominance for 13,915 english lemmas. *Behavior research methods* 45(4):1191–1207. 142, 143, 157, 160

Joseph Weizenbaum. 1966. Eliza—a computer program for the study of natural language communication between man and machine. *Communications of the ACM* 9(1):36–45. 169

James B Wendt, Michael Bendersky, Lluis Garcia-Pueyo, Vanja Josifovski, Balint Miklos, Ivo Krka, Amitabh Saikia, Jie Yang, Marc-Allen Cartright, and Sujith Ravi. 2016. Hierarchical label propagation and discovery for machine generated email. In *Proceedings of the Ninth ACM International Conference on Web Search and Data Mining*. ACM, pages 317–326. 69

Lydia White. 1985. The "pro-drop" parameter in adult second language acquisition. *Language learning* 35(1):47–61. 144

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation* 39:165–210. 48

Ronald J Williams. 1992a. Simple statistical gradient-following algorithms for connectionist reinforcement learning. In *Reinforcement Learning*, Springer, pages 5–32. 34

Ronald J Williams. 1992b. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8(3-4):229–256. 107

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*. Association for Computational Linguistics, pages 347–354. 72

Sam Wiseman, Stuart M Shieber, and Alexander M Rush. 2017. Challenges in data-to-document generation. *arXiv preprint arXiv:1707.08052* . 63

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *ArXiv* abs/1910.03771. 95

Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the conference on empirical methods in natural language processing*. Association for Computational Linguistics,

pages 409–420. 69

Ledell Yu Wu, Adam Fisch, Sumit Chopra, Keith Adams, Antoine Bordes, and Jason Weston. 2018. Starspace: Embed all the things! In *AAAI*. 102

Diyi Yang, Alon Lavie, Chris Dyer, and Eduard H Hovy. 2015. Humor recognition and humor anchor extraction. In *EMNLP*. pages 2367–2376. 159

Zhilin Yang, Zihang Dai, Ruslan R. Salakhutdinov, and William W. Cohen. 2018. Breaking the softmax bottleneck: A high-rank rnn language model. In *ICLR*. 108

Zichao Yang, Diyi Yang, Chris Dyer, Xiaodong He, Alex Smola, and Eduard Hovy. 2016. Hierarchical attention networks for document classification. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 67, 82

Denis Yarats and Mike Lewis. 2018. Hierarchical text generation and planning for strategic dialogue. In *ICML*. 68

Wenpeng Yin, Dan Roth, and Hinrich Schütze. 2018. End-task oriented textual entailment via deep exploring inter-sentence interactions. In *ACL*. 57

Michael Miller Yoder. 2019. *Computational Models of Identity Presentation in Language*. PhD dissertation, Carnegie Mellon University. 139

Dani Yogatama, Fei Liu, and Noah A Smith. 2015. Extractive summarization by maximizing semantic volume. In *EMNLP*. pages 1961–1966. 69, 126, 127, 128

R Michael Young and Johanna D Moore. 1994. Dpocl: A principled approach to discourse planning. In *Proceedings of the Seventh International Workshop on Natural Language Generation*. Association for Computational Linguistics, pages 13–20. 67

David Zajic, Bonnie Dorr, and Richard Schwartz. 2004. Bbn/umd at duc-2004: Topiary. In *Proceedings of the HLT-NAACL 2004 Document Understanding Workshop, Boston*. pages 112–119. 69

Biao Zhang, Deyi Xiong, Jinsong Su, Qun Liu, Rongrong Ji, Hong Duan, and Min Zhang. 2016. Variational neural discourse relation recognizer. In *EMNLP*. 67

Saizheng Zhang, Emily Dinan, Jack Urbanek, Arthur Szlam, Douwe Kiela, and Jason Weston. 2018. Personalizing dialogue agents: I have a dog, do you have pets too? *arXiv preprint arXiv:1801.07243* . 111

Xiang Zhang, Junbo Zhao, and Yann LeCun. 2015. Character-level convolutional networks for text classification. In *Proc. of NIPS*. 121, 152

Yiheng Zhou, Yulia Tsvetkov, Alan W Black, and Zhou Yu. 2019. Augmenting non-collaborative dialog systems with explicit semantic and strategic dialog history. *arXiv preprint arXiv:1909.13425* . 170

Yukun Zhu, Ryan Kiros, Richard S. Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. *2015 IEEE International Conference on Computer Vision (ICCV)* pages 19–27. 84, 95