Deep Contextual Modeling for Natural Language Understanding, Generation, and Grounding

A Dissertation
Presented to the Faculty of the Graduate School
of
Yale University
in Partial Fulfillment of the Requirements for the Degree of
Doctor of Philosophy

by
Rui Zhang

Dissertation Director: Dragomir R. Radev

March, 2020

Abstract

Deep Contextual Modeling for Natural Language Understanding, Generation, and Grounding

Rui Zhang

2020

Natural language is a fundamental form of information and communication. In both human-human and human-computer communication, people reason about the context of text and world state to understand language and produce language response. Natural Language Processing aims to build intelligent systems that can understand language, generate language, and ground language in other forms such as formal programs.

This dissertation aims to present several deep neural network based systems that first understand the meaning of language grounded in various contexts where the language is used, and then generate effective language responses in different forms for information retrieval and human-computer communication. In particular, the following research objectives and challenges are addressed: (1) Deep Neural Modeling of Text Units. We present several end-to-end deep neural networks for (i) entity extraction and coreference resolution in documents, (ii) sentiment analysis and text classification for sentences and documents, and (iii) addressee and response selection for multi-turn multi-party conversations based on explicit representations for different discourse participants. (2) Text Summarization for Generating Email Subject Lines. We create the first dataset for this task and find that email subject line generation favors extremely abstractive summary, differentiating it from news headline generation or news single document summarization. For proper evaluation, we build a neural network to score the quality of an email subject given the email body. To summarize messages to short subjects with a high compression ratio, we combine the extractive and abstractive approaches and employ a multi-stage training strategy with supervised pretraining and reinforcement learning to optimize the email subject qual-

ity scores. (3) Neural Learning-to-rank for Low-Resource Cross-lingual Information Retrieval. We propose to combine evidence from different query and document translations by using cross-lingual word embeddings and deep relevance ranking models in a low-resource setting. By including the query likelihood retrieval score as an extra feature, the model effectively learns to rerank from only a few hundred relevance labels. In addition, by aligning word embedding spaces for multiple languages, the model can be directly applied under a zero-shot transfer setting when no training data is available for another language pair. (4) Multi-turn Text-to-SQL Generation by Language Grounding to Relational Databases. Generating SQL queries from user utterances is important to help people acquire information from databases. Furthermore, in real-world applications, users often access information in a multi-turn interaction with the system by asking a sequence of related questions. We propose an editing-based model for the cross-domain multi-turn text-to-SQL task. Observing that adjacent natural language questions are often linguistically dependent and their corresponding SQL queries tend to overlap, we utilize the interaction history by editing the previous predicted query to improve the generation quality. Moreover, to deal with complex table structures in different domains, we employ an utterance-table encoder and a table-aware decoder to incorporate the context of the user utterance and the table schema.

Acknowledgments

First and foremost, I give my greatest thanks to my brilliant advisor Professor Dragomir Radev for his tremendous support throughout my undergraduate years and the five years of my Ph.D. study. In the summer after my junior year, I sent an inquiry email to Professor Dragomir Radev about the possibility to join an undergraduate project, and he replied almost immediately even though I had never heard about “Natural Language Processing”. This started my research with Drago who constantly supports me with his patience, guidance, and insights. My Ph.D. and this thesis would not have been possible without him.

Second, I am also grateful to my advisor Professor Honglak Lee at the University of Michigan who has led me into the deep learning research area and helped me a lot with many of my initial research projects and papers. All of the work in this thesis is based on deep learning, which I all learned from Professor Honglak Lee.

Furthermore, I would also like to thank Professor Robert Frank and Professor John Lafferty for being on my qualifying exam and my thesis committee. Professor Robert Frank is extremely enthusiastic and knowledgeable and I always learned a lot from talking with him. It is also my great honor to have Professor John Lafferty, one of the most influential scientists in machine learning, on my committee. My work in this thesis follows many of his seminal papers. I am also grateful for Professor Kathleen McKeown and Professor Douglas W. Oard for their support on my job search and their incredible leadership in the huge cross-team effort on the MATERIAL project. I am also thankful to Professor Ed Durfee and Professor Michael Wellman for giving me the chance to do TA and supporting

me for my graduate school application and job search. I am really appreciated for the help and advice from Professor Marynel Vázquez on my job application.

I would like to thank my internship mentors Dr. Bing Xiang and Dr. Cícero Nogueira dos Santos at IBM, Dr. Joel Tetreault at Grammarly, and Dr. Yang Li at Google. I spent wonderful summers with you.

During my college and graduate school years, I feel lucky to have so many amazing mentors and collaborators: Professor Weikang Qian and Professor Sam Ro from Shanghai Jiao Tong University, Professor H. V. Jagadish, Professor Walter Lasecki, Professor Rada Mihalcea, Professor Jia Deng, Professor Steven Abney, Professor Jason Corso, Professor Jenna Wiens, Professor Semyon Meerkov, Dr. Donald Winsor, Yang Yang, Kihyuk Sohn, Yuting Zhang, Yi Zhang, Xinchen Yan, Kibok Lee, Sungryull Sohn, Seunghoon Hong, Ruben Villegas, Junhyuk Oh, Scott Reed, Xiaoshi Wang, Hongji Wang, Ben King, Rahul Jha, Meixing Dong, Steve Wilson from the University of Michigan, Cathy Finegan-Dollak, Lajanugen Logeswaran, Lazaros Polymenakos, Jonathan K. Kummerfeld, Nikita Bhutani, Ananda Narayan, Chang-Hong Hsu, Chongruo Wu, Yelin Kim, Janarthanan Rajendran, Sai R. Gouravajhala, Charlie Welch, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Reed Coke, Xiangyi Ye on the Sapphire project, Petra Galuščáková, Anton Ragni, Professor Mark Gales, Professor Philip Resnik, Professor Marine Carpuat, Professor Kenneth Heafield, Smaranda Muresan, Elena Zotkina, Chris Kedzie, Yanda Chen, Efsun Kayi, Suraj Nair, Han-Chin Shing, Joseph Barrow, Xing Niu, Weijia Xu, Ramy Eskander on the MATERIAL project, Xi Victoria Lin, Tianze Shi, Caiming Xiong, Richard Socher, Zifan Li, Zihan Li, Youxuan Jiang, Luyao Chen, Yuwen Zhang on the Text-to-SQL project, Nazneen Rajani on the ESPRIT project, Tao Yu, Javid Dadashkarimi, Alexander Fabbri, Irene Li, Yavuz Nuzumlali, Michihiro Yasunaga, Jungo Kasai, Yi Chern Tan, He Yang Er, Jeremy Weiss, Abhijit Gupta, Aadit Vyas, Kai Yang, Zilin Zhang, Dongxu Wang, Qingning Yao, Suyi Li, Bo Pang, Eric Xue, Sungrok Shim, Tao Chen, David Proctor, Vincent Zhang, Caitlin Westerfield, Garrett Bingham, Neha Verma, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan at LILY Lab. You together make my Ph.D. process a special one.

Finally, I want to thank my father, my mother, my brother, and all my family members for their love. Special thanks to Shaobo, for the time we spend together and the joy you bring to my life since we were high school classmates, and the best is yet to come.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Objectives and Challenges

Natural language is a fundamental form of information and communication. In both human-human and human-computer communication, people reason about the contexts of text and world state to understand the meaning of natural language and choose to produce what to say. There is a long semantic tradition of "Discourse Primacy" arguing that it is the discourse that has truth-conditions and information content and the meaning of all linguistic expressions contributes to the information of the sentences in which they occur (Kamp, 1981; Heim, 1982). This calls for the "Dynamic Semantics" (Yalcin, 2013) approach to articulating the meaning of language according to its context, which considers the compositional semantic value of a sentence as an operation of the information body for updating a certain kind of representation of the discourse. In communications with a background of mutually shared presuppositions (i.e., a common ground), the assertion of sentences (Stalnaker, 1978) serves to add the informational content determined by the sentence asserted in context and to update the common ground (Lewis, 1979).

Natural Language Processing aims to build intelligent systems that can understand language, generate language, and ground language in other forms. Different natural language problems in understanding, generation, and grounding require effective comprehension and usage of contexts. This dissertation aims to present several deep neural network based machine learning systems that first understand the meaning of language grounded in various contexts where the language is used, and then generate effective language responses in different forms for information request and human-computer communication. In particular, the following research objectives and challenges are addressed involving effective comprehension and usage of contexts for language understanding, generation, and grounding. We summarize the objectives in each chapter with the form of context as in Figure 1.1. The

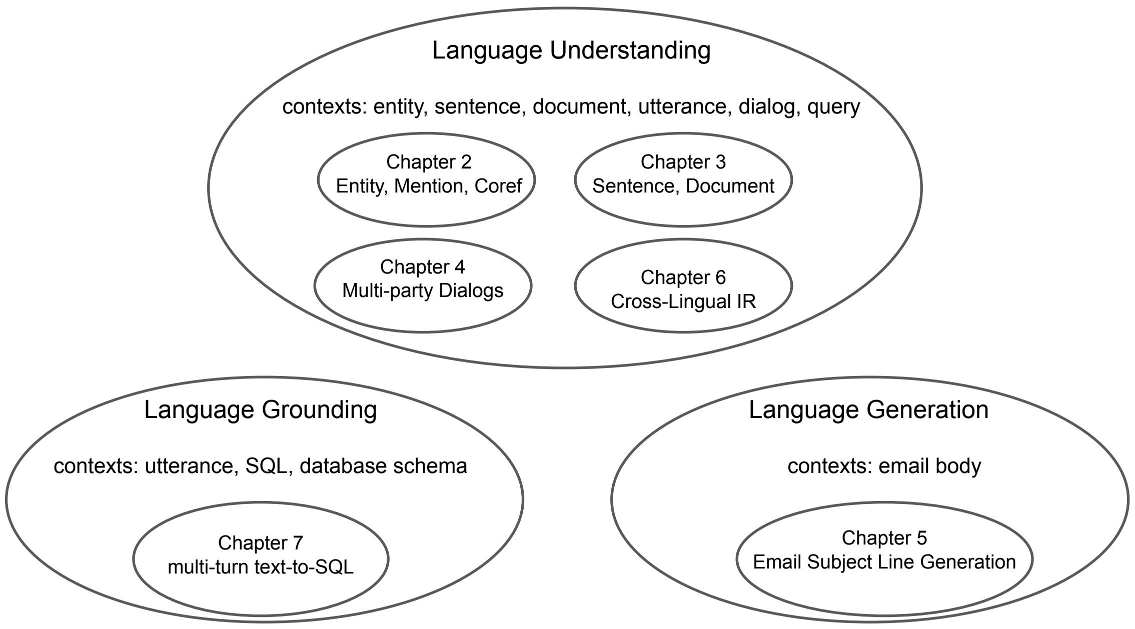


Figure 1.1: Contexts for different natural language problems in understanding, generation, and grounding.

challenges and solutions are inherently multi-disciplinary, spanning areas including natural language processing, deep learning, information retrieval, databases, and human-computer interaction.

Deep Neural Modeling of Text Units. Natural language consists of text units at different granularity levels, such as entity mentions, sentences, documents, and conversations.

Small text units make up larger ones to convey meaning, and the semantics of text units depend on each other. Therefore, modeling text units and their dependency relationships is fundamental for natural language understanding. In particular, I am interested in how deep learning models can help us better understand and utilize the dependency relationships among text units in different NLP tasks. In particular, this thesis aims to use end-to-end deep neural networks for (1) entity extraction and coreference resolution in documents, (2) sentiment analysis and text classification for sentences and documents, and (3) addressee and response selection for multi-turn multi-party conversations where we would like to build explicit distinct representations for different discourse participants (Portner, 2007).

Text Summarization for Generating Email Subject Lines. Text summarization aims to produce fluent and coherent synopses covering salient information in documents. Recently, neural methods have shown promising results for both extractive and abstractive approaches. However, most neural summarization work focuses on the single-document setting in the news domain and relies on huge amounts of training data, and much less effort has been made on summarization in other settings or domains with limited amounts of data such as multi-document news summarization, scientific article summarization incorporating citation information, and emails. Furthermore, while email is a ubiquitous form of communication and there have been several research tracks around email usage, no previous work has focused on email subject line generation.

An email message consists of two main elements: an *email subject line* and an *email body*. The subject line should tell what the email body is about and what the sender wants to convey. Table 1.1 shows an email body with three possible subject lines. Compared with news headline generation or news single document summarization, generating email subjects is more challenging because email subjects are generally much shorter and the dataset contains far fewer training examples. This requires systems that can summarize with a high compression ratio and can be trained in a data-efficient manner. Furthermore,

Email Body: Hi All, I would be grateful if you could get to me today via email a job description for your current role. I would like to get this to the immigration attorneys so that they can finalise the paperwork in preparation for INS filing once the UBS deal is signed. Kind regards,
Subject 1: Current Job Description Needed *(COMMENT: This is good because it is both informative and succinct.)*
Subject 2: Job Description *(COMMENT: This is okay but not informative enough.)*
Subject 3: Request *(COMMENT: This is bad because it does not contain any specific information about the request.)*

Table 1.1: An email with three possible subject lines.

the community also lacks proper metrics for this task beyond ROUGE and METEOR which simply measure n-gram overlappings with references.

Low-Resource Cross-lingual Information Retrieval. Cross-lingual Information Retrieval (CLIR) is the task of ranking documents in one language for responsiveness to a user query in another language. As multilingual documents become more accessible, CLIR is increasingly important whenever relevant information is in other languages. Traditional CLIR systems consist of two components: machine translation and monolingual information retrieval. This approach first solves the translation problem and then performs retrieval in a monolingual setting. However, while conceptually simple, the performance of this modular approach is fundamentally limited by the quality of machine translation.

Recently, many deep neural learning-to-rank models have shown promising results in monolingual information retrieval. They learn a scoring function directly from the relevance label of query-document pairs. However, applying these techniques to CLIR has been challenging, primarily for two reasons. First, when queries and documents are in different languages, it is not clear how to measure their similarity in word embedding representation space. Furthermore, deep neural networks need a large amount of training data to achieve decent performance. Annotations are prohibitively expensive for low-resource language pairs in our cross-lingual case.

Multi-turn Text-to-SQL Generation by Language Grounding to Relational Databases. Generating SQL queries from user utterances is important to help people acquire information from databases. Such a text-to-SQL semantic parsing system bridges the data and the user through an intelligent natural language interface, greatly promoting the possibility and efficiency of information access for many users besides database experts. Furthermore, in real-world applications, users often access information in a multi-turn interaction with the system by asking a sequence of related questions. The users may explicitly refer to or omit previously mentioned entities and constraints, and may introduce refinements, additions, or substitutions to what has already been said. This requires text-to-SQL systems to process context information to synthesize correct SQL queries.

1.2 Contributions

The contributions of this thesis are summarized as follows.

Deep Neural Models for Entities, Sentences, Documents, and Multi-Turn Multi-Party Dialogs. Chapter 2, Chapter 3, and Chapter 4 present several deep neural networks for understanding text units including entities, sentences and documents, and multi-turn multi-party dialogs (Zhang et al., 2016, 2018b,a; Finegan-Dollak et al., 2016).

First, Chapter 2 describes an end-to-end neural network for coreference resolution by joint mention detection and mention clustering. Coreference resolution aims to identify in a text all mentions that refer to the same real-world entity. The state-of-the-art end-to-end neural coreference model considers all text spans in a document as potential mentions and learns to link an antecedent for each possible mention. In this chapter, we propose to improve the end-to-end coreference resolution system by (1) using a biaffine attention model to get antecedent scores for each possible mention, and (2) jointly optimizing the mention detection accuracy and the mention clustering log-likelihood given the mention cluster labels. Our model achieves the state-of-the-art performance on the CoNLL-2012

Shared Task English test set.

Then, in Chapter 3, we present Dependency Sensitive Convolutional Neural Networks (DSCNN) as a general-purpose classification system for both sentences and documents. The goal of sentence and document modeling is to accurately represent the meaning of sentences and documents for various Natural Language Processing tasks. DSCNN hierarchically builds textual representations by processing pretrained word embeddings via Long Short-Term Memory networks and subsequently extracting features with convolution operators. Compared with existing recursive neural models with tree structures, DSCNN does not rely on parsers and expensive phrase labeling, and thus is not restricted to sentence-level tasks. Moreover, unlike other CNN-based models that analyze sentences locally by sliding windows, our system captures both the dependency information within each sentence and relationships across sentences in the same document. Experiment results demonstrate that our approach is achieving state-of-the-art performance on several tasks, including sentiment analysis, question type classification, and subjectivity classification.

To model the complexity of multi-party dialogues, Chapter 4 presents the Speaker Interaction Recurrent Neural Network (SI-RNN). Understanding multi-party conversations is challenging because of complex speaker interactions: multiple speakers exchange messages with each other, playing different roles (sender, addressee, observer), and these roles vary across turns. In recent years, there has been promising progress in using deep neural networks for utterance understanding and dialog response generation. However, these models are single-turn responding machines and thus still are limited to short contexts with only two speakers, yet relatively less effort has been devoted to modeling multi-party multi-turn dialogs. SI-RNN uses its dialog encoder to maintain speaker embeddings in a role-sensitive way. Speaker embeddings are updated in different GRU-based units based on their roles (sender, addressee, observer). Furthermore, since the addressee and response are mutually dependent, SI-RNN models the conditional probability (of an addressee given the response and vice versa) and selects the addressee and response pair by maximizing

thread id	Sender	Addressee	Utterance
1	codepython	wafflejock	thanks
1	wafflejock	codepython	yup np
2	wafflejock	theoletom	you can use sudo apt-get install packagename – reinstall, to have apt-get install reinstall some package/metapackage and redo the configuration for the program.
3	codepython	-	i installed ubuntu on a separate external drive. now when i boot into mac, the external drive does not show up as bootable. the blue light is on. any ideas?
4	Guest54977	-	hi. wondering who knows where an ubuntu backup can be retrieved from.
2	theoletom	wafflejock	it's not a program. it's a desktop environment.
4	Guest54977	-	did some searching on my system and googling, but couldn't find an answer
2	theoletom	-	be a trace of it left yet there still is.
5	releaf	-	what's your opinion on a $500 laptop that will be a dedicated ubuntu machine?
3	codepython	-	my usb stick shows up as bootable (efi) when i boot my mac. but not my external hard drive on which i just installed ubuntu. how do i make it bootable from mac hardware?
3	Jordan_U	codepython	did you install ubuntu to this external drive from a different machine?
5	Umeaboy	releaf	what country you from?
5	wafflejock		
Model Prediction		**Addressee**	**Response**
SI-RNN		⋆ releaf	⋆ there are a few ubuntu dedicated laptop providers like umeaboy is asking depends on where you are

Table 1.2: An example of addressee and response selection in Ubuntu IRC multi-party dialog. My SI-RNN engages in a new sub-conversation by suggesting a solution to "releaf" about Ubuntu dedicated laptops. ⋆ denotes the ground-truth.

the joint probability. On the Ubuntu IRC benchmark, SI-RNN significantly improves the addressee and response selection performance by 10% accuracy, particularly in complex conversations with many speakers and with responses to distant messages many turns in the past. Our model output for a task example is given in Table 1.2.

Email Subject Line Generation by Optimizing Quality Estimation Scores. My solution involves both an evaluation metric and a data-efficient model. First, to properly evaluate the subject, I built a neural network for Email Subject Quality Estimator (ESQE) to score the quality of an email subject given the email body. Statistical analysis showed that ESQE has a higher correlation with human evaluation than metrics based on n-gram matching. Next, to summarize messages to short subjects with a high compression ratio, I combined extractive and abstractive approaches. My model first used an extractor to select multiple sentences from the input email body, capturing salient information such as named entities and dates. Then it used an abstractor to rewrite multiple selected sentences into a succinct subject line while preserving key information. Furthermore, to make use of limited amounts of training data with only about 10k examples, I used a two-stage training strategy instead of an end-to-end fashion. At the first stage, I created proxy sentence labels by checking the word overlap between the subject and the body sentence. The extractor and the abstractor are separately trained using supervised cross-entropy loss. The second step is to train the extractor in a reinforcement learning framework to directly optimize the ESQE metric. Both automatic metrics (ROUGE and ESQE) and human evaluations (Informative and Fluent) demonstrated that our method outperformed other state-of-the-art summarization models and approached human-level quality.

Neural Learning-to-rank for Low-Resource Cross-lingual Information Retrieval. Motivated by these observations, I proposed a deep relevance ranking model to combine different translations by using cross-lingual word embeddings in a low-resource setting. As shown in Figure 1.2, the model first translates queries and documents and then uses four components to match them in both the source and target language. The final relevance score adds up all components to combine complementary evidence from different translations. Each component is implemented as a term interaction network that learns relevance scores from a similarity matrix of each pair of a query term and a document term. To measure the similarity of two words in different languages, I built cross-lingual embeddings by aligning monolingual embeddings onto one shared space. Furthermore, to deal with small amounts of training data, I first performed query likelihood retrieval and included the score as an extra feature in the model. In this way, the model effectively learns to rerank from only a few hundred relevance labels. In addition, by aligning word embedding spaces for multiple languages, the model can be directly applied under a zero-shot transfer setting when no training data is available for another language pair. On the MATERIAL CLIR dataset with three language pairs (English to Swahili, English to Tagalog, and English to Somali), the model outperformed other translation-based query likelihood retrieval models and state-of-the-art monolingual deep relevance ranking approaches by 2%-4% mAP (mean average precision) scores.

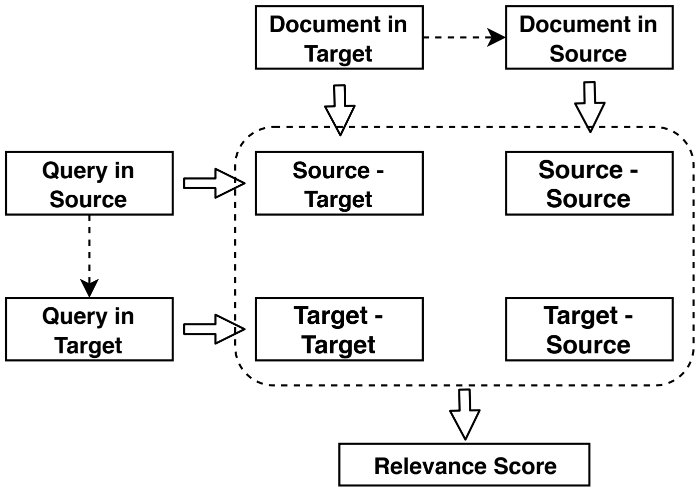
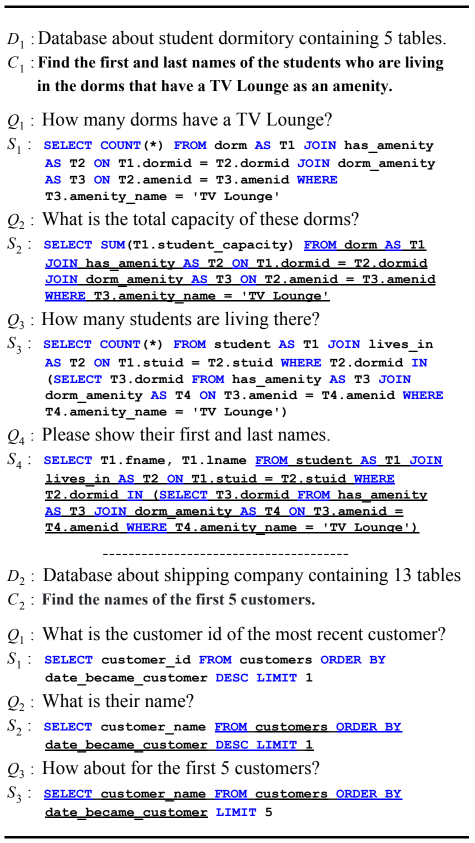


Figure 1.2: Deep Cross-lingual Relevance Ranking with Bilingual Query and Document Representation.

Editing-based SQL Query Generation for Cross-Domain Multi-turn Questions. To advance the state-of-the-art in this field, I contributed to two multi-turn text-to-SQL data sets with Tao Yu and other collaborators: (1) SParC (Yu et al., 2019b) (Figure 1.3) for cross-domain **S**emantic **Par**sing in **C**ontext which contains 4,298 unique multi-turn question sequences, comprised of 12k+ questions annotated with SQL queries, and (2) CoSQL (Yu et al., 2019a) (Figure 1.4) for building database-querying dialogue systems, which consists of 30k+ turns with 10k+ annotated SQL queries obtained from a Wizard-of-Oz collection of 3k dialogues. Both of them are built on top of our Spider dataset (Yu et al., 2018c), the largest cross-domain context-independent text-to-SQL dataset available in the field, and thus span 200 complex databases over 138 domains. The large number of domains provide rich contextual phenomena and thematic relations between the questions, which general-



D_1 : Database about student dormitory containing 5 tables.

C_1 : **Find the first and last names of the students who are living in the dorms that have a TV Lounge as an amenity.**

Q_1 : How many dorms have a TV Lounge?

S_1 :
```
SELECT COUNT(*) FROM dorm AS T1 JOIN has_amenity
AS T2 ON T1.dormid = T2.dormid JOIN dorm_amenity
AS T3 ON T2.amenid = T3.amenid WHERE
T3.amenity_name = 'TV Lounge'
```

Q_2 : What is the total capacity of these dorms?

S_2 :
```
SELECT SUM(T1.student_capacity) FROM dorm AS T1
JOIN has_amenity AS T2 ON T1.dormid = T2.dormid
JOIN dorm_amenity AS T3 ON T2.amenid = T3.amenid
WHERE T3.amenity_name = 'TV Lounge'
```

Q_3 : How many students are living there?

S_3 :
```
SELECT COUNT(*) FROM student AS T1 JOIN lives_in
AS T2 ON T1.stuid = T2.stuid WHERE T2.dormid IN
(SELECT T3.dormid FROM has_amenity AS T3 JOIN
dorm_amenity AS T4 ON T3.amenid = T4.amenid WHERE
T4.amenity_name = 'TV Lounge')
```

Q_4 : Please show their first and last names.

S_4 :
```
SELECT T1.fname, T1.lname FROM student AS T1 JOIN
lives_in AS T2 ON T1.stuid = T2.stuid WHERE
T2.dormid IN (SELECT T3.dormid FROM has_amenity
AS T3 JOIN dorm_amenity AS T4 ON T3.amenid =
T4.amenid WHERE T4.amenity_name = 'TV Lounge')
```

D_2 : Database about shipping company containing 13 tables

C_2 : **Find the names of the first 5 customers.**

Q_1 : What is the customer id of the most recent customer?

S_1 :
```
SELECT customer_id FROM customers ORDER BY
date_became_customer DESC LIMIT 1
```

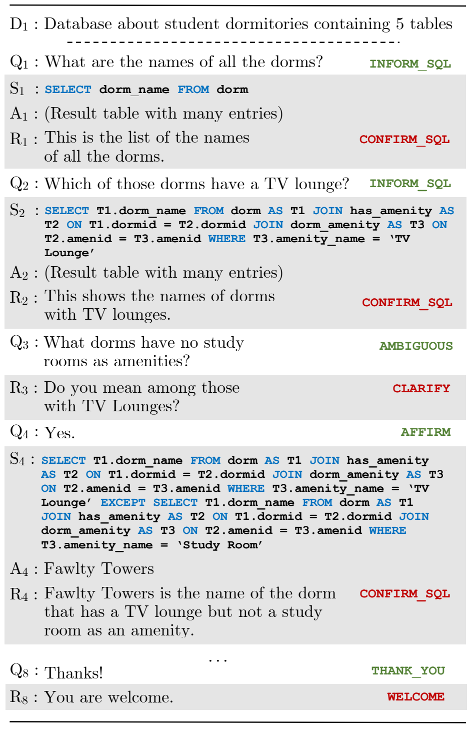
Q_2 : What is their name?

S_2 :
```
SELECT customer_name FROM customers ORDER BY
date_became_customer DESC LIMIT 1
```

Q_3 : How about for the first 5 customers?

S_3 :
```
SELECT customer_name FROM customers ORDER BY
date_became_customer LIMIT 5
```

Figure 1.3: Two question sequences from SParC. Questions (Q_i) in each sequence query a database (D_m), obtaining information sufficient to complete the interaction goal (C_m). Each question is annotated with a SQL query (S_i). SQL segments from the interaction context are underlined.



D_1 : Database about student dormitories containing 5 tables

Q_1 : What are the names of all the dorms? INFORM_SQL

S_1 : `SELECT dorm_name FROM dorm`

A_1 : (Result table with many entries)

R_1 : This is the list of the names of all the dorms. CONFIRM_SQL

Q_2 : Which of those dorms have a TV lounge? INFORM_SQL

S_2 :
```
SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity AS
T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3 ON
T2.amenid = T3.amenid WHERE T3.amenity_name = ‘TV
Lounge’
```

A_2 : (Result table with many entries)

R_2 : This shows the names of dorms with TV lounges. CONFIRM_SQL

Q_3 : What dorms have no study rooms as amenities? AMBIGUOUS

R_3 : Do you mean among those with TV Lounges? CLARIFY

Q_4 : Yes. AFFIRM

S_4 :
```
SELECT T1.dorm_name FROM dorm AS T1 JOIN has_amenity
AS T2 ON T1.dormid = T2.dormid JOIN dorm_amenity AS T3
ON T2.amenid = T3.amenid WHERE T3.amenity_name = ‘TV
Lounge’ EXCEPT SELECT T1.dorm_name FROM dorm AS T1
JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN
dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE
T3.amenity_name = ‘Study Room’
```

A_4 : Fawlty Towers

R_4 : Fawlty Towers is the name of the dorm that has a TV lounge but not a study room as an amenity. CONFIRM_SQL

. . .

Q_8 : Thanks! THANK_YOU

R_8 : You are welcome. WELCOME

Figure 1.4: A dialog from CoSQL. Gray boxes separate the user inputs (Q_i) querying the database (D_i) from the SQL queries (S_i), returned answers (A_i), and expert responses (R_i). Users send an input to the expert, who writes the corresponding SQL query (only seen by the expert) if possible and sends an answer and response description back. Dialogue acts are on the right-hand side (e.g., Q_3 is “ambiguous” and R_3 is “clarify”).

purpose natural language interfaces to databases have to address. In addition, it enables us to test the generalization of the trained systems to unseen databases and domains. We are actively maintaining the datasets and leaderboards of our Text-to-SQL Challenge Series including Spider (https://yale-lily.github.io/spider), SParC (https://yale-lily.github.io/sparc), and CoSQL (https://yale-lily.github.io/cosql).

Furthermore, I proposed an editing-based model for our cross-domain multi-turn text-to-SQL task. Based on the observation that adjacent natural language questions are often linguistically dependent and their corresponding SQL queries tend to overlap, I utilized the interaction history by editing the previous predicted query to improve the generation quality. This editing mechanism views SQL as sequences and reuses generation results at the token level in a simple manner. It is flexible to change individual tokens and robust to error propagation. Moreover, to deal with complex table structures in different domains, I employed an utterance-table encoder and a table-aware decoder to incorporate the context of the user utterance and the table schema. Experimental results on SParC showed that by generating from the previous query, the model delivered an improvement of 7% question match accuracy and 11% interaction match accuracy over the previous state-of-the-art.

1.3 Outline

The list of chapters in this thesis is based on the following publications:

- **Chapter 2: Neural Coreference Resolution with Deep Biaffine Attention by Joint Mention Detection and Mention Clustering**. In The 56th Annual Meeting of the Association for Computational Linguistics (ACL), 2018.

- **Chapter 3: Dependency Sensitive Convolutional Neural Networks for Modeling Sentences and Documents**. In The 15th Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT), 2016.

- **Chapter 4: Addressee and Response Selection in Multi-Party Conversations with Speaker Interaction RNNs**. In The Thirty-Second AAAI Conference on Artificial Intelligence (AAAI), 2018.

- **Chapter 5: This Email Could Save Your Life: Introducing the Task of Email Subject Line Generation**. In The 57th Annual Meeting of the Association for Computational Linguistics (ACL), 2019.

- **Chapter 6: Improving Low-Resource Cross-lingual Document Retrieval by Reranking with Deep Bilingual Representations**. In The 57th Annual Meeting of the Association for Computational Linguistics (ACL), 2019.

- **Chapter 7: Editing-based SQL Query Generation for Cross-Domain Context-Dependent Questions**. In The 2019 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2019.

Chapter 2

Neural Coreference Resolution by Joint Mention Detection and Mention Clustering

Coreference resolution aims to identify in a text all mentions that refer to the same real-world entity. The state-of-the-art end-to-end neural coreference model considers all text spans in a document as potential mentions and learns to link an antecedent for each possible mention. In this chapter, we propose to improve the end-to-end coreference resolution system by (1) using a biaffine attention model to get antecedent scores for each possible mention, and (2) jointly optimizing the mention detection accuracy and the mention clustering log-likelihood given the mention cluster labels. Our model achieves the state-of-the-art performance on the CoNLL-2012 Shared Task English test set.

This chapter is based on Zhang et al. (2018b).

2.1 Introduction

End-to-end coreference resolution is the task of identifying and grouping *mentions* in a text such that all mentions in a cluster refer to the same entity. An example is given below (Björkelund and Kuhn, 2014) where mentions for two entities are labeled in two clusters:

> [Drug Emporium Inc.]$_{a1}$ said [Gary Wilber]$_{b1}$ was named CEO of [this drug-store chain]$_{a2}$. [He]$_{b2}$ succeeds his father, Philip T. Wilber, who founded [the company]$_{a3}$ and remains chairman. Robert E. Lyons III, who headed the [company]$_{a4}$'s Philadelphia region, was appointed president and chief operating officer, succeeding [Gary Wilber]$_{b3}$.

Many traditional coreference systems, either rule-based (Haghighi and Klein, 2009; Lee et al., 2011) or learning-based (Bengtson and Roth, 2008; Fernandes et al., 2012; Durrett and Klein, 2013; Björkelund and Kuhn, 2014), usually solve the problem in two separate stages: (1) a mention detector to propose entity mentions from the text, and (2) a coreference resolver to cluster proposed mentions. At both stages, they rely heavily on complicated, fine-grained, conjoined features via heuristics. This pipeline approach can cause cascading errors, and in addition, since both stages rely on a syntactic parser and complicated hand-craft features, it is difficult to generalize to new data sets and languages.

Very recently, Lee et al. (2017) proposed the first state-of-the-art end-to-end neural coreference resolution system. They consider all text spans as potential mentions and therefore eliminate the need of carefully hand-engineered mention detection systems. In addition, thanks to the representation power of pre-trained word embeddings and deep neural networks, the model only uses a minimal set of hand-engineered features (speaker ID, document genre, span distance, span width).

The core of the end-to-end neural coreference resolver is the scoring function to compute the mention scores for all possible spans and the antecedent scores for a pair of spans. Furthermore, one major challenge of coreference resolution is that most mentions in the

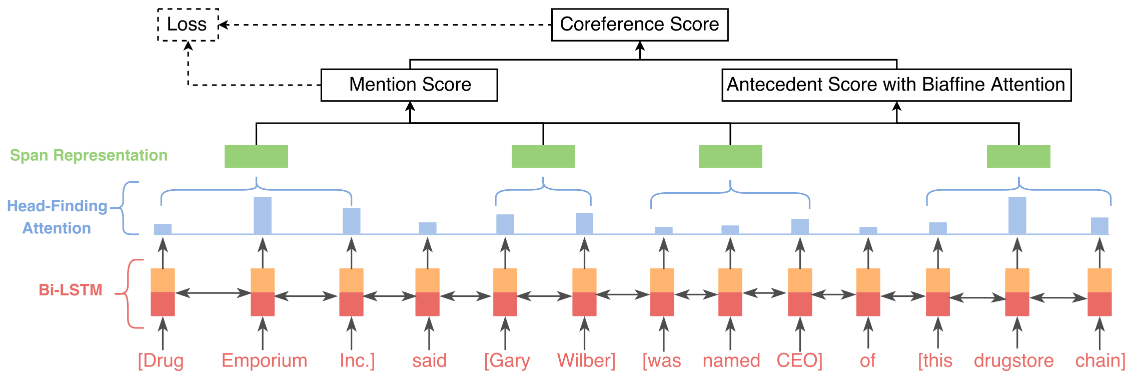


Figure 2.1: Model architecture. We consider all text spans up to 10-word length as possible mentions. For brevity, we only show three candidate antecedent spans ("Drug Emporium Inc.", "Gary Wilber", "was named CEO") for the current span "this drugstore chain".

document are singleton or non-anaphoric, i.e., not coreferent with any previous mention (Wiseman et al., 2015). Since the data set only have annotations for mention clusters, the end-to-end coreference resolution system needs to detect mentions, detect anaphoricity, and perform coreference linking. Therefore, research questions still remain on good designs of the scoring architecture and the learning strategy for both mention detection and antecedent scoring given only the gold cluster labels.

To this end, we propose to use a biaffine attention model instead of pure feed forward networks to compute antecedent scores. Furthermore, instead of training only to maximize the marginal likelihood of gold antecedent spans, we jointly optimize the mention detection accuracy and the mention clustering log-likelihood given the mention cluster labels. We optimize mention detection loss explicitly to extract mentions and also perform anaphoricity detection.

We evaluate our model on the CoNLL-2012 English data set and achieve new state-of-the-art performances of 67.8% F1 score using a single model and 69.2% F1 score using a 5-model ensemble.

2.2 Task Formulation

In end-to-end coreference resolution, the input is a document D with T words, and the output is a set of mention clusters each of which refers to the same entity. A possible *span* is an N-gram within a single sentence. We consider all possible spans up to a predefined maximum width. To impose an ordering, spans are sorted by the start position START(i) and then by the end position END(i). For each span i the system needs to assign an antecedent a_i from all preceding spans or a dummy antecedent ϵ: $a_i \in \{\epsilon, 1, \ldots, i-1\}$. If a span j is a true antecedent of the span i, then we have $a_i = j$ and $1 \leq j \leq i-1$. The dummy antecedent ϵ represents two possibilities: (1) the span i is not an entity mention, or (2) the span i is an entity mention but not coreferent with any previous span. Finally, the system groups mentions according to coreference links to form the mention clusters.

2.3 Model

Figure 2.1 illustrates our model. We adopt the same span representation approach as in Lee et al. (2017) using bidirectional LSTMs and a head-finding attention. Thereafter, a feed forward network produces scores for spans being entity mentions. For antecedent scoring, we propose a biaffine attention model (Dozat and Manning, 2017) to produce distributions of possible antecedents. Our training data only provides gold mention cluster labels. To make best use of this information, we propose to jointly optimize the mention scoring and antecedent scoring in our loss function.

Span Representation Suppose the current sentence of length L is $[w_1, w_2, \ldots, w_L]$, we use $\mathbf{w}_t$ to denote the concatenation of fixed pretrained word embeddings and CNN character embeddings (dos Santos and Zadrozny, 2014) for word w_t. Bidirectional LSTMs

	MUC			B^3			CEAF_{ϕ_4}			
	P	R	F1	P	R	F1	P	R	F1	Avg. F1
Our work (5-model ensemble)	82.1	73.6	77.6	73.1	62.0	67.1	67.5	59.0	62.9	**69.2**
Lee et al. (2017) (5-model ensemble)	81.2	73.6	77.2	72.3	61.7	66.6	65.2	60.2	62.6	68.8
Our work (single model)	79.4	73.8	76.5	69.0	62.3	65.5	64.9	58.3	61.4	**67.8**
Lee et al. (2017) (single model)	78.4	73.4	75.8	68.6	61.8	65.0	62.7	59.0	60.8	67.2
Clark and Manning (2016a)	79.2	70.4	74.6	69.9	58.0	63.4	63.5	55.5	59.2	65.7
Clark and Manning (2016b)	79.9	69.3	74.2	71.0	56.5	63.0	63.8	54.3	58.7	65.3
Wiseman et al. (2016)	77.5	69.8	73.4	66.8	57.0	61.5	62.1	53.9	57.7	64.2
Wiseman et al. (2015)	76.2	69.3	72.6	66.2	55.8	60.5	59.4	54.9	57.1	63.4
Fernandes et al. (2014)	75.9	65.8	70.5	77.7	65.8	71.2	43.2	55.0	48.4	63.4
Clark and Manning (2015)	76.1	69.4	72.6	65.6	56.0	60.4	59.4	53.0	56.0	63.0
Martschat and Strube (2015)	76.7	68.1	72.2	66.1	54.2	59.6	59.5	52.3	55.7	62.5
Durrett and Klein (2014)	72.6	69.9	71.2	61.2	56.4	58.7	56.2	54.2	55.2	61.7
Björkelund and Kuhn (2014)	74.3	67.5	70.7	62.7	55.0	58.6	59.4	52.3	55.6	61.6
Durrett and Klein (2013)	72.9	65.9	69.2	63.6	52.5	57.5	54.3	54.4	54.3	60.3

Table 2.1: Experimental results on the CoNLL-2012 English test set. The F1 improvements are statistical significant with $p < 0.05$ under the paired bootstrap resample test (Koehn, 2004) compared with Lee et al. (2017).

(Hochreiter and Schmidhuber, 1997) recurrently encode each w_t:

$$\begin{aligned}
\overrightarrow{\mathbf{h}}_t &= \text{LSTM}^{\text{forward}}(\overrightarrow{\mathbf{h}}_{t-1}, \mathbf{w}_t) \\
\overleftarrow{\mathbf{h}}_t &= \text{LSTM}^{\text{backward}}(\overleftarrow{\mathbf{h}}_{t+1}, \mathbf{w}_t) \\
\mathbf{h}_t &= [\overrightarrow{\mathbf{h}}_t, \overleftarrow{\mathbf{h}}_t]
\end{aligned} \tag{2.1}$$

Then, the head-finding attention computes a score distribution over different words in a span s_i:

$$\begin{aligned}
\alpha_t &= \mathbf{v}_\alpha^\intercal \text{FFNN}_\alpha(\mathbf{h}_t) \\
s_{i,t} &= \frac{\exp(\alpha_t)}{\sum\limits_{k=\text{START}(i)}^{\text{END}(i)} \exp(\alpha_k)} \\
\mathbf{w}_i^{\text{head-att}} &= \sum_{t=\text{START}(i)}^{\text{END}(i)} s_{i,t}\mathbf{w}_t
\end{aligned} \tag{2.2}$$

where FFNN is a feed forward network outputting a vector.

Effective span representations encode both contextual information and internal structure of spans. Therefore, we concatenate different vectors, including a feature vector $\phi(i)$

for the span size, to produce the span representation $\mathbf{s}_i$ for s_i:

$$\mathbf{s}_i = [\mathbf{h}_{\text{START}(i)}, \mathbf{h}_{\text{END}(i)}, \mathbf{w}_i^{\text{head-att}}, \phi(i)] \tag{2.3}$$

Mention Scoring The span representation is input to a feed forward network which measures if it is an entity mention using a score $m(i)$:

$$m(i) = \mathbf{v}_{\text{m}}^{\mathsf{T}} \text{FFNN}_{\text{m}}(\mathbf{s}_i) \tag{2.4}$$

Since we consider all possible spans, the number of spans is $O(T^2)$ and the number of span pairs is $O(T^4)$. Due to computation efficiency, we prune candidate spans during both inference and training. We keep λT spans with highest mention scores.

Biaffine Attention Antecedent Scoring Consider the current span s_i and its previous spans s_j ($1 \leq j \leq i-1$), we propose to use a biaffine attention model to produce scores $c(i,j)$:

$$\begin{aligned} \hat{\mathbf{s}}_{\text{i}} &= \text{FFNN}_{\text{anaphora}}(\mathbf{s}_{\text{i}}) \\ \hat{\mathbf{s}}_{\text{j}} &= \text{FFNN}_{\text{antecedent}}(\mathbf{s}_{\text{j}}), 1 \leq j \leq i-1 \\ c(i,j) &= \hat{\mathbf{s}}_{\text{j}}^{\mathsf{T}} \mathbf{U}_{\text{bi}} \hat{\mathbf{s}}_{\text{i}} + \mathbf{v}_{\text{bi}}^{\mathsf{T}} \hat{\mathbf{s}}_{\text{i}} \end{aligned} \tag{2.5}$$

$\text{FFNN}_{\text{anaphora}}$ and $\text{FFNN}_{\text{antecedent}}$ reduce span representation dimensions and only keep information relevant to coreference decisions. Compared with the traditional FFNN approach in Lee et al. (2017), biaffine attention directly models both the compatibility of s_i and s_j by $\hat{\mathbf{s}}_{\text{j}}^{\mathsf{T}} \mathbf{U}_{\text{bi}} \hat{\mathbf{s}}_{\text{i}}$ and the prior likelihood of s_i having an antecedent by $\mathbf{v}_{\text{bi}}^{\mathsf{T}} \hat{\mathbf{s}}_{\text{i}}$.

Inference The final coreference score $s(i,j)$ for span s_i and span s_j consists of three terms: (1) if s_i is a mention, (2) if s_j is a mention, (3) if s_j is an antecedent for s_i. Furthermore, for dummy antecedent ϵ, we fix the final score to be 0:

$$s(i,j) = \begin{cases} m(i) + m(j) + c(i,j), & j \neq \epsilon \\ 0, & j = \epsilon \end{cases} \tag{2.6}$$

During inference, the model only creates a link if the highest antecedent score is positive.

Joint Mention Detection and Mention Cluster During training, only mention cluster labels are available rather than antecedent links. Therefore, Lee et al. (2017) train the model end-to-end by maximizing the following marginal log-likelihood where GOLD(i) are gold antecedents for s_i:

$$\mathcal{L}_{\text{cluster}}(i) = \log \frac{\sum_{j' \in \text{GOLD}(i)} \exp(s(i,j'))}{\sum_{j=\epsilon,0,\ldots,i-1} \exp(s(i,j))} \tag{2.7}$$

However, the initial pruning is completely random and the mention scoring model only receives distant supervision if we only optimize the above mention cluster performance. This makes learning slow and ineffective especially for mention detection. Based on this observation, we propose to directly optimize mention detection:

$$\mathcal{L}_{\text{detect}}(i) = y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i) \tag{2.8}$$

where $\hat{y}_i = \text{sigmoid}(m(i))$, $y_i = 1$ if and only if s_i is in one of the gold mention clusters. Our final loss combines mention detection and clustering:

$$\mathcal{L}_{\text{loss}} = -\lambda_{\text{detect}} \sum_{i=1}^{N} \mathcal{L}_{\text{detect}}(i) - \sum_{i'=1}^{N'} \mathcal{L}_{\text{cluster}}(i')$$

where N is the number of all possible spans, N' is the number of unpruned spans, and $\lambda_{\text{detection}}$ controls weights of two terms.

	Avg. F1
Our model (single)	67.8
without mention detection loss	67.5
without biaffine attention	67.4
Lee et al. (2017)	67.3

Table 2.2: Ablation study on the development set.

2.4 Experiments

Data Set and Evaluation We evaluate our model on the CoNLL-2012 Shared Task English data (Pradhan et al., 2012) which is based on the OntoNotes corpus (Hovy et al., 2006). It contains 2,802/343/348 train/development/test documents in different genres.

We use three standard metrics: MUC (Vilain et al., 1995), B^3 (Bagga and Baldwin, 1998), and CEAF$_{\phi_4}$ (Luo, 2005). We report Precision, Recall, F1 for each metric and the average F1 as the final CoNLL score.

Implementation Details For fair comparisons, we follow the same hyperparameters as in Lee et al. (2017). We consider all spans up to 10 words and up to 250 antecedents. $\lambda = 0.4$ is used for span pruning. We use fixed concatenations of 300-dimension GloVe (Pennington et al., 2014) embeddings and 50-dimension embeddings from Turian et al. (2010). Character CNNs use 8-dimension learned embeddings and 50 kernels for each window size in {3,4,5}. LSTMs have hidden size 200, and each FFNN has two hidden layers with 150 units and ReLU (Nair and Hinton, 2010) activations. We include (speaker ID, document genre, span distance, span width) features as 20-dimensional learned embeddings. Word and character embeddings use 0.5 dropout. All hidden layers and feature embeddings use 0.2 dropout. The batch size is 1 document. Based on the results on the development set, $\lambda_{\text{detection}} = 0.1$ works best from $\{0.05, 0.1, 0.5, 1.0\}$. Model is trained with ADAM optimizer (Kingma and Ba, 2015a) and converges in around 200K updates, which is faster than that of Lee et al. (2017).

Overall Performance In Table 2.1, we compare our model with previous state-of-the-art systems. We obtain the best results in all F1 metrics. Our single model achieves 67.8%

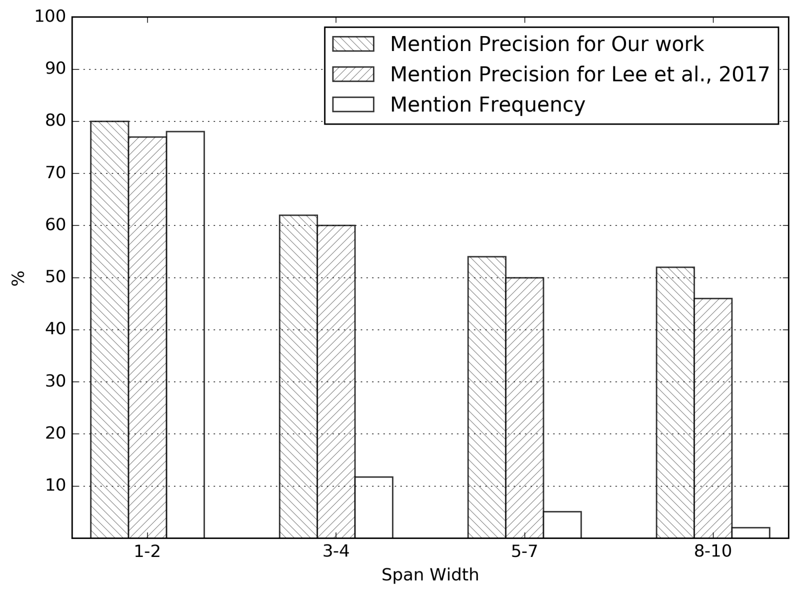


Figure 2.2: Mention detection subtask on development set. We plot accuracy and frequency breakdown by span widths.

F1 and our 5-model ensemble achieves 69.2% F1. In particular, compared with Lee et al. (2017), our improvement mainly results from the precision scores. This indicates that the mention detection loss does produce better mention scores and the biaffine attention more effectively determines if two spans are coreferent.

Ablation Study To understand the effect of different proposed components, we perform ablation study on the development set. As shown in Table 2.2, removing the mention detection loss term or the biaffine attention decreases 0.3/0.4 final F1 score, but still higher than the baseline. This shows that both components have contributions and when they work together the total gain is even higher.

Mention Detection Subtask To further understand our model, we perform a mention detection subtask where spans with mention scores higher than 0 are considered as mentions. We show the mention detection accuracy breakdown by span widths in Figure 2.2. Our model indeed performs better thanks to the mention detection loss. The advantage is even clearer for longer spans which consist of 5 or more words.

In addition, it is important to note that our model can detect mentions that do not exist in the training data. While Moosavi and Strube (2017) observe that there is a large overlap between the gold mentions of the training and dev (test) sets, we find that our model can correctly detect 1048 mentions which are not detected by Lee et al. (2017), consisting of 386 mentions existing in training data and 662 mentions not existing in training data. From those 662 mentions, some examples are (1) a suicide murder (2) Hong Kong Island (3) a US Airforce jet carrying robotic undersea vehicles (4) the investigation into who was behind the apparent suicide attack. This shows that our mention loss helps detection by generalizing to new mentions in test data rather than memorizing the existing mentions in training data.

2.5 Related Work

As summarized by Ng (2010), learning-based coreference models can be categorized into three types: (1) Mention-pair models train binary classifiers to determine if a pair of mentions are coreferent (Soon et al., 2001; Ng and Cardie, 2002; Bengtson and Roth, 2008). (2) Mention-ranking models explicitly rank all previous candidate mentions for the current mention and select a single highest scoring antecedent for each anaphoric mention (Denis and Baldridge, 2007b; Wiseman et al., 2015; Clark and Manning, 2016a; Lee et al., 2017). (3) Entity-mention models learn classifiers to determine whether the current mention is coreferent with a preceding, partially-formed mention cluster (Clark and Manning, 2015; Wiseman et al., 2016; Clark and Manning, 2016b).

In addition, we also note latent-antecedent models (Fernandes et al., 2012; Björkelund and Kuhn, 2014; Martschat and Strube, 2015). Fernandes et al. (2012) introduce coreference trees to represent mention clusters and learn to extract the maximum scoring tree in the graph of mentions.

Recently, several neural coreference resolution systems have achieved impressive gains (Wiseman et al., 2015, 2016; Clark and Manning, 2016b,a). They utilize distributed representations of mention pairs or mention clusters to dramatically reduce the number of hand-crafted features. For example, Wiseman et al. (2015) propose the first neural coreference resolution system by training a deep feed-forward neural network for mention ranking. However, these models still employ the two-stage pipeline and require a syntactic parser or a separate designed hand-engineered mention detector.

Finally, we also note the relevant work on joint mention detection and coreference resolution. Daumé III and Marcu (2005) propose to model both mention detection and coreference of the Entity Detection and Tracking task simultaneously. Denis and Baldridge (2007a) propose to use integer linear programming framework to model anaphoricity and coreference as a joint task.

Chapter 3

Dependency Sensitive Convolutional Neural Networks for Modeling Sentences and Documents

The goal of sentence and document modeling is to accurately represent the meaning of sentences and documents for various Natural Language Processing tasks. In this chapter, we present Dependency Sensitive Convolutional Neural Networks (DSCNN) as a general-purpose classification system for both sentences and documents. DSCNN hierarchically builds textual representations by processing pretrained word embeddings via Long Short-Term Memory networks and subsequently extracting features with convolution operators. Compared with existing recursive neural models with tree structures, DSCNN does not rely on parsers and expensive phrase labeling, and thus is not restricted to sentence-level tasks. Moreover, unlike other CNN-based models that analyze sentences locally by sliding windows, our system captures both the dependency information within each sentence and relationships across sentences in the same document. Experiment results demonstrate that our approach is achieving state-of-the-art performance on several tasks, including sentiment analysis, question type classification, and subjectivity classification.

This chapter is based on Zhang et al. (2016).

3.1 Introduction

Sentence and document modeling systems are important for many Natural Language Processing (NLP) applications. The challenge for textual modeling is to capture features for different text units and to perform compositions over variable-length sequences (e.g., phrases, sentences, documents). As a traditional method, the bag-of-words model treats sentences and documents as unordered collections of words. In this way, however, the bag-of-words model fails to encode word orders and syntactic structures.

By contrast, order-sensitive models based on neural networks are becoming increasingly popular thanks to their ability to capture word order information. Many prevalent order-sensitive neural models can be categorized into two classes: Recursive models and Convolutional Neural Networks (CNN) models. Recursive models can be considered as generalizations of traditional sequence-modeling neural networks to tree structures. For example, Socher et al. (2013) uses Recursive Neural Networks to build representations of phrases and sentences by combining neighboring constituents based on the parse tree. In their model, the composition is performed in a bottom-up way from leaf nodes of tokens until the root node of the parsing tree is reached. CNN based models, as the second category, utilize convolutional filters to extract local features (Kalchbrenner et al., 2014; Kim, 2014b) over embedding matrices consisting of pretrained word vectors. Therefore, the model actually splits the sentence locally into n-grams by sliding windows.

However, despite their ability to account for word orders, order-sensitive models based on neural networks still suffer from several disadvantages. First, recursive models depend on well-performing parsers, which can be difficult for many languages or noisy domains (Iyyer et al., 2015; Ma et al., 2015). Besides, since tree-structured neural networks are vulnerable to the vanishing gradient problem (Iyyer et al., 2015), recursive models require heavy labeling on phrases to add supervisions on internal nodes. Furthermore, parsing is restricted to sentences and it is unclear how to model paragraphs and documents using recursive neural networks. In CNN models, convolutional operators process word vectors sequentially using small windows. Thus sentences are essentially treated as a bag of n-grams, and the long dependency information spanning sliding windows is lost.

These observations motivate us to construct a textual modeling architecture that captures long-term dependencies without relying on parsing for both sentence and document inputs. Specifically, we propose Dependency Sensitive Convolutional Neural Networks (DSCNN), an end-to-end classification system that hierarchically builds textual representations with only root-level labels.

DSCNN consists of a convolutional layer built on top of Long Short-Term Memory (LSTM) networks. DSCNN takes slightly different forms depending on its input. For a single sentence (Figure 3.1), the LSTM network processes the sequence of word embeddings to capture long-distance dependencies within the sentence. The hidden states of the LSTM are extracted to form the low-level representation, and a convolutional layer with variable-size filters and max-pooling operators follows to extract task-specific features for classification purposes. As for document modeling (Figure 3.2), DSCNN first applies independent LSTM networks to each subsentence. Then a second LSTM layer is added between the first LSTM layer and the convolutional layer to encode the dependency across different sentences.

We evaluate DSCNN on several sentence-level and document-level tasks including sentiment analysis, question type classification, and subjectivity classification. Experimental results demonstrate the effectiveness of our approach comparable with the state-of-the-art. In particular, our method achieves highest accuracies on MR sentiment analysis (Pang and Lee, 2005), TREC question classification (Li and Roth, 2002), and subjectivity classification task SUBJ (Pang and Lee, 2004) compared with several competitive baselines.

The remaining part of this paper is the following. Section 2 discusses related work. Section 3 presents the background including LSTM networks and convolution operators. We then describe our architectures for sentence modeling and document modeling in Section 4, and report experimental results in Section 5.

3.2 Related Work

The success of deep learning architectures for NLP is first based on the progress in learning distributed word representations in semantic vector space (Bengio et al., 2003; Mikolov et al., 2013; Pennington et al., 2014), where each word is modeled with a real-valued vector called a word embedding. In this formulation, instead of using one-hot vectors by indexing words into a vocabulary, word embeddings are learned by projecting words onto a low dimensional and dense vector space that encodes both semantic and syntactic features of words.

Given word embeddings, different models have been proposed to learn the composition of words to build up phrase and sentence representations. Most methods fall into three types: unordered models, sequence models, and Convolutional Neural Networks models.

In unordered models, textual representations are independent of the word order. Specifically, ignoring the token order in the phrase and sentence, the bag-of-words model produces the representation by averaging the constituting word embeddings (Landauer and Dumais, 1997). Besides, a neural-bag-of-words model described in Kalchbrenner et al. (2014) adds an additional hidden layer on top of the averaged word embeddings before the softmax layer for classification purposes.

In contrast, sequence models, such as standard Recurrent Neural Networks (RNN) and Long Short-Term Memory networks, construct phrase and sentence representations in an order-sensitive way. For example, thanks to its ability to capture long-distance dependencies, LSTM has re-emerged as a popular choice for many sequence-modeling tasks, including machine translation (Bahdanau et al., 2014), image caption generation (Vinyals et al., 2014), and natural language generation (Wen et al., 2015a). Besides, RNN and LSTM can be both converted to tree-structured networks by using parsing information. For example, Socher et al. (2013) applied Recursive Neural Networks as a variant of the standard RNN structured by syntactic trees to the sentiment analysis task. Tai et al. (2015) also generalizes LSTM to Tree-LSTM where each LSTM unit combines information from its children units.

Recently, CNN-based models have demonstrated remarkable performances on sentence modeling and classification tasks. Leveraging convolution operators, these models can extract features from variable-length phrases corresponding to different filters. For example, DCNN in Kalchbrenner et al. (2014) constructs hierarchical features of sentences by one-dimensional convolution and dynamic k-max pooling. Yin and Schütze (2015) further utilizes multichannel embeddings and unsupervised pretraining to improve classification results.

3.3 Preliminaries

In this section, we describe two building blocks for our system. We first discuss Long Short-Term Memory as a powerful network for modeling sequence data, and then formulate convolution and max-over-time pooling operators for the feature extraction over sequence inputs.

3.3.1 Long Short-Term Memory

Recurrent Neural Network (RNN) is a class of models to process arbitrary-length input sequences by recursively constructing hidden state vectors $\vec{h}_t$. At each time step t, the hidden state $\vec{h}_t$ is an affine function of the input vector $\vec{x}_t$ at time t and its previous hidden state $\vec{h}_{t-1}$, followed by a non-linearity such as the hyperbolic tangent function:

$$\vec{h}_t = \tanh(\vec{W}\vec{x}_t + \vec{U}\vec{h}_{t-1} + b) \tag{3.1}$$

where $\vec{W}$, $\vec{U}$ and b are parameters of the model.

However, traditional RNN suffers from the exploding or vanishing gradient problems, where the gradient vectors can grow or decay exponentially as they propagate to earlier time steps. This problem makes it difficult to train RNN to capture long-distance dependencies in a sequence (Bengio et al., 1994; Hochreiter, 1998).

To address this problem of capturing long-term relations, Long Short-Term Memory (LSTM) networks, proposed by Hochreiter and Schmidhuber (1997) introduce a vector of memory cells and a set of gates to control how the information flows through the network. We thus have the input gate $\vec{i}_t$, the forget gate $\vec{f}_t$, the output gate $\vec{o}_t$, the memory cell $\vec{c}_t$, the input at the current step t as $\vec{x}_t$, and the hidden state $\vec{h}_t$, which are all in $\mathbb{R}^d$. Denote the sigmoid function as σ, and the element-wise multiplication as $\odot$. At each time step t, the LSTM unit manipulates a collection of vectors described by the following equations:

$$\begin{aligned}
\vec{i}_t &= \sigma\left(\vec{W}^{(i)}\vec{x}_t + \vec{U}^{(i)}\vec{h}_{t-1} + b^{(i)}\right)\\
\vec{f}_t &= \sigma\left(\vec{W}^{(f)}\vec{x}_t + \vec{U}^{(f)}\vec{h}_{t-1} + b^{(f)}\right)\\
\vec{o}_t &= \sigma\left(\vec{W}^{(o)}\vec{x}_t + \vec{U}^{(o)}\vec{h}_{t-1} + b^{(o)}\right)\\
\vec{u}_t &= \tanh\left(\vec{W}^{(u)}\vec{x}_t + \vec{U}^{(u)}\vec{h}_{t-1} + b^{(u)}\right)\\
\vec{c}_t &= \vec{i}_t \odot \vec{u}_t + \vec{f}_t \odot \vec{c}_{t-1}\\
\vec{h}_t &= \vec{o}_t \odot \tanh(\vec{c}_t)
\end{aligned} \tag{3.2}$$

Note that the gates $\vec{i}_t$, $\vec{f}_t$, $\vec{o}_t \in [0,1]^d$ and they control at time step t how the input is updated, how much the previous memory cell is forgotten, and the exposure of the memory to form the hidden state vector respectively.

3.3.2 Convolution and Max-over-time Pooling

Convolution operators have been extensively used in object recognition (LeCun et al., 1998), phoneme recognition (Waibel et al., 1989), sentence modeling and classification (Kalchbrenner et al., 2014; Kim, 2014b), and other traditional NLP tasks (Collobert and Weston, 2008). Given an input sentence of length s: $[w_1, w_2, ..., w_s]$, convolution operators apply a number of filters to extract local features of the sentence.

In this work, we employ one-dimensional wide convolution described in Kalchbrenner et al. (2014). Let $\vec{h}_t \in \mathbb{R}^d$ denote the representation of w_t, and $\vec{F} \in \mathbb{R}^{d \times l}$ be a filter where l is the window size. One-dimensional wide convolution computes the feature map $\vec{c}$ of length $(s + l - 1)$

$$\vec{c} = [c_1, c_2, ..., c_{s+l-1}] \tag{3.3}$$

for the input sentence.

Specifically, in wide convolution, we stack $\vec{h}_t$ column by column, and add $(l - 1)$ zero vectors to both ends of the sentence respectively. This formulates an input feature map $\vec{X} \in \mathbb{R}^{d \times (s+2l-2)}$. Thereafter, one-dimensional convolution applies the filter $\vec{F}$ to each set of consecutive l columns in $\vec{X}$ to produce $(s + l - 1)$ activations. The k-th activation is produced by

$$\vec{c}_k = f\left(b + \sum_{i,j}\left(\vec{F} \odot \vec{X}_{k:k+l-1}\right)_{i,j}\right) \tag{3.4}$$

where $\vec{X}_{k:k+l-1} \in \mathbb{R}^{d \times l}$ is the k-th sliding window in $\vec{X}$, and b is the bias term. $\odot$ performs element-wise multiplications and f is an nonlinear function such as Rectified Linear Units (ReLU) or the hyperbolic tangent.

Then, the max-over-time pooling selects the maximum value in the feature map

$$c_{\vec{F}} = \max(\vec{c}) \tag{3.5}$$

as the feature corresponding to the filter $\vec{F}$.

In practice, we apply many filters with different window sizes l to capture features encoded in l-length windows of the input.

3.4 Model Architecture

Convolutional Neural Networks have demonstrated state-of-the-art performances in sentence modeling and classification. Despite the fact that CNN is an order-sensitive model, traditional convolution operators extract local features from each possible window of words through filters with predefined sizes. Therefore, sentences are effectively processed like a bag of n-grams, and long-distance dependencies can be only captured if we have long enough filters.

To capture long-distance dependencies, much recent effort has been dedicated to building tree-structured models from the syntactic parsing information. However, we observe that these methods suffer from three problems. First, they require an external parser and are vulnerable to parsing errors (Iyyer et al., 2015). Besides, tree-structured models need heavy supervisions to overcome vanishing gradient problems. For example, in Socher et al. (2013), input sentences are labeled for each subphrase, and softmax layers are applied at each internal node. Finally, tree-structured models are restricted to sentence level, and cannot be generalized to model documents.

In this work, we propose a novel architecture to address these three problems. Our model hierarchically builds text representations from input words without parsing information. Only labels at the root level are required at the top softmax layer, so there is no need for labeling subphrases in the text. The system is not restricted to sentence-level inputs: the architecture can be restructured based on the sentence tokenization for modeling documents.

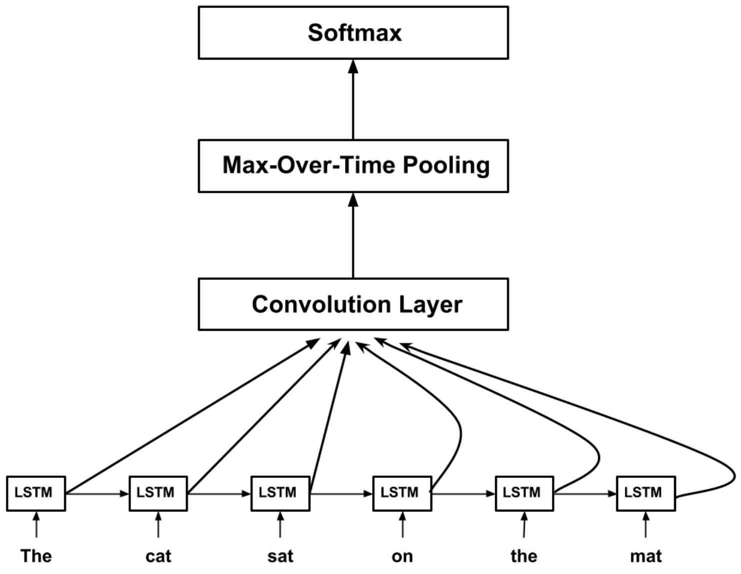


Figure 3.1: An example for sentence modeling. The bottom LSTM layer processes the input sentence and feed-forwards hidden state vectors at each time step. The one-dimensional wide convolution layer and the max-over-time pooling operation extract features from the LSTM output. For brevity, only one version of word embedding is illustrated in this figure.

3.4.1 Sentence Modeling

Let the input of our model be a sentence of length s: $[w_1, w_2, ..., w_s]$, and c be the total number of word embedding versions. Different versions come from pre-trained word vectors such as word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014).

The first layer of our model consists of LSTM networks processing multiple versions of word embedding. For each version of word embedding, we construct an LSTM network where the input $\vec{x}_t \in \mathbb{R}^d$ is the d-dimensional word embedding vector for w_t. As described in the previous section, the LSTM layer will produce a hidden state representation $\vec{h}_t \in \mathbb{R}^d$ at each time step. We collect hidden state representations as the output of LSTM layers:

$$\vec{h}^{(i)} = [\vec{h}_1^{(i)}, \vec{h}_2^{(i)}, ..., \vec{h}_t^{(i)}, ..., \vec{h}_s^{(i)}] \tag{3.6}$$

for $i = 1, 2, ..., c$.

A convolution neural network follows as the second layer. To deal with multiple word embeddings, we use filter $\vec{F} \in \mathbb{R}^{c \times d \times l}$, where l is the window size. Each hidden state sequence $\vec{h}^{(i)}$ produced by the i-th version of word embeddings forms one channel of the feature map. These feature maps are stacked as c-channel feature maps $\vec{X} \in \mathbb{R}^{c \times d \times (s+2(l-1))}$.

Similar to the single channel case, activations are computed as a slight modification of equation 3.4:

$$\vec{c}_k = f\left(b + \sum_{i,j,r} \left(\vec{F} \odot \vec{X}_{k:k+l-1}\right)_{i,j,r}\right) \tag{3.7}$$

A max-over-time pooling layer is then added on top of the convolution neural network. Finally, the pooled features are used in a softmax layer for classification. A sentence modeling example is illustrated in Figure 3.1.

3.4.2 Document Modeling

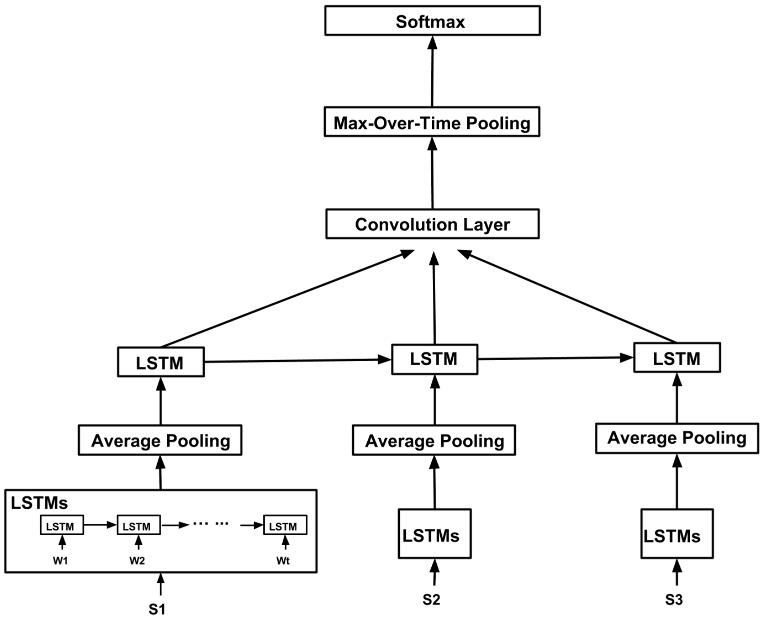


Figure 3.2: A schematic for document modeling hierarchy, which can be viewed as a variant of the one for sentence modeling. Independent LSTM networks process subsentences separated by punctuation. Hidden states of LSTM networks are averaged as the sentence representations, from which the high-level LSTM layer creates the joint meaning of sentences.

Our model is not restricted to sentences; it can be restructured to model documents.

The intuition comes from the fact that as the composition of words builds up the semantic meaning for sentences, the composition of sentences establishes the semantic meaning for documents (Li et al., 2015).

Now suppose that the input of our model is a document consisting of n subsentences: $[s_1, s_2, ..., s_n]$. Subsentences can be obtained by splitting the document using punctuation (comma, period, question mark, and exclamation point) as delimiters.

We employ independent LSTM networks for each subsentence in the same way as the first layer of the sentence modeling architecture. For each subsentence we feed-forward the hidden states of the corresponding LSTM network to the average pooling layer. Take the first sentence of the document as an example,

$$\vec{h}_{s1}^{(i)} = \frac{1}{\text{len}(s1)} \sum_{j=1}^{\text{len}(s1)} \vec{h}_{s1,j}^{(i)} \tag{3.8}$$

where $\vec{h}_{s1,j}^{(i)}$ is the hidden state of the first sentence at time step j, and len$(s1)$ denotes the length of the first sentence. In this way, after the averaging pooling layers, we have a representation sequence consisting of averaged hidden states for subsentences,

$$\vec{h}^{(i)} = [\vec{h}_{s1}^{(i)}, \vec{h}_{s2}^{(i)}, ..., \vec{h}_{sn}^{(i)}] \tag{3.9}$$

for $i = 1, 2, ..., c$.

Thereafter, a high-level LSTM network comes into play to capture the joint meaning created by the sentences.

Similar as sentence modeling, a convolutional layer is placed on top of the high-level LSTM for feature extraction. Finally, a max-over-time pooling layer and a softmax layer follow to pool features and perform the classification task. Figure 3.2 gives the schematic for the hierarchy.

3.5 Experiments

3.5.1 Datasets

Movie Review Data (MR) proposed by Pang and Lee (2005) is a dataset for sentiment analysis of movie reviews. The dataset consists of 5,331 positive and 5,331 negative reviews, mostly in one sentence. We follow the practice of using 10-fold cross validation to report results.

Stanford Sentiment Treebank (SST) is another popular sentiment classification dataset introduced by Socher et al. (2013). The sentences are labeled in a fine-grained way (SST-5): {very negative, negative, neutral, positive, very positive}. The dataset has been split into 8,544 training, 1,101 validation, and 2,210 testing sentences. Without neutral sentences, SST can also be used in binary mode (SST-2), where the split is 6,920 training, 872 validation, and 1,821 testing.

Furthermore, we apply DSCNN on question type classification task on TREC dataset (Li and Roth, 2002), where sentences are questions in the following 6 classes: {abbreviation, entity, description, location, numeric}. The entire dataset consists of 5,452 training examples and 500 testing examples.

We also benchmark our system on the subjectivity classification dataset (SUBJ) released by Pang and Lee (2004). The dataset contains 5,000 subjective sentences and 5,000 objective sentences. We report 10-fold cross validation results as the baseline does.

For document-level dataset, we use Large Movie Review (IMDB) created by Maas et al. (2011). There are 25,000 training and 25,000 testing examples with binary sentiment polarity labels, and 50,000 unlabeled examples. Different from Stanford Sentiment Treebank and Movie Review dataset, every example in this dataset has several sentences.

3.5.2 Training Details and Implementation

We use two sets of 300-dimensional pre-trained embeddings, word2vec[1] and GloVe[2], forming two channels for our network. For all datasets, we use 100 convolution filters each for window sizes of 3, 4, 5. Rectified Linear Units (ReLU) is chosen as the nonlinear function in the convolutional layer.

For regularization, before the softmax layers, we employ Dropout operation (Hinton et al., 2012) with dropout rate 0.5, and we do not perform any l_2 constraints over the parameters. We use the gradient-based optimizer Adadelta (Zeiler, 2012) to minimize cross-entropy loss between the predicted and true distributions, and the training is early stopped when the accuracy on validation set starts to drop.

As for training cost, our system processes around 4000 tokens per second on a single GTX 670 GPU. As an example, this amounts to 1 minute per epoch on the TREC dataset, converging within 50 epochs.

3.5.3 Pretraining of LSTM

We experiment with two variants of parameter initialization of sentence level LSTMs. The first variant (DSCNN in Table 3.1) initializes the weight matrices in LSTMs as random orthogonal matrices. In the second variant (DSCNN-Pretrain in Table 3.1), we first train sequence autoencoders (Dai and Le, 2015) which read input sentences at the encoder and reconstruct the input at the decoder. We pretrain separately on each task based on the same train/valid/test splits. The pretrained encoders are used to be the start points of LSTM layers for later supervised classification tasks.

1. https://code.google.com/p/word2vec/

2. http://nlp.stanford.edu/projects/glove/

Method	MR	SST-2	SST-5	TREC	SUBJ	IMDB
SVM (Socher et al., 2013)	—	79.4	40.7	—	—	—
NB (Socher et al., 2013)	—	81.8	41.0	—	—	—
NBSVM-bi (Wang and Manning, 2012)	79.4	—	—	—	93.2	91.2
SVM$_S$ (Silva et al., 2011)	—	—	—	95.0	—	—
Standard-RNN (Socher et al., 2013)	—	82.4	43.2	—	—	—
MV-RNN (Socher et al., 2012)	79.0	82.9	44.4	—	—	—
RNTN (Socher et al., 2013)	—	85.4	45.7	—	—	—
DRNN (Irsoy and Cardie, 2014)	—	86.6	49.8	—	—	—
Standard-LSTM (Tai et al., 2015)	—	86.7	45.8	—	—	—
bi-LSTM (Tai et al., 2015)	—	86.8	49.1	—	—	—
Tree-LSTM (Tai et al., 2015)	—	88.0	51.0	—	—	—
SA-LSTM (Dai and Le, 2015)	80.7	—	—	—	—	92.8
DCNN (Kalchbrenner et al., 2014)	—	86.8	48.5	93.0	—	—
CNN-MC (Kim, 2014b)	81.1	88.1	47.4	92.2	93.2	—
MVCNN (Yin and Schütze, 2015)	—	89.4	49.6	—	93.9	—
Dep-CNN (Ma et al., 2015)	81.9	—	49.5	95.4	—	—
Neural-BoW (Kalchbrenner et al., 2014)	—	80.5	42.4	88.2	—	—
DAN (Iyyer et al., 2015)	80.3	86.3	47.7	—	—	89.4
Paragraph-Vector (Le and Mikolov, 2014)	—	87.8	48.7	—	—	92.6
WRRBM+BoW(bnc) (Dahl et al., 2012)	—	—	—	—	—	89.2
Full+Unlabeled+BoW(bnc) (Maas et al., 2011)	—	—	—	—	88.2	88.9
DSCNN	81.5	89.1	49.7	95.4	93.2	90.2
DSCNN-Pretrain	82.2	88.7	50.6	95.6	93.9	90.7

Table 3.1: Experiment results of DSCNN compared with other models. Performance is measured in accuracy (%). Models are categorized into five classes. The first block is baseline methods including SVM and Naive Bayes and their variations. The second is the class of Recursive Neural Networks models. Constituent parsers and phrase-level supervision are needed. The third category is LSTMs. CNN models are fourth block, and the last category is a collection of other models achieving state-of-the-art results. **SVM**: Support Vector Machines with unigram features **NB**: Naive Bayes with unigram features **NBSVM-bi**: Naive Bayes SVM and Multinomial Naive Bayes with bigrams **SVM$_S$**: SVM with features including uni-bi-trigrams, POS, parser, and 60 hand-coded rules **Standard-RNN**: Standard Recursive Neural Network **MV-RNN**: Matrix-Vector Recursive Neural Network **RNTN**:Recursive Neural Tensor Network **DRNN**: Deep Recursive Neural Network **Standard-LSTM**: Standard Long Short-Term Memory Network **bi-LSTM**: Bidirectional LSTM **Tree-LSTM**: Tree-Structured LSTM **SA-LSTM**: Sequence Autoencoder LSTM. For fair comparison, we report the result on **MR** trained without unlabeled data from IMDB or Amazon reviews. **DCNN**: Dynamic Convolutional Neural Network with k-max pooling **CNN-MC**: Convolutional Neural Network with static pretrained and fine-tuned pretrained word-embeddings **MVCNN**: Multichannel Variable-Size Convolution Neural Network **Dep-CNN**: Dependency-based Convolutional Neural Network. Dependency parser is required. The result is for the combined model ancestor+sibling+sequential. **Neural-BoW** : Neural Bag-of-Words Models **DAN**: Deep Averaging Network **Paragraph-Vector**: Logistic Regression on Paragraph-Vector **WRRBM+BoW(bnc)**: word representation Restricted Boltzmann Machine combined with bag-of-words features **Full+Unlabeled+BoW(bnc)**:word vector based model capturing both semantic and sentiment, trained on unlabeled examples, and with bag-of-words features concatenated.

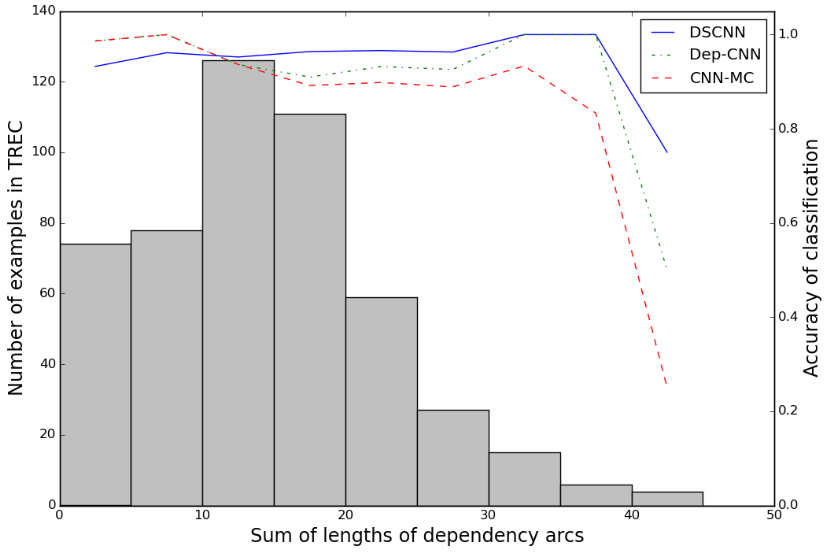


Figure 3.3: Number of sentences in TREC, and classification performances of DSCNN-Pretrain/Dep-CNN/CNN-MC as functions of dependency lengths. DSCNN and Dep-CNN clearly outperforms CNN-MC when the dependency length in the sentence grows.

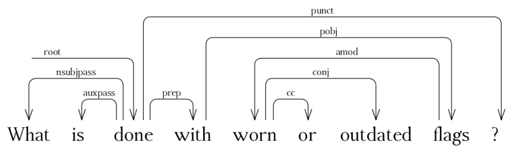
3.5.4 Results and Discussions

Table 3.1 reports the results of DSCNN on different datasets, demonstrating its effectiveness in comparison with other state-of-the-art methods.

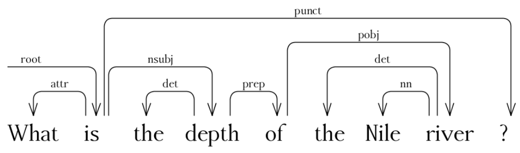
Sentence Modeling

For sentence modeling tasks, DSCNN beats all baselines on MR and TREC, and achieves the same best result on SUBJ as MVCNN. In SST-2, DSCNN only reports a slightly lower accuracy than MVCNN. In MVCNN, however, the author uses more resources including five versions of word embeddings. For SST-5, DSCNN is second only to Tree-LSTM, which nonetheless relies on parsers to build tree-structured neural models.

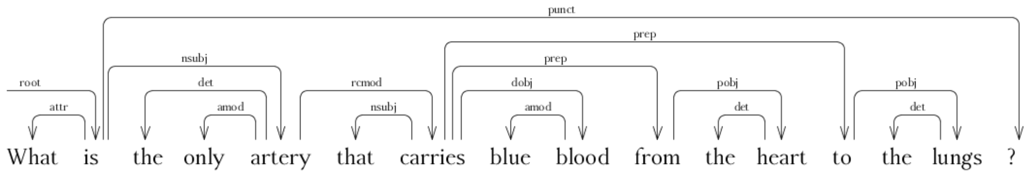
The benefit of DSCNN is illustrated by its consistently better results over the sequential CNN models including DCNN and CNN-MC. The superiority of DSCNN is mainly attributed to its ability to maintain long-term dependencies. Figure 3.3 depicts the cor-



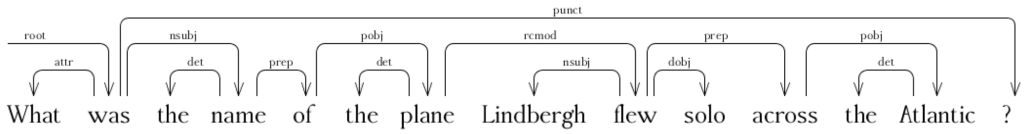
(a) *description → entity*



(b) *numeric → location*

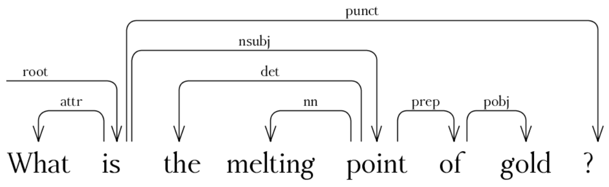

punct
prep
nsubj
prep
root
det
rcmod
dobj
pobj
pobj
attr
amod
nsubj
amod
det
det
What is the only artery that carries blue blood from the heart to the lungs ?

(c) *entity → location*

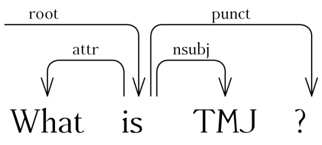

punct
root
nsubj
pobj
rcmod
prep
pobj
attr
det
prep
det
nsubj
dobj
det
What was the name of the plane Lindbergh flew solo across the Atlantic ?

(d) *entity → human*

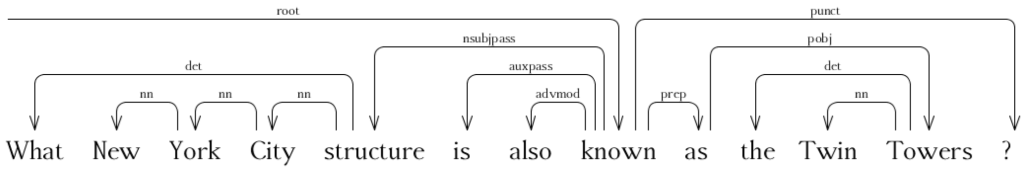
Figure 3.4: TREC examples that are misclassified by CNN-MC but correctly classified by DSCNN. For example, CNN-MC labels (a) as *entity* while the ground truth is *description*. Dependency Parsing is done by ClearNLP (Choi and Palmer, 2012).



(a) *numeric → description*



(b) *abbreviation → description*


root
punct
nsubjpass
pobj
det
auxpass
det
nn
nn
nn
advmod
prep
nn
What New York City structure is also known as the Twin Towers ?

(c) *location → entity*

Figure 3.5: TREC examples that are misclassified by DSCNN. For example, DSCNN labels (a) as *description* while the ground truth is *numeric*. Dependency Parsing is done by ClearNLP (Choi and Palmer, 2012).

relation between the dependency length and the classification accuracy. While CNN-MC and DSCNN are similar when the sum of dependency arc lengths is below 15, DSCNN gains obvious advantages when dependency lengths grow for long and complex sentences. Dep-CNN is also more robust than CNN-MC, but it relies on the dependency parser and predefined patterns to model longer linguistic structures.

Figure 3.4 gives some examples where DSCNN makes correct predictions while CNN-MC fails. In the first example, CNN-MC classifies the question as *entity* due to its focus on the noun phrase “worn or outdated flags”, while DSCNN captures the long dependency between “done with” and “flags”, and assigns the correct label *description*. Similarly in the second case, due to “Nile”, CNN-MC labels the question as *location*, while the dependency between “depth of” and “river” is ignored. As for the third example, the question involves a complicated and long attributive clause for the subject “artery”. CNN-MC gets easily confused and predicts the type as *location* due to words “from” and “to”, while DSCNN keeps correct. Finally, “Lindbergh” in the last example make CNN-MC bias to *human*.

We also sample some misclassified examples of DSCNN in Figure 3.5. Example (a) fails because the numeric meaning of “point” is not captured by the word embedding. Similarly, in the second example, the error is due to the out-of-vocabulary word “TMJ” and it is thus apparently difficult for DSCNN to figure out that it is an abbreviation. Example (c) is likely to be an ambiguous or mistaken annotation. The finding here agrees with the discussion in Dep-CNN work (Ma et al., 2015).

Document Modeling

For document modeling, the result of DSCNN on IMDB against other baselines is listed on the last column of Table 3.1. Documents in IMDB consist of several sentences and thus very long: the average length is 241 tokens per document and the maximum length is 2526 words (Dai and Le, 2015). As a result, there is no result reported using CNN-based models due to prohibited computation time, and most previous works are unordered models including variations of bag-of-words.

DSCNN outperforms bag-of-words model (Maas et al., 2011), Deep Averaging Network (Iyyer et al., 2015), and word representation Restricted Boltzmann Machine model combined with bag-of-words features (Dahl et al., 2012). The key weakness of bag-of-words prevents those models from capturing long-term dependencies.

Besides, Paragraph Vector (Le and Mikolov, 2014) and SA-LSTM (Dai and Le, 2015) achieve better results than DSCNN. It is worth mentioning that both methods, as unsupervised learning algorithms, can gain much positive effects from unlabeled data (they are using 50,000 unlabeled examples in IMDB). For example in Dai and Le (2015), with additional data from Amazon reviews, the error rate of SA-LSTM on MR dataset drops by 3.6%.

Chapter 4

Addressee and Response Selection in Multi-Party Conversations with Speaker Interaction RNNs

In this chapter, we study the problem of addressee and response selection in multi-party conversations. Understanding multi-party conversations is challenging because of complex speaker interactions: multiple speakers exchange messages with each other, playing different roles (sender, addressee, observer), and these roles vary across turns. To tackle this challenge, we propose the Speaker Interaction Recurrent Neural Network (SI-RNN). Whereas the previous state-of-the-art system updated speaker embeddings only for the sender, SI-RNN uses a novel dialog encoder to update speaker embeddings in a role-sensitive way. Additionally, unlike the previous work that selected the addressee and response separately, SI-RNN selects them jointly by viewing the task as a sequence prediction problem. Experimental results show that SI-RNN significantly improves the accuracy of addressee and response selection, particularly in complex conversations with many speakers and responses to distant messages many turns in the past.

This chapter is based on Zhang et al. (2018a).

4.1 Introduction

Real-world conversations often involve more than two speakers. In the Ubuntu Internet Relay Chat channel (IRC), for example, one user can initiate a discussion about an Ubuntu-related technical issue, and many other users can work together to solve the problem. Dialogs can have complex speaker interactions: at each turn, users play one of three roles (sender, addressee, observer), and those roles vary across turns.

In this paper, we study the problem of addressee and response selection in multi-party conversations: given a responding speaker and a dialog context, the task is to select an addressee and a response from a set of candidates for the responding speaker. The task requires modeling multi-party conversations and can be directly used to build retrieval-based dialog systems (Lu and Li, 2013; Hu et al., 2014; Ji et al., 2014; Wang et al., 2015a).

The previous state-of-the-art DYNAMIC-RNN model from Ouchi and Tsuboi (2016) maintains speaker embeddings to track each speaker status, which dynamically changes across time steps. It then produces the context embedding from the speaker embeddings and selects the addressee and response based on embedding similarity. However, this model updates only the sender embedding, not the embeddings of the addressee or observers, with the corresponding utterance, and it selects the addressee and response separately. In this way, it only models *who* says *what* and fails to capture addressee information. Experimental results show that the separate selection process often produces inconsistent addressee-response pairs.

To solve these issues, we introduce the Speaker Interaction Recurrent Neural Network (SI-RNN). SI-RNN redesigns the dialog encoder by updating speaker embeddings in a role-sensitive way. Speaker embeddings are updated in different GRU-based units depending on their roles (sender, addressee, observer). Furthermore, we note that the addressee and response are mutually dependent and view the task as a joint prediction problem. Therefore, SI-RNN models the conditional probability (of addressee given the response and vice versa) and selects the addressee and response pair by maximizing the joint probability.

On a public standard benchmark data set, SI-RNN significantly improves the addressee and response selection accuracy, particularly in complex conversations with many speakers and responses to distant messages many turns in the past. Our code and data set are available online.[3]

4.2 Related Work

We follow a data-driven approach to dialog systems. Singh et al. (1999), Henderson et al. (2008), and Young et al. (2013) optimize the dialog policy using Reinforcement Learning or the Partially Observable Markov Decision Process framework. In addition, Henderson et al. (2014a) propose to use a predefined ontology as a logical representation for the information exchanged in the conversation. The dialog system can be divided into different modules, such as Natural Language Understanding (Yao et al., 2014; Mesnil et al., 2015), Dialog State Tracking (Henderson et al., 2014b; Williams et al., 2016), and Natural Language Generation (Wen et al., 2015b). Furthermore, Wen et al. (2016) and Bordes and Weston (2017) propose end-to-end trainable goal-oriented dialog systems.

Recently, short text conversation has been popular. The system receives a short dialog context and generates a response using statistical machine translation or sequence-to-sequence networks (Ritter et al., 2011; Vinyals and Le, 2015; Shang et al., 2015; Serban et al., 2016; Li et al., 2016; Mei et al., 2017). In contrast to response generation, the retrieval-based approach uses a ranking model to select the highest scoring response from candidates (Lu and Li, 2013; Hu et al., 2014; Ji et al., 2014; Wang et al., 2015a). However, these models are single-turn responding machines and thus still are limited to short contexts with only two speakers. As for larger context, Lowe et al. (2015) propose the Next Utterance Classification (NUC) task for multi-turn two-party dialogs. Ouchi and Tsuboi (2016) extend NUC to multi-party conversations by integrating the addressee detection problem.

3. The released code: https://github.com/ryanzhumich/sirnn

Since the data is text based, they use only textual information to predict addressees as opposed to relying on acoustic signals or gaze information in multimodal dialog systems (Jovanović et al., 2006; op den Akker and Traum, 2009).

Furthermore, several other papers are recently presented focusing on modeling role-specific information given the dialogue contexts (Meng et al., 2017; Chi et al., 2017; Chen et al., 2017). For example, Meng et al. (2017) combine content and temporal information to predict the utterance speaker. By contrast, our SIRNN explicitly utilizes the speaker interaction to maintain speaker embeddings and predicts the addressee and response by joint selection.

4.3 Preliminaries

4.3.1 Addressee and Response Selection

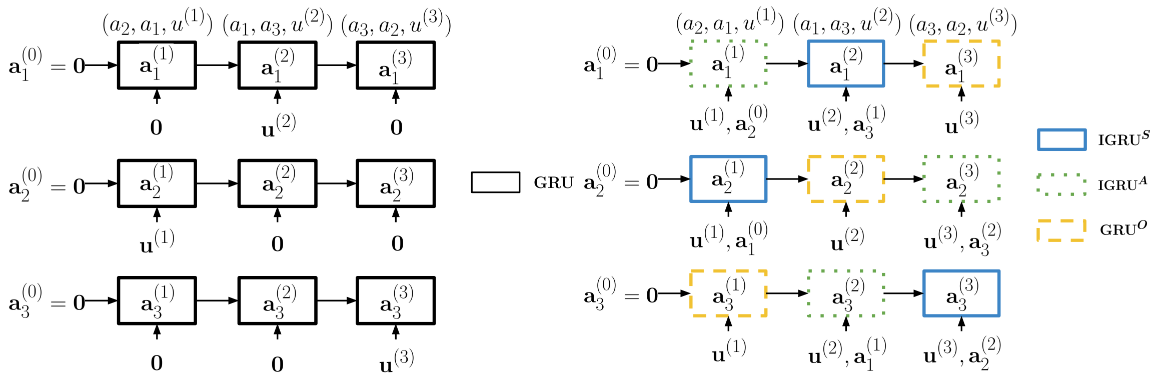
Ouchi and Tsuboi (2016) propose the addressee and response selection task for multi-party conversation. Given a responding speaker a_{res} and a dialog context $\mathcal{C}$, the task is to select a response and an addressee. $\mathcal{C}$ is a list ordered by time step:

$$\mathcal{C} = [(a^{(t)}_{sender}, a^{(t)}_{addressee}, u^{(t)})]^{T}_{t=1}$$

where $a^{(t)}_{sender}$ says $u^{(t)}$ to $a^{(t)}_{addressee}$ at time step t, and T is the total number of time steps before the response and addressee selection. The set of speakers appearing in $\mathcal{C}$ is denoted $\mathcal{A}(\mathcal{C})$. As for the output, the addressee is selected from $\mathcal{A}(\mathcal{C})$, and the response is selected from a set of candidates $\mathcal{R}$. Here, $\mathcal{R}$ contains the ground-truth response and one or more false responses. We provide some examples in Table 4.4 (Section 4.6).

	Data	Notation
Input	Responding Speaker	a_{res}
	Context	$\mathcal{C}$
	Candidate Responses	$\mathcal{R}$
Output	Addressee	$a \in \mathcal{A}(\mathcal{C})$
	Response	$r \in \mathcal{R}$
	Sender ID at time t	$a^{(t)}_{sender}$
	Addressee ID at time t	$a^{(t)}_{addressee}$
	Utterance at time t	$u^{(t)}$
	Utterance embedding at time t	$\mathbf{u}^{(t)}$
	Speaker embedding of a_i at time t	$\mathbf{a}^{(t)}_i$

Table 4.1: Notations for the task and model.



$(a_2, a_1, u^{(1)})$ $(a_1, a_3, u^{(2)})$ $(a_3, a_2, u^{(3)})$
$\mathbf{a}_1^{(0)} = \mathbf{0}$
$\mathbf{a}_2^{(0)} = \mathbf{0}$
$\mathbf{a}_3^{(0)} = \mathbf{0}$
GRU
IGRU^S
IGRU^A
GRU^O

Figure 4.1: Dialog encoders in DYNAMIC-RNN (Left) and SI-RNN (Right) for an example context at the top. Speaker embeddings are initialized as zero vectors and updated recurrently as hidden states along the time step. In SI-RNN, the same speaker embedding is updated in different units depending on the role (IGRU^S for sender, IGRU^A for addressee, GRU^O for observer).

4.3.2 DYNAMIC-RNN Model

In this section, we briefly review the state-of-the-art DYNAMIC-RNN model (Ouchi and Tsuboi, 2016), which our proposed model is based on. DYNAMIC-RNN solves the task in two phases: 1) the dialog encoder maintains a set of speaker embeddings to track each speaker status, which dynamically changes with time step t; 2) then DYNAMIC-RNN produces the context embedding from the speaker embeddings and selects the addressee and response based on embedding similarity among context, speaker, and utterance.

Dialog Encoder. Figure 4.1 (Left) illustrates the dialog encoder in DYNAMIC-RNN on an example context. In this example, a_2 says $u^{(1)}$ to a_1, then a_1 says $u^{(2)}$ to a_3, and finally a_3 says $u^{(3)}$ to a_2. The context $\mathcal{C}$ will be:

$$\mathcal{C} = [(a_2, a_1, u^{(1)}), (a_1, a_3, u^{(2)}), (a_3, a_2, u^{(3)})] \tag{4.1}$$

with the set of speakers $\mathcal{A}(\mathcal{C}) = \{a_1, a_2, a_3\}$.

For a speaker a_i, the bold letter $\mathbf{a}_i^{(t)} \in \mathbb{R}^{d_s}$ denotes its embedding at time step t. Speaker embeddings are initialized as zero vectors and updated recurrently as hidden states of GRUs (Cho et al., 2014; Chung et al., 2014). Specifically, for each time step t with the sender a_{sdr} and the utterance $u^{(t)}$, the sender embedding $\mathbf{a}_{sdr}$ is updated recurrently from the utterance:

$$\mathbf{a}_{sdr}^{(t)} = \mathrm{GRU}(\mathbf{a}_{sdr}^{(t-1)}, \mathbf{u}^{(t)}),$$

where $\mathbf{u}^{(t)} \in \mathbb{R}^{d_u}$ is the embedding for utterance $u^{(t)}$. Other speaker embeddings are updated from $\mathbf{u}^{(t)} = \mathbf{0}$. The speaker embeddings are updated until time step T.

Selection Model. To summarize the whole dialog context $\mathcal{C}$, the model applies element-wise max pooling over all the speaker embeddings to get the context embedding $\mathbf{h}_{\mathcal{C}}$:

$$\mathbf{h}_{\mathcal{C}} = \max_{a_i = a_1, \ldots, a_{|\mathcal{A}(\mathcal{C})|}} \mathbf{a}_i^{(T)} \in \mathbb{R}^{d_s} \tag{4.2}$$

The probability of an addressee and a response being the ground truth is calculated based on embedding similarity. To be specific, for addressee selection, the model compares the candidate speaker a_p, the dialog context $\mathcal{C}$, and the responding speaker a_{res}:

$$\mathbb{P}(a_p|\mathcal{C}) = \sigma([\mathbf{a}_{res}; \mathbf{h}_{\mathcal{C}}]^\top \mathbf{W}_a \mathbf{a}_p) \tag{4.3}$$

where $\mathbf{a}_{res}$ is the final speaker embedding for the responding speaker a_{res}, $\mathbf{a}_p$ is the final speaker embedding for the candidate addressee a_p, σ is the logistic sigmoid function, $[\,;\,]$ is the row-wise concatenation operator, and $\mathbf{W}_a \in \mathbb{R}^{2d_s \times d_s}$ is a learnable parameter. Similarly, for response selection,

$$\mathbb{P}(r_q|\mathcal{C}) = \sigma([\mathbf{a}_{res}; \mathbf{h}_{\mathcal{C}}]^\top \mathbf{W}_r \mathbf{r}_q) \tag{4.4}$$

where $\mathbf{r}_q \in \mathbb{R}^{d_u}$ is the embedding for the candidate response r_q, and $\mathbf{W}_r \in \mathbb{R}^{2d_s \times d_u}$ is a learnable parameter.

The model is trained end-to-end to minimize a joint cross-entropy loss for the addressee selection and the response selection with equal weights. At test time, the addressee and the response are separately selected to maximize the probability in Eq 4.3 and Eq 4.4.

4.4 Speaker Interaction RNN

While DYNAMIC-RNN can track the speaker status by capturing *who* says *what* in multi-party conversation, there are still some issues. First, at each time step, only the sender embedding is updated from the utterance. Therefore, other speakers are blind to what is being said, and the model fails to capture addressee information. Second, while the addressee and response are mutually dependent, DYNAMIC-RNN selects them independently. Consider a case where the responding speaker is talking to two other speakers in separate conversation threads. The choice of addressee is likely to be either of the two speakers, but the choice is much less ambiguous if the correct response is given, and vice versa. DYNAMIC-RNN often produces inconsistent addressee-response pairs due to the separate selection. See Table 4.4 for examples.

In contrast to DYNAMIC-RNN, the dialog encoder in SI-RNN updates embeddings for all the speakers besides the sender at each time step. Speaker embeddings are updated depending on their roles: the update of the sender is different from the addressee, which is different from the observers. Furthermore, the update of a speaker embedding is not only from the utterance, but also from other speakers. These are achieved by designing variations of GRUs for different roles. Finally, SI-RNN selects the addressee and response jointly by maximizing the joint probability.

4.4.1 Utterance Encoder

To encode an utterance $u = (w_1, w_2, ..., w_N)$ of N words, we use a RNN with Gated Recurrent Units (Cho et al., 2014; Chung et al., 2014):

$$\mathbf{h}_j = \mathrm{GRU}(\mathbf{h}_{j-1}, \mathbf{x}_j)$$

where $\mathbf{x}_j$ is the word embedding for w_j, and $\mathbf{h}_j$ is the GRU hidden state. $\mathbf{h}_0$ is initialized as a zero vector, and the utterance embedding is the last hidden state, i.e. $\mathbf{u} = \mathbf{h}_N$.

4.4.2 Dialog Encoder

Figure 4.1 (Right) shows how SI-RNN encodes the example in Eq 4.1. Unlike DYNAMIC-RNN, SI-RNN updates all speaker embeddings in a role-sensitive manner. For example, at the first time step when a_2 says $u^{(1)}$ to a_1, DYNAMIC-RNN only updates $\mathbf{a}_2$ using $\mathbf{u}^{(1)}$, while other speakers are updated using $\mathbf{0}$. In contrast, SI-RNN updates each speaker status with different units: IGRU^S updates the sender embedding $\mathbf{a}_2$ from the utterance embedding $\mathbf{u}^{(1)}$ and the addressee embedding $\mathbf{a}_1$; IGRU^A updates the addressee embedding $\mathbf{a}_1$ from $\mathbf{u}^{(1)}$ and $\mathbf{a}_2$; GRU^O updates the observer embedding $\mathbf{a}_3$ from $\mathbf{u}^{(1)}$.

Algorithm 1 Dialog Encoder in SI-RNN.

```
1: Input
2: Dialog context $\mathcal{C}$ with speakers $\mathcal{A}(\mathcal{C})$:
3: $\mathcal{C} = [(a^{(t)}_{sender}, a^{(t)}_{addressee}, u^{(t)})]^{T}_{t=1}$
4: $\mathcal{A}(\mathcal{C}) = \{a_1, a_2, \ldots, a_{|\mathcal{A}(\mathcal{C})|}\}$
5: where $a^{(t)}_{sender}, a^{(t)}_{addressee} \in \mathcal{A}(\mathcal{C})$
6: // Initialize speaker embeddings
7: for $a_i = a_1, a_2, \ldots, a_{|\mathcal{A}(\mathcal{C})|}$ do
8:     $\mathbf{a}^{(0)}_i \leftarrow \mathbf{0} \in \mathbb{R}^{d_s}$
9: end for
10: //Update speaker embeddings
11: for $t = 1, 2, \ldots, T$ do
12:     // Update sender, addressee, observers
13:     $a_{sdr} \leftarrow a^{(t)}_{sender}$
14:     $a_{adr} \leftarrow a^{(t)}_{addressee}$
15:     $\mathcal{O} \leftarrow \mathcal{A}(\mathcal{C}) - \{a_{sdr}, a_{adr}\}$
16:     // Compute utterance embedding
17:     $\mathbf{u}^{(t)} \leftarrow \mathrm{UtteranceEncoder}(u^{(t)})$
18:     $\mathbf{u}^{(t)} \leftarrow \mathrm{Concatenate}(\mathbf{a}^{(t-1)}_{sdr}, \mathbf{u}^{(t)})$
19:     // Update sender embedding
20:     $\mathbf{a}^{(t)}_{sdr} \leftarrow \mathrm{IGRU}^{S}(\mathbf{a}^{(t-1)}_{sdr}, \mathbf{a}^{(t-1)}_{adr}, \mathbf{u}^{(t)})$
21:     // Update addressee embedding
22:     $\mathbf{a}^{(t)}_{adr} \leftarrow \mathrm{IGRU}^{A}(\mathbf{a}^{(t-1)}_{adr}, \mathbf{a}^{(t-1)}_{sdr}, \mathbf{u}^{(t)})$
23:     // Update observer embeddings
24:     for $a_{obr}$ in $\mathcal{O}$ do
25:         $\mathbf{a}^{(t)}_{obr} \leftarrow \mathrm{GRU}^{O}(\mathbf{a}^{(t-1)}_{obr}, \mathbf{u}^{(t)})$
26:     end for
27: end for
28: // Return final speaker embeddings
29: Output
30: return $\mathbf{a}^{(T)}_i$ for $a_i = a_1, a_2, \ldots, a_{|\mathcal{A}(\mathcal{C})|}$
```

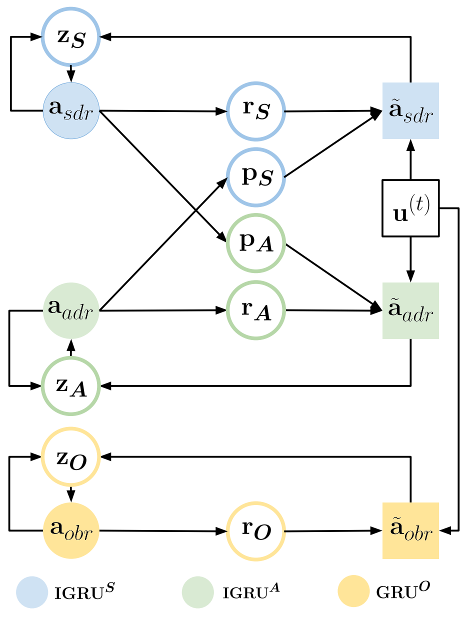


Figure 4.2: Illustration of IGRU^S (upper, blue), IGRU^A (middle, green), and GRU^O (lower, yellow). Filled circles are speaker embeddings, which are recurrently updated. Unfilled circles are gates. Filled squares are speaker embedding proposals.

Algorithm 1 gives a formal definition of the dialog encoder in SI-RNN. The dialog encoder is a function that takes as input a dialog context $\mathcal{C}$ (lines 1-5) and returns speaker embeddings at the final time step (lines 28-30). Speaker embeddings are initialized as d_s-dimensional zero vectors (lines 6-9). Speaker embeddings are updated by iterating over each line in the context (lines 10-27).

4.4.3 Role-Sensitive Update

In this subsection, we explain in detail how IGRU^S/IGRU^A/GRU^O update speaker embeddings according to their roles at each time step (Algorithm 1 lines 19-26).

As shown in Figure 4.2, IGRU^S/IGRU^A/GRU^O are all GRU-based units. IGRU^S updates the sender embedding from the previous sender embedding $\mathbf{a}_{sdr}^{(t-1)}$, the previous addressee embedding $\mathbf{a}_{adr}^{(t-1)}$, and the utterance embedding $\mathbf{u}^{(t)}$:

$$\mathbf{a}_{sdr}^{(t)} \leftarrow \text{IGRU}^S(\mathbf{a}_{sdr}^{(t-1)}, \mathbf{a}_{adr}^{(t-1)}, \mathbf{u}^{(t)})$$

The update, as illustrated in the upper part of Figure 4.2, is controlled by three gates. The $\mathbf{r}_S^{(t)}$ gate controls the previous sender embedding $\mathbf{a}_{sdr}^{(t-1)}$, and $\mathbf{p}_S^{(t)}$ controls the previous addressee embedding $\mathbf{a}_{adr}^{(t-1)}$. Those two gated interactions together produce the sender embedding proposal $\widetilde{\mathbf{a}}_{sdr}^{(t)}$. Finally, the update gate $\mathbf{z}_S^{(t)}$ combines the proposal $\widetilde{\mathbf{a}}_{sdr}^{(t)}$ and the previous sender embedding $\mathbf{a}_{sdr}^{(t-1)}$ to update the sender embedding $\mathbf{a}_{sdr}^{(t)}$. The computations in IGRU^S (including gates $\mathbf{r}_S^{(t)}$, $\mathbf{p}_S^{(t)}$, $\mathbf{z}_S^{(t)}$, the proposal embedding $\widetilde{\mathbf{a}}_{sdr}^{(t)}$, and the final updated embedding $\mathbf{a}_{sdr}^{(t)}$) are formulated as:

$$\begin{aligned}
\mathbf{r}_S^{(t)} &= \sigma(\mathbf{W}_S^r \mathbf{u}^{(t)} + \mathbf{U}_S^r \mathbf{a}_{sdr}^{(t-1)} + \mathbf{V}_S^r \mathbf{a}_{adr}^{(t-1)}) \\
\mathbf{p}_S^{(t)} &= \sigma(\mathbf{W}_S^p \mathbf{u}^{(t)} + \mathbf{U}_S^p \mathbf{a}_{sdr}^{(t-1)} + \mathbf{V}_S^p \mathbf{a}_{adr}^{(t-1)}) \\
\mathbf{z}_S^{(t)} &= \sigma(\mathbf{W}_S^z \mathbf{u}^{(t)} + \mathbf{U}_S^z \mathbf{a}_{sdr}^{(t-1)} + \mathbf{V}_S^z \mathbf{a}_{adr}^{(t-1)}) \\
\widetilde{\mathbf{a}}_{sdr}^{(t)} &= \tanh(\mathbf{W}_S \mathbf{u}^{(t)} + \mathbf{U}_S(\mathbf{r}_S^{(t)} \odot \mathbf{a}_{sdr}^{(t-1)}) \\
&\qquad + \mathbf{V}_S(\mathbf{p}_S^{(t)} \odot \mathbf{a}_{adr}^{(t-1)})) \\
\mathbf{a}_{sdr}^{(t)} &= \mathbf{z}_S^{(t)} \odot \mathbf{a}_{sdr}^{(t-1)} + (1 - \mathbf{z}_S^{(t)}) \odot \widetilde{\mathbf{a}}_{sdr}^{(t)}
\end{aligned}$$

where $\{\mathbf{W}_S^r, \mathbf{W}_S^p, \mathbf{W}_S^z, \mathbf{U}_S^r, \mathbf{U}_S^p, \mathbf{U}_S^z, \mathbf{V}_S^r, \mathbf{V}_S^p, \mathbf{V}_S^z, \mathbf{W}_S, \mathbf{U}_S, \mathbf{V}_S\}$ are learnable parameters. IGRU^A uses the same formulation with a different set of parameters, as illustrated in the middle of Figure 4.2. In addition, we update the observer embeddings from the utterance. GRU^O is implemented as the traditional GRU unit in the lower part of Figure 4.2. Note that the parameters in IGRU^S/IGRU^A/GRU^O are not shared. This allows SI-RNN to learn role-dependent features to control speaker embedding updates. The formulations of IGRU^A and GRU^O are similar.

4.4.4 Joint Selection

The dialog encoder takes the dialog context $\mathcal{C}$ as input and returns speaker embeddings at the final time step, $\mathbf{a}_i^{(T)}$. Recall from Section 4.3.2 that DYNAMIC-RNN produces the context embedding $\mathbf{h}_{\mathcal{C}}$ using Eq 4.2 and then selects the addressee and response separately using Eq 4.3 and Eq 4.4.

In contrast, SI-RNN performs addressee and response selection jointly: the response is dependent on the addressee and vice versa. Therefore, we view the task as a sequence prediction process: given the context and responding speaker, we first predict the addressee, and then predict the response given the addressee. (We also use the reversed prediction order as in Eq 4.7.)

In addition to Eq 4.3 and Eq 4.4, SI-RNN is also trained to model the conditional probability as follows. To predict the addressee, we calculate the probability of the candidate speaker a_p to be the ground-truth *given* the ground-truth response r (available during training time):

$$\mathbb{P}(a_p|\mathcal{C}, r) = \sigma([\mathbf{a}_{res}; \mathbf{h}_{\mathcal{C}}; \mathbf{r}]^\top \mathbf{W}_{ar} \mathbf{a}_p) \tag{4.5}$$

The key difference from Eq 4.3 is that Eq 4.5 is conditioned on the correct response r with embedding $\mathbf{r}$. Similarly, for response selection, we calculate the probability of a candidate response r_q *given* the ground-truth addressee a_{adr}:

$$\mathbb{P}(r_q|\mathcal{C}, a_{adr}) = \sigma([\mathbf{a}_{res}; \mathbf{h}_{\mathcal{C}}; \mathbf{a}_{adr}]^\top \mathbf{W}_{ra} \mathbf{r}_q) \tag{4.6}$$

At test time, SI-RNN selects the addressee-response pair from $\mathcal{A}(\mathcal{C}) \times \mathcal{R}$ to maximize the joint probability $\mathbb{P}(r_q, a_p|\mathcal{C})$:

$$\begin{aligned} \hat{a}, \hat{r} &= \mathop{\arg\max}_{a_p, r_q \in \mathcal{A}(\mathcal{C}) \times \mathcal{R}} \mathbb{P}(r_q, a_p|\mathcal{C}) \\ &= \mathop{\arg\max}_{a_p, r_q \in \mathcal{A}(\mathcal{C}) \times \mathcal{R}} \mathbb{P}(r_q|\mathcal{C}) \cdot \mathbb{P}(a_p|\mathcal{C}, r_q) \\ &\qquad + \mathbb{P}(a_p|\mathcal{C}) \cdot \mathbb{P}(r_q|\mathcal{C}, a_p) \end{aligned} \tag{4.7}$$

In Eq 4.7, we decompose the joint probability into two terms: the first term selects the response given the context, and then selects the addressee given the context and the selected response; the second term selects the addressee and response in the reversed order.[4]

		RES-CAND = 2				RES-CAND = 10			
		DEV	TEST			DEV	TEST		
	T	ADR-RES	ADR-RES	ADR	RES	ADR-RES	ADR-RES	ADR	RES
Chance	-	0.62	0.62	1.24	50.00	0.12	0.12	1.24	10.00
Recent+TF-IDF	15	37.11	37.13	55.62	67.89	14.91	15.44	55.62	29.19
Direct-Recent+TF-IDF	15	45.83	45.76	67.72	67.89	18.94	19.50	67.72	29.40
Static-RNN	5	47.08	46.99	60.39	75.07	21.96	21.98	60.26	33.27
Ouchi and Tsuboi (2016)	10	48.52	48.67	60.97	77.75	22.78	23.31	60.66	35.91
	15	49.03	49.27	61.95	78.14	23.73	23.49	60.98	36.58
Static-Hier-RNN	5	49.19	49.38	62.20	76.70	23.68	23.75	62.24	34.51
Zhou et al. (2016)	10	51.37	51.76	64.61	78.28	25.46	25.83	64.86	36.94
Serban et al. (2016)	15	52.78	53.04	65.84	79.08	26.31	26.62	65.89	37.85
Dynamic-RNN	5	49.38	49.80	63.19	76.07	23.44	23.72	63.28	33.62
Ouchi and Tsuboi (2016)	10	52.76	53.85	66.94	78.16	25.44	25.95	66.70	36.14
	15	54.45	54.88	68.54	78.64	26.73	27.19	68.41	36.93
	5	60.57	60.69	74.08	78.14	30.65	30.71	72.59	36.45
SI-RNN (Ours)	10	65.34	65.63	78.76	80.34	34.18	34.09	77.13	39.20
	15	**67.01**	**67.30**	**80.47**	**80.91**	**35.50**	**35.76**	**78.53**	**40.83**
SI-RNN w/ shared IGRUs	15	59.50	59.47	74.20	78.08	28.31	28.45	73.35	36.00
SI-RNN w/o joint selection	15	63.13	63.40	77.56	80.38	32.24	32.53	77.61	39.73

Table 4.2: Addressee and response selection results on the Ubuntu Multiparty Conversation Corpus. Metrics include accuracy of addressee selection (ADR), response selection (RES), and pair selection (ADR-RES). RES-CAND: the number of candidate responses. T: the context length.

4.5 Experimental Setup

Data Set. We use the Ubuntu Multiparty Conversation Corpus (Ouchi and Tsuboi, 2016) and summarize the data statistics in Table 4.3. The whole data set including the Train, Dev, Test split and the false response candidates is publicly available.[5] The data set is built from the Ubuntu IRC chat room where a number of users discuss Ubuntu-related technical issues. The log is organized as one file per day corresponding to a document $\mathcal{D}$. Each document consists of (Time, SenderID, Utterance) lines. If users explicitly mention addressees at the beginning of the utterance, the addresseeID is extracted. Then a sample, namely a unit of input (the dialog context and the current sender) and output (the addressee

4. Detail: We also considered an alternative decomposition of the joint probability as $\log \mathbb{P}(r_q, a_p|\mathcal{C}) = \frac{1}{2}[\log \mathbb{P}(r_q|\mathcal{C}) + \log \mathbb{P}(a_p|\mathcal{C}, r_q) + \log \mathbb{P}(a_p|\mathcal{C}) + \log \mathbb{P}(r_q|\mathcal{C}, a_p)]$, but the performance was similar to Eq 4.7.

5. https://github.com/hiroki13/response-ranking/tree/master/data/input

	Total	Train	Dev	Test
# Docs	7,355	6,606	367	382
# Utters	2.4M	2.1M	132.4k	151.3k
# Samples	-	665.6k	45.1k	51.9k
Adr Mention Freq	-	0.32	0.34	0.34
# Speakers / Doc	26.8	26.3	30.7	32.1
# Utters / Doc	326.3	317.9	360.8	396.1
# Words / Utter	11.1	11.1	11.2	11.3

Table 4.3: Data Statistics. "Adr Mention Freq" is the frequency of explicit addressee mention.

and response prediction) for the task, is created to predict the ground-truth addressee and response of this line. Note that samples are created only when the addressee is explicitly mentioned for clear, unambiguous ground-truth labels. False response candidates are randomly chosen from all other utterances *within the same document*. Therefore, distractors are likely from the same sub-conversation or even from the same sender but at different time steps. This makes it harder than Lowe et al. (2015) where distractors are randomly chosen from all documents. If no addressee is explicitly mentioned, the addressee is left blank and the line is marked as a part of the context.

Baselines. Apart from DYNAMIC-RNN, we also include several other baselines. As the first baseline, RECENT+TF-IDF always selects the most recent speaker (except the responding speaker a_{res}) as the addressee and chooses the response to maximize the tf-idf cosine similarity with the context. We improve it by using a slightly different addressee selection heuristic (DIRECT-RECENT+TF-IDF): select the most recent speaker that *directly talks to* a_{res} by an explicit addressee mention. We select from the previous 15 utterances, which is the longest context among all the experiments. This works much better when there are multiple concurrent sub-conversations, and a_{res} responds to a distant message in the context. We also include another GRU-based model STATIC-RNN from Ouchi and Tsuboi (2016). Unlike DYNAMIC-RNN, speaker embeddings in STATIC-RNN are based on the order of speakers and are fixed. Furthermore, inspired by Zhou et al. (2016) and Serban et al. (2016), we implement STATIC-HIER-RNN, a hierarchical version of STATIC-RNN.

It first builds utterance embeddings from words and then uses high-level RNNs to process utterance embeddings.

4.6 Results and Discussion

For fair and meaningful quantitative comparisons, we follow Ouchi and Tsuboi (2016)'s evaluation protocols. SI-RNN improves the overall accuracy on the addressee and response selection task. Two ablation experiments further analyze the contribution of role-sensitive units and joint selection respectively. We then confirm the robustness of SI-RNN with the number of speakers and distant responses. Finally, in a case study we discuss how SI-RNN handles complex conversations by either engaging in a new sub-conversation or responding to a distant message.

4.6.1 Overall Result

As shown in Table 4.2, SI-RNN significantly improves upon the previous state-of-the-art. In particular, addressee selection (ADR) benefits most, with different number of candidate responses (denoted as RES-CAND): around 12% in RES-CAND $= 2$ and more than 10% in RES-CAND $= 10$. Response selection (RES) is also improved, suggesting role-sensitive GRUs and joint selection are helpful for response selection as well. The improvement is more obvious with more candidate responses (2% in RES-CAND $= 2$ and 4% in RES-CAND $= 10$). These together result in significantly better accuracy on the ADR-RES metric as well.

4.6.2 Ablation Study

We show an ablation study in the last rows of Table 4.2. First, we share the parameters of IGRU^S/IGRU^A/GRU^O. The accuracy decreases significantly, indicating that it is crucial to learn role-sensitive units to update speaker embeddings. Second, to examine our joint

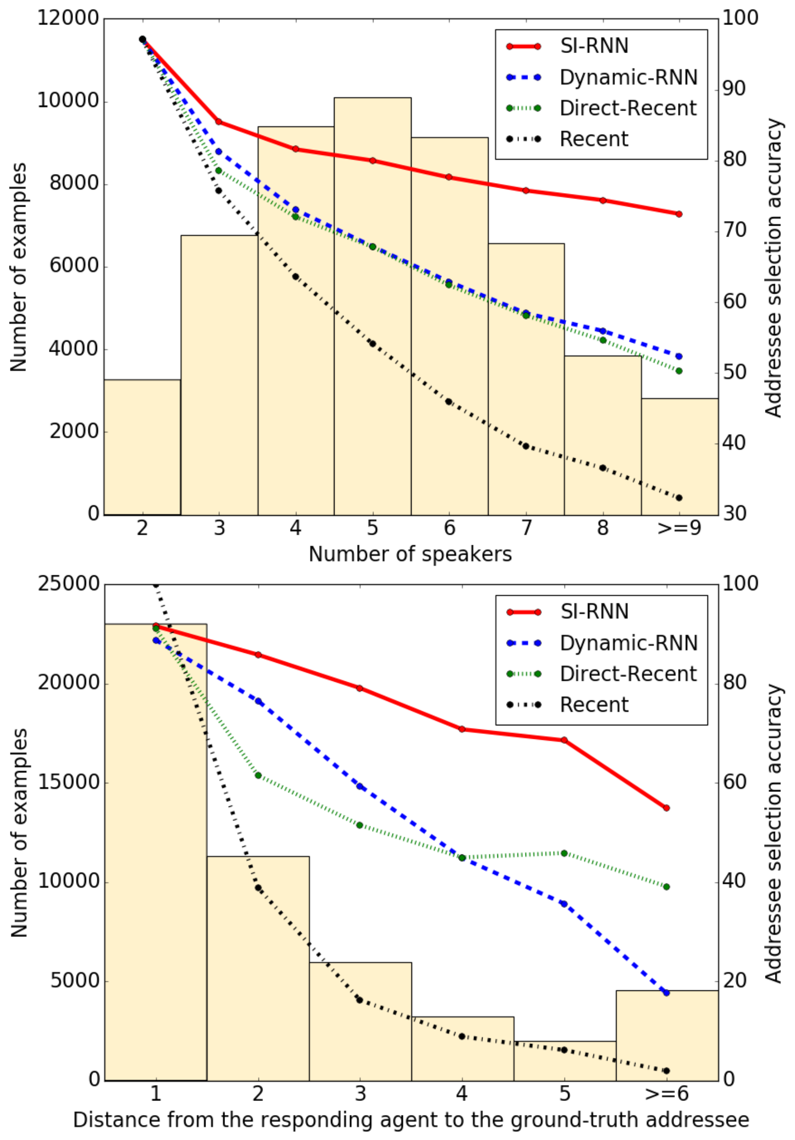


Figure 4.3: Effect of the number of speakers in the context (Upper) and the addressee distance (Lower). Left axis: the histogram shows the number of test examples. Right axis: the curves show ADR accuracy on the test set.

selection, we fall back to selecting the addressee and response separately, as in DYNAMIC-RNN. We find that joint selection improves ADR and RES individually, and it is particularly helpful for pair selection ADR-RES.

4.6.3 Number of Speakers

Numerous speakers create complex dialogs and increased candidate addressee, thus the task becomes more challenging. In Figure 4.3 (Upper), we investigate how ADR accuracy changes with the number of speakers in the context of length 15, corresponding to the rows with T=15 in Table 4.2. RECENT+TF-IDF always chooses the most recent speaker and the accuracy drops dramatically as the number of speakers increases. DIRECT-RECENT+TF-IDF shows better performance, and DYNAMIC-RNN is marginally better. SI-RNN is much more robust and remains above 70% accuracy across all bins. The advantage is more obvious for bins with more speakers.

4.6.4 Addressing Distance

Addressing distance is the time difference from the responding speaker to the ground-truth addressee. As the histogram in Figure 4.3 (Lower) shows, while the majority of responses target the most recent speaker, many responses go back five or more time steps. It is important to note that for those distant responses, DYNAMIC-RNN sees a clear performance decrease, even worse than DIRECT-RECENT+TF-IDF. In contrast, SI-RNN handles distant responses much more accurately.

	Sender	Addressee	Utterance
1	codepython	wafflejock	thanks
1	wafflejock	codepython	yup np
2	wafflejock	theoletom	you can use sudo apt-get install packagename – reinstall, to have apt-get install reinstall some package/metapackage and redo the configuration for the program as well
3	codepython	-	i installed ubuntu on a separate external drive. now when i boot into mac, the external drive does not show up as bootable. the blue light is on. any ideas?
4	Guest54977	-	hello there. wondering to anyone who knows, where an ubuntu backup can be retrieved from.
2	theoletom	wafflejock	it's not a program. it's a desktop environment.
4	Guest54977	-	did some searching on my system and googling, but couldn't find an answer
2	theoletom	-	be a trace of it left yet there still is.
2	theoletom	-	i think i might just need a fresh install of ubuntu. if there isn't a way to revert to default settings
5	releaf	-	what's your opinion on a $500 laptop that will be a dedicated ubuntu machine?
5	releaf	-	are any of the pre-loaded ones good deals?
5	releaf	-	if not, are there any laptops that are known for being oem-heavy or otherwise ubuntu friendly?
3	codepython	-	my usb stick shows up as bootable (efi) when i boot my mac. but not my external hard drive on which i just installed ubuntu. how do i make it bootable from mac hardware?
3	Jordan_U	codepython	did you install ubuntu to this external drive from a different machine?
5	Umeaboy	releaf	what country you from?
5	wafflejock		

Model Prediction	Addressee	Response
Direct-Recent+TF-IDF	theoletom	ubuntu install fresh
Dynamic-RNN	codepython	no prime is the replacement
SI-RNN	⋆ releaf	⋆ there are a few ubuntu dedicated laptop providers like umeaboy is asking depends on where you are

(a) SI-RNN chooses to engage in a new sub-conversation by suggesting a solution to "releaf" about Ubuntu dedicated laptops.

	Sender	Addressee	Utterance
1	VeryBewitching	nicomachus	anything i should be concerned about before i do it?
1	nicomachus	VeryBewitching	always back up before partitioning.
1	VeryBewitching	nicomachus	i would have assumed that, i was wondering more if this is something that tends to be touch and go, want to know if i should put coffee on :)
2	TechMonger	-	it was hybernating. i can ping it now
2	TechMonger	-	why does my router pick up disconnected devices when i reset my device list? or how
2	Ionic	-	because the dhcp refresh interval hasn't passed yet?
2	TechMonger	-	so dhcp refresh is different than device list refresh?
2	D33p	TechMonger	what an enlightenment @techmonger :)
2	BuzzardBuzz	-	dhcp refresh for all clients is needed when you change your subnet ip
2	BuzzardBuzz	-	if you want them to work together
2	Ionic	BuzzardBuzz	uhm, no.
2	chingao	TechMonger	nicomachus asked this way at the beginning: is the machine that you 're trying to ping turned on?
1	nicomachus		

Model Prediction	Addressee	Response
Direct-Recent+TF-IDF	⋆ VeryBewitching	i have tried with this program y-ppa manager, yet still doesn't work.
Dynamic-RNN	chingao	install the package "linux-generic", that will install the kernel and the headers if they are not installed
SI-RNN	⋆ VeryBewitching	⋆ if it's the last partition on the disk, it won't take long. if gparted has to copy data to move another partition too, it can take a couple hours.

(b) SI-RNN remembers the distant sub-conversation 1 and responds to "VeryBewitching" with a detailed answer.

Table 4.4: Case Study. ⋆ denotes the ground-truth. Sub-conversations are coded with different numbers for the purpose of analysis (sub-conversation labels are not available during training or testing).

4.6.5 Case Study

Examples in Table 4.4 show how SI-RNN can handle complex multi-party conversations by selecting from 10 candidate responses. In both examples, the responding speakers participate in two or more concurrent sub-conversations with other speakers.

Example (a) demonstrates the ability of SI-RNN to engage in a new sub-conversation. The responding speaker “wafflejock” is originally involved in two sub-conversations: the sub-conversation 1 with “codepython”, and the ubuntu installation issue with “theoletom”. While it is reasonable to address “codepython” and “theoletom”, the responses from other baselines are not helpful to solve corresponding issues. TF-IDF prefers the response with the “install” key-word, yet the response is repetitive and not helpful. DYNAMIC-RNN selects an irrelevant response to “codepython”. SI-RNN chooses to engage in a new sub-conversation by suggesting a solution to “releaf” about Ubuntu dedicated laptops.

Example (b) shows the advantage of SI-RNN in responding to a distant message. The responding speaker “nicomachus” is actively engaged with “VeryBewitching” in the sub-conversation 1 and is also loosely involved in the sub-conversation 2: “chingao” mentions “nicomachus” in the most recent utterance. SI-RNN remembers the distant sub-conversation 1 and responds to “VeryBewitching” with a detailed answer. By contrast, DIRECT-RECENT+TF-IDF selects the ground-truth addressee because “VeryBewitching” talks to “nicomachus”, but the response is not helpful. DYNAMIC-RNN is biased to the recent speaker “chingao”, yet the response is not relevant.

Chapter 5

This Email Could Save Your Life: Introducing the Task of Email Subject Line Generation

Given the overwhelming number of emails, an effective subject line becomes essential to better inform the recipient of the email's content. In this chapter, we propose and study the task of *email subject line generation*: automatically generating an email subject line from the email body. We create the first dataset for this task and find that email subject line generation favor extremely abstractive summary which differentiates it from news headline generation or news single document summarization. We then develop a novel deep learning method and compare it to several baselines as well as recent state-of-the-art text summarization systems. We also investigate the efficacy of several automatic metrics based on correlations with human judgments and propose a new automatic evaluation metric. Our system outperforms competitive baselines given both automatic and human evaluations. To our knowledge, this is the first work to tackle the problem of effective email subject line generation.

This chapter is based on Zhang and Tetreault (2019).

5.1 Introduction

Email is a ubiquitous form of online communication. An email message consists of two basic elements: an *email subject line* and an *email body*. The subject line, which is displayed to the recipient in the list of inbox messages, should tell what the email body is about and what the sender wants to convey. An effective email subject line becomes essential since it can help people manage a large number of emails. Table 5.1 shows an email body with three possible subject lines.

There have been several research tracks around email usage. While much effort has been focused on email summarization (Muresan et al., 2001; Nenkova and Bagga, 2003; Rambow et al., 2004), email keyword extraction and action detection (Turney, 2000; Lahiri et al., 2017; Lin et al., 2018), and email classification (Prabhakaran and Rambow, 2014; Alkhereyf and Rambow, 2017), to our knowledge there is no previous work on generating email subjects. In this paper, we propose the task of Subject Line Generation (SLG): automatically producing email subjects given the email body. While this is similar to email summarization, the two tasks serve different purposes in the process of email composition and consumption. A subject line is required when the sender writes the email, while a summary is more useful for long emails to benefit the recipient. An automatically generated email subject can also be used for downstream applications such as email triaging to help people manage emails more efficiently. Furthermore, while being similar to news headline generation or news single document summarization, email subjects are generally much shorter, which means a system must have the ability to summarize with a high compression ratio (Table 5.2). Therefore, we believe this task can also benefit other highly abstractive summarization such as generating section titles for long documents to improve reading comprehension speed and accuracy.

To introduce the task, we build the first dataset, Annotated Enron Subject Line Corpus (AESLC), by leveraging the Enron Corpus (Klimt and Yang, 2004) and crowdsourcing. Furthermore, in order to properly evaluate the subject, we use a combination of automatic

Email Body: Hi All, I would be grateful if you could get to me today via email a job description for your current role. I would like to get this to the immigration attorneys so that they can finalise the paperwork in preparation for INS filing once the UBS deal is signed. Kind regards,
Subject 1: Current Job Description Needed *(COMMENT: This is good because it is both informative and succinct.)*
Subject 2: Job Description *(COMMENT: This is okay but not informative enough.)*
Subject 3: Request *(COMMENT: This is bad because it does not contain any specific information about the request.)*

Table 5.1: An email with three possible subject lines.

metrics from the text summarization and machine translation fields, in addition to building our own regression-based Email Subject Quality Estimator (ESQE). Third, to generate effective email subjects, we propose a method which combines extractive and abstractive summarization using a two-stage process by *Multi-Sentence Selection and Rewriting with Email Subject Quality Estimation Reward*. The multi-sentence extractor first selects multiple sentences from the input email body. Extracted sentences capture salient information for writing a subject such as named entities and dates. Thereafter, the multi-sentence abstractor rewrites multiple selected sentences into a succinct subject line while preserving key information. For training the network, we use a multi-stage training strategy incorporating both supervised cross-entropy training and reinforcement learning (RL) by optimizing the reward provided by the ESQE model.

Our contributions are threefold: (1) We introduce the task of email subject line generation (SLG) and build a benchmark dataset AESLC.[6] (2) We investigate possible automatic metrics for SLG and study their correlations with human judgments. We also introduce a new email subject quality estimation metric (ESQE). (3) We propose a novel model to

6. dataset available at https://github.com/ryanzhumich/AESLC

generate email subjects. Our automatic and human evaluations demonstrate that our model outperforms competitive baselines and approaches human-level quality.

5.2 Annotated Enron Subject Line Corpus

To prepare our email subject line dataset, we use the Enron dataset (Klimt and Yang, 2004) which is a collection of email messages of employees in the Enron Corporation. We use Enron because it can be released to the public and it contains business and personal type emails for which the subject line is already well-defined and useful. As shown in Table 5.2, email subjects are typically much shorter than summaries generated in previous news datasets. While being similar to news headline generation (Rush et al., 2015), email subject generation is also more challenging in the sense that it deals with different types of email subjects while the first sentence of a news article is often already a good headline and summary.

Dataset	domain	docs (train/val/test)	avg doc words	avg summary words
CNN (Cheng and Lapata, 2016)	News	90,266/1,220/1,093	760	46
XSum (Narayan et al., 2018a)	News	204,045/11,332/11,334	431	23
Gigaword News Headline (Rush et al., 2015)	News	3,799,588/394,622/381,197	31	8
Annotated Enron Subject Line Corpus	Business/Personal	14,436/1,960/1,906	75	4

Table 5.2: Annotated Enron Subject Line Corpus compared with other datasets.

5.2.1 Data Preprocessing

The original Enron dataset contains 517,401 email messages from 150 user mailboxes. To extract body and subject pairs from the dataset, we take all messages from the inbox and sent folders of all mailboxes. We then perform email body cleaning, email filtering, and email de-duplication.

We first remove any content from the email body that has not been written by the author of the email. This includes automatically appended boilerplate material such as advertisements, attachments, legal disclaimers etc. Since we are interested in emails with enough information to generate meaningful subjects, we only keep emails with at least 3 sentences and 25 words in the email body. Furthermore, to ensure that the email subject truly corresponds to the content in the email body, we only take the first email of a thread and exclude replies or forward emails. So we filter out follow up messages which contain "Original Message" section in the email body or have subject lines starting with "RE:" (reply-to messages) or "FW:" (forward messages). Finally, we observe that the same message can be sent to multiple recipients so we remove duplicate emails to make sure there is no overlap between the train and test set. We only keep the subject and body while other information such as the sender/recipient identity can be incorporated in future work.

5.2.2 Subject Annotation

We noted that using only the original subject lines as references may be problematic for automatic evaluation purposes. First, there can be many different valid, effective subject lines for the same email, yet the original email subject is only one of them. This is similar to why automatic machine translation evaluation often relies on multiple references. Second, the email subject may be too general or too vague when the sender does not put that much effort into writing. Third, the sender may assume some shared knowledge with the recipient so that the email subject contains information that cannot be found in the email body.

To address the issues above, we ask workers on Amazon Mechanical Turk to read Enron emails in our dev and test sets and write an appropriate subject line. Each email is annotated with 3 subject lines from 3 different annotators. For quality control, we manually review and reject improper email subjects such as empty subject lines, subject lines with typos, and subject lines that are too general or too vague, e.g., "Update", "Schedule", "Attention to Detail" because they contain no body-specific information and can be applied generically to many emails. We found that while three annotations are different, they often contain common keywords. To further quantify the variation among human annotations, we compute ROUGE-L F1 scores for each pair of annotations: 34.04, 33.38, 34.26.

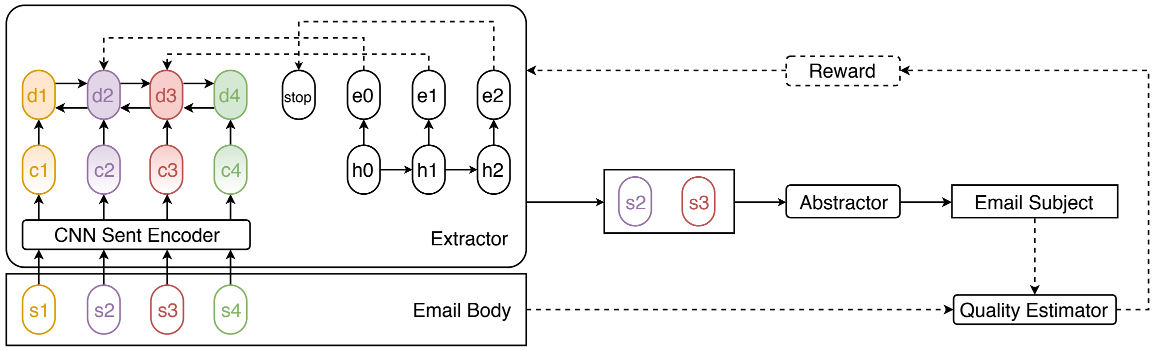


Figure 5.1: Our model architecture. In this example, the input email body consists of four sentences from which the extractor selects the second and the third. The abstractor generates an email subject from the selected sentences. The quality estimator provides rewards by scoring the subject against the email body.

5.3 Our Model

Our model is illustrated in Figure 5.1. Based on recent progress in news summarization (Chen and Bansal, 2018), our model generates email subjects in two stages: (1) The extractor selects multiple sentences containing salient information for writing a subject (§5.3.1). (2) The abstractor rewrites multiple selected sentences into a succinct subject line while preserving key information (§5.3.2).

We employ a multi-stage training strategy (§5.3.4) including a Reinforcement Learning (RL) phase because of its usefulness for text generation tasks (Ranzato et al., 2016; Bahdanau et al., 2017) to optimize the non-differentiable metrics such as ROUGE and METEOR. However, unlike ROUGE for summarization or METEOR for machine translation, there is no available automatic metric designed for email subject generation. Motivated by recent work on regression-based metrics for machine translation (Shimanaka et al., 2018) and dialog response generation (Lowe et al., 2017), we build a neural network (ESQE) to estimate the quality of an email subject given the email body (§5.3.3). The estimator is pretrained and fixed during RL training phase to provide rewards for the extractor agent.

While our model is based on Chen and Bansal (2018), they assume that there is a one-to-one relationship between the summary sentence and the document sentence: every summary sentence can be rewritten from exactly one sentence in the document. They also use ROUGE to make extraction labels and to provide rewards in their RL training phase. In contrast, our model extracts multiple sentences and rewrites them together into a single subject line. We also use word overlap to make extraction labels and use our novel ESQE as a reward function.

5.3.1 Multi-sentence Extractor

For the first stage, we need to select multiple sentences from the email body which contain the necessary information for writing a subject. This task can be formulated as a sequence-to-sequence learning problem where the output sequence corresponds to the position of "positive" sentences in the input email body. Therefore, we use a pointer network (Vinyals et al., 2015) to first build hierarchical sentence representations during encoding and then extract "positive" sentences during decoding.

Suppose our input is an email body D which consists of $|D|$ sentences:

$$D = [d_1, d_2, \ldots, d_j, \ldots, d_{|D|}]$$

We first use a temporal CNN (Kim, 2014a) to build individual sentence representations. For each sentence, we feed the sequence of its word vectors into 1-D convolutional filters with various window sizes. We then apply ReLU activation and then max-over-time pooling. The sentence representation is a concatenation of activations from all filters

$$\mathbf{c}_j = \text{CNN}(d_j), j = 1, \ldots, |D| \tag{5.1}$$

Then we use a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) to capture document-level inter-sentence information over CNN outputs:

$$\begin{aligned}
\overrightarrow{\mathbf{d}}_j &= \text{LSTM}^{\text{forward}}(\overrightarrow{\mathbf{d}}_{j-1}, \mathbf{c}_j) \\
\overleftarrow{\mathbf{d}}_j &= \text{LSTM}^{\text{backward}}(\overleftarrow{\mathbf{d}}_{j+1}, \mathbf{c}_j) \\
\mathbf{d}_j &= [\overrightarrow{\mathbf{d}}_j, \overleftarrow{\mathbf{d}}_j]
\end{aligned} \tag{5.2}$$

For sentence extraction, another LSTM as decoder outputs one "positive" sentence at each time step t. Denoting the decoder hidden state as $\mathbf{h}^t$, we choose a "positive" sentence from a 2-hop attention process. First, we build a context vector $\mathbf{e}^t$ by attending all $\mathbf{d}_j$:

$$\begin{aligned}
\hat{\alpha}_j^t &= \mathbf{v}_e^\intercal \tanh(\mathbf{W}_e \mathbf{d}_j + \mathbf{U}_e \mathbf{h}^t) \\
\alpha^t &= \text{softmax}(\hat{\alpha}^t) \\
\mathbf{e}^t &= \sum_j \alpha_j^t \mathbf{W}_e \mathbf{d}_j
\end{aligned} \tag{5.3}$$

Then, we get an extraction probability distribution o^t over input sentences:

$$\begin{aligned}
\hat{o}_j^t &= \mathbf{v}_o^\intercal \tanh(\mathbf{W}_o \mathbf{d}_j + \mathbf{U}_o \mathbf{e}^t) \\
P(o^t | o^1, o^2, \ldots, o^{t-1}) &= \text{softmax}(\hat{o}^t)
\end{aligned} \tag{5.4}$$

where $\{\mathbf{v}, \mathbf{W}, \mathbf{U}\}$ are trainable parameters.

We also add a trainable "stop" vector with the same dimension as the sentence representation. The decoder can choose to stop by pointing to this "stop" sentence.

5.3.2 Multi-sentence Abstractor

In the second stage, the abstractor takes the selected sentences from the extractor and rewrites them into an email subject. We implement the abstractor as a sequence-to-sequence encoder-decoder model with the bilinear multiplicative attention (Luong et al., 2015) and copy mechanism (See et al., 2017). The copy mechanism enables the decoder to copy words directly from the input document, which is helpful to generate accurate information verbatim even for out-of-vocabulary words.

5.3.3 Email Subject Quality Estimator

Since there is no established automatic metric for SLG, we build our own Email Subject Quality Estimator (ESQE). Given an email body D and a potential subject for the subject s, our quality estimator outputs a real-valued Subject Quality score $\mathrm{SQ}(D, s)$. The email subject and the email body are fed to a temporal CNN.

$$\mathbf{D} = \mathrm{CNN}(D), \mathbf{s} = \mathrm{CNN}(s) \tag{5.5}$$

We concatenate the output of CNNs as the email body and subject pair representation. Then, a single layer feed-forward neural net follows to predict the quality score from the representation.

$$\mathrm{SQ}(D, s) = \mathrm{FFNN}([\mathbf{D}, \mathbf{s}]) \tag{5.6}$$

To train the estimator, we collect human evaluations on 3,490 email subjects. In order to expose the estimator to both good and bad examples, 2,278 of the 3,490 are the original subjects and the remaining 1,212 subjects are generated by an existing summarization system. Each subject has 3 human evaluation scores (the same human evaluation as explained in §5.4.1) and we train our estimator to regress the average.

The inter-annotator agreement is 0.64 by Pearson's r correlation. Even though there is no value range restriction for the estimator output, we found the scores returned by our ESQE after training are bounded from 0.0 to 4.0.

5.3.4 Multi-Stage Training

Supervised Pretraining. We pretrain the extractor and the abstractor separately using supervised learning. To this end, we first create "proxy" sentence labels by checking word overlap between the subject and the body sentence. For each sentence in the body, we label it as "positive" if there is some token overlap of non-stopwords with the subject, negative otherwise. The multi-sentence extractor is trained to predict "positive" sentences by minimizing the cross-entropy loss. For the multi-sentence abstractor, we create training examples by pairing the "positive" sentences and the original subject in the training set. Then the abstractor is trained to generate the subject by maximizing the log-likelihood.

Reinforcement Learning for Extractor. To formulate this RL task at this stage, we treat the extractor as an agent, while the abstractor is pretrained and fixed. The ESQE provides the reward by judging the output subject. At each time step t, it observes a state $s_t = (D, d_{o^{t-1}})$, and samples an action a_t to pick a sentence from the distribution in Equation 5.4:

$$a_t \sim \pi_\theta(s_t, a_t = j) = P(o^t = j) \tag{5.7}$$

where π_θ denotes the policy network described in Section 5.3.1 with a set of trainable parameters θ. The episode is finished in T actions until the extractor picks the "end-of-extraction" signal. Then, the abstractor generates a subject from the extracted sentences and the quality estimator calculates the score. The quality estimator is the reward received by the extractor:

$$r(a_{1:T}) = \text{SQ}(D, s) \tag{5.8}$$

For training, we maximize the expected reward:

$$\mathcal{L}(\theta) = \mathbb{E}_{a_{1:T}\sim\pi_\theta}[r(a_{1:T})] \tag{5.9}$$

with the following gradient given by the REINFORCE algorithm (Williams, 1992):

$$\begin{aligned} \nabla_\theta \mathcal{L}(\theta) &= \mathbb{E}_{\pi_\theta}[\nabla_\theta \log \pi_\theta (r - b)] \\ &\approx \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta (s_t, a_t)(r(a_{1:T}) - b_t) \end{aligned} \tag{5.10}$$

b_t is the baseline reward introduced to reduce the high variance of gradients. The baseline network has the same architecture as the decoder of the extractor. But it has another set of trainable parameters θ_b and predicts the reward by minimizing the following mean squared error:

$$\mathcal{L}(\theta_b) = (b_t - r)^2 \tag{5.11}$$

5.4 Experimental Setup

5.4.1 Evaluation

Automatic Evaluation. Since SLG is a new task, we analyze the usefulness of automatic metrics from sister tasks, and also use human evaluation. We first use automatic metrics from text summarization and machine translation: (1) ROUGE (Lin, 2004) including F1 scores of ROUGE-1, ROUGE-2, and ROUGE-L. (2) METEOR (Denkowski and Lavie, 2014). They all rely on one or more references and measure the similarity between the output and the reference. In addition, we include ESQE, which is a reference-less metric.

Human Evaluation. While those automatic scores are quick and inexpensive to calculate, only our quality estimator is designed for evaluation of subject line generation. Therefore, we also conduct an extensive human evaluation on the *overall* score and two aspects of email quality: informativeness and fluency. An email subject is *informative* if it contains accurate and consistent details with the body, and it is *fluent* if free of grammar errors. We show the email body along with different system outputs as potential subjects (the models

	Dev				Test			
	R-1	R-2	R-L	METEOR	R-1	R-2	R-L	METEOR
LEAD-2	11.28	4.61	10.48	10.76	11.00	4.33	10.20	11.27*
TextRank	11.12	3.75	10.15	9.19	11.32	3.88	10.14	10.64*
LexRank	13.02	4.96	11.89	10.84	12.46	4.62	11.37	11.56*
Shang et al. (2018)	10.56	3.28	9.92	6.17	10.40	3.09	9.77	6.15
See et al. (2017)	18.02	5.73	16.63	10.83	17.02	5.45	15.78	10.31
Paulus et al. (2018)	14.08	5.09	13.36	9.07	13.49	4.55	12.83	8.65
Hsu et al. (2018)	16.59	4.67	15.12	**13.22***	15.75	4.54	14.41	**12.49***
Narayan et al. (2018a)	13.52	3.27	13.33	4.64	12.60	3.09	12.52	4.66
Our System	**25.41**	**11.34**	**25.07**	9.83	**23.67**	**10.29**	**23.44**	9.37
Human Annotation	23.43*	9.71*	22.17	10.87*	23.90*	10.09*	22.75*	11.04*

(a) Against the original subject as reference.

	Dev				Test			
	R-1	R-2	R-L	METEOR	R-1	R-2	R-L	METEOR
LEAD-2	18.88	9.47	17.41	20.70	18.29	8.54	16.62	**20.23**
TextRank	18.29	8.04	16.45	17.00*	17.93	7.47	16.00	16.98*
LexRank	21.82	10.83*	19.78	**20.82**	20.84	9.57*	18.68	19.97
Shang et al. (2018)	16.28	6.14	15.07	12.12	16.11	5.50	14.88	11.81
See et al. (2017)	23.37	7.36	20.99	16.27*	23.31	7.28	20.83	15.68*
Paulus et al. (2018)	15.12	4.62	13.98	10.82	14.56	4.39	13.53	10.37
Hsu et al. (2018)	22.98	7.07	19.95	18.83	22.80	7.09	19.85	18.45
Narayan et al. (2018a)	11.33	1.45	11.14	4.90	11.53	1.37	11.40	5.04
Our System	**25.39**	**10.94**	**24.72**	13.04	**26.11**	**11.43**	**25.64**	13.52
Original Subject	24.38*	10.15*	23.00*	16.49*	24.57	10.40	23.15	14.08
Human Annotation	35.93	17.76	33.55	21.74	36.19	17.75	33.50	21.42

(b) Against two human annotations as reference.

Table 5.3: Automatic metric scores. **bold**: best. underlined: second best. * indicates there is no statistically significant difference from our system with $p < 0.01$ under a paired t-test.

are anonymous). For each subject and each aspect, the human judge chooses a rating from 1 for Poor, 2 for Fair, 3 for Good, 4 for Great. We randomly select 500 samples and have each rated by 3 human judges.

5.4.2 Baselines

To benchmark our method, we use several methods from the summarization field, including some recent state-of-the-art systems, because the email subject line can be viewed as a short summary of the email content. They can be clustered into two groups.

(1) Unsupervised extractive or/and abstractive summarization. LEAD-2 directly uses the first two sentences as the subject line. We choose lead-2 to include both the greeting and the first sentence of main content. **TextRank** (Mihalcea and Tarau, 2004) and **LexRank** (Erkan and Radev, 2004) are two graph-based ranking models to extract the most salient sentence as the subject line. **Shang et al. (2018)** use a graph-based framework to extract topics and then generate a single abstractive sentence for each topic under a budget constraint.

(2) Neural summarization using encoder-decoder networks with attention mechanisms. (Sutskever et al., 2014; Bahdanau et al., 2015). The Pointer-Generator Network from **See et al. (2017)** augments the standard encoder-decoder network by adding the ability to copy words from the source text and using the coverage loss to avoid repetitive generation. **Paulus et al. (2018)** propose neural intra-attention models with a mixed objective of supervised training and policy learning. **Hsu et al. (2018)** extend the pointer-generator network by unifying the sentence-level attention and the word-level attention. **Narayan et al. (2018a)** use a topic-based convolutional neural network to generate extreme summarization for news documents. While they are quite successful in single document summarization, they are mostly extractive, exhibiting a small degree of abstraction (Narayan et al., 2018a). It is unclear how they perform to generate email subject lines of extremely abstractive summarization. We train these models on our dataset.

5.4.3 Implementation Details

Our Model. We pretrain 128-dimensional word2vec (Mikolov et al., 2013) on our corpus as initialization and update word embeddings during training. We use single layer bidirectional LSTMs with 256 hidden units in all models. The convolutional sentence encoders have filters with window sizes (3,4,5) and there are 100 filters for each size. The batch size is 16 for all training phases. We use the Adam optimizer (Kingma and Ba, 2015b) with learning rates of 0.001 for supervised pretraining and 0.0001 for RL. We apply gradient clipping (Pascanu et al., 2013) with L2-norm of 2.0. The training is stopped early if the validation performance is not improved for 3 consecutive epochs. All experiments are performed on a Tesla K80 GPU. All submodels can converge within 1-2 hours and 10 epochs so the whole training takes about 4 hours.

Baselines. For **TextRank** and **LexRank**, we use the sumy[7] implementation which uses the snowball stemmer, the sentence and word tokenizer from NLTK[8]. For **Shang et al. (2018)**, we use their extension of the Multi-Sentence Compression Graph (MSCG) of Filippova (2010) and a budget of 10 words in the submodular maximization. We choose the number of communities from [1,2,3,4,5] based on the dev set and we find that 1 works best. For the Pointer-Generator Network from **See et al. (2017)**, we follow their implementation[9] and use a batch size 16. For **Paulus et al. (2018)**, we use an implementation from Keneshloo et al. (2018)[10]. We did not include the intra-temporal attention and the intra-decoder attention because they hurt the performance. For **Hsu et al. (2018)**, we follow their code[11] with a batch size 16. All training is early stopped based on the dev set performance.

5.5 Results and Discussion

5.5.1 Automatic Metric Evaluation

We report the automatic metric scores against the original subject and the subjects generated by Turkers (human annotations) as references in Tables 5.3a and 5.3b respectively. Table 5.4 also shows the ESQE scores. Overall, our method outperforms the other base-

7. https://github.com/miso-belica/sumy

8. https://www.nltk.org/

9. https://github.com/abisee/pointer-generator

10. https://github.com/yaserkl/RLSeq2Seq

11. https://github.com/HsuWanTing/unified-summarization

lines in all metrics except METEOR. Other systems can achieve higher METEOR scores because METEOR emphasizes recall (recall weighted 9 times more than precision) and other extractive systems such as LexRank can generate longer sentences as subject lines.

In Table 5.3a, where the original subject is the singular reference, the score of our system is rated close to and even higher than the human annotation on both sets. This is because our system is trained on the original subject and is likely a better domain fit. In Table 5.3b, all systems use two human annotations as the reference to have a fair comparison to the human-to-human agreement in the last row. Our system output is actually rated a bit higher than the original subject. This is because the original subject can differ from the human annotation when the sender and the recipient share some background knowledge hidden from the email content. Furthermore, in the last row, the human-to-human agreement is much higher than all the system outputs and the original subject. This indicates that different annotators write subjects with a similar choice of words. In Table 5.4, ESQE still considers our system better than other baselines, while the human annotation has the best quality score.

Evaluation of sub-components. Our extractor captures salient information by selecting multiple sentences from the email body. We measure its performance as a classification problem against the "proxy" sentence labels as explained in Section 5.3.4. The overall precision and recall on the test set is 74% and 42%, respectively. Out of 1906 test examples, 691 examples have more than one sentence selected, and 1626 first sentences and 973 non-first sentences are extracted. Furthermore, during RL training phase, the dev ESQE score increases from 2.30 to 2.40.

5.5.2 Human Evaluation

Table 5.5 shows that our system is rated higher than the baselines on overall, informative, and fluent aspects. For overall scores, the baselines are all between 1.5 and 2.0, indicating the subjects are usually considered as poor or fair (recall that the scale is 1-4, with 4 being

	Dev	Test
LEAD-2	1.56	1.55
TextRank (Mihalcea and Tarau, 2004)	1.59	1.59
LexRank (Erkan and Radev, 2004)	1.57	1.56
Shang et al. (2018)	2.10	2.09
See et al. (2017)	2.22	2.19
Paulus et al. (2018)	<u>2.30</u>	<u>2.30</u>
Hsu et al. (2018)	1.44	1.46
Narayan et al. (2018a)	1.53	1.54
Our System	**2.40**	**2.39**
Original Subject	2.52	2.51
Human Annotation	2.53	2.54

Table 5.4: ESQE score. Compared with our system, all other are statistically significant with $p < 0.01$ under a paired t-test.

	Overall	Informative	Fluent
Random	1.10*	1.45	2.21
See et al. (2017)	1.45*	1.98	1.61
Our System	**2.28**	**2.38**	**2.89**
Original Subject	2.56	2.66	3.11
Human Annotation	2.74*	3.07	2.94

Table 5.5: Human evaluation. * indicates the difference from our system is statistically significant with $p < 0.01$ under a paired t-test.

	Pearson's r	Spearman's ρ
ESQE	**0.49**	**0.46**
ROUGE-1 F1	0.44	0.43
METEOR	0.40	0.40
Inter-Rater Agreement	0.64	0.58

Table 5.6: Correlation analysis between the automatic scores and the human evaluation.

the highest). Our system is 2.28, while the original subject and human annotation are between 2.5 and 3.0. This means more than half of our system outputs are at least fair, and the original subject and human annotation are often good or great. We also find that in 89 out of 500 emails, our system outputs have ratings higher than or equal to the original and human annotated subjects. Furthermore, the raters prefer the human annotated subject to the original subject.

Email Body: Dear Rick, Thanks for speaking with me today. Here is the position description for the KWI President of the Americas Opportunity. I feel that this is a tremendous opportunity to be an integral player with a very exciting relatively early stage Applications Software company, in the very exciting and hot Energy Commodities Sector; They are already profitable, pre-IPO. This position has a great compensation package. Please get back to me if you have an interest or if you know someone who might be intrigued by this opportunity. Thanks, Dal Coger
Original Subject: KWI President of the Americas
Human Annotation: KWI President of the Americas Opportunity
See et al., ACL 2017: Position Description - the Americas Sector Opportunity
Our System: KWI President of the Americas Position

(a) **Email ID:** buy-r_inbox_321

Email Body: Attached for your information are the following two filings made at FERC on Monday on behalf of WPTF: 1.. Motion to Intervene and Protest of the Western Power Trading Forum. This was filed in connection witht the ISO status report filing dealing with creditworthiness issues. 2.. Motion to Intervene and Comments of the Western Power Trading Forum. This was filed in connection with the Reliant and Mirant filing of a joint Section 206 complaint on October 18, 2001. My thanks to those who responded to the drafts with comments and suggestions. Dan
Original Subject: Monday's FERC Filings
Human Annotation: Two Filings Made at FERC
See et al., ACL 2017: FERC filings - FERC power and at monday was filing
Our System: Western Power Trading Filings

(b) **Email ID:** dasovich-j_inbox_1473

Email Body: Hi Evening MBA students, If you plan to graduate this semester for a December 2001 degree, will you please come by the Evening MBA office soon (by Tuesday, September 25 at the latest) and fill out an Application for Candidacy form? We have your fall transcript to assist you in filling out the form. Since we need your original signature, an office visit is best. Thanks, congratulations, and see you!
Original Subject: Planning to graduate this semester?
Human Annotation: December 2001 degree
See et al., ACL 2017: December application(graduate) - September 25
Our System: December 2001 degree application

(c) **Email ID:** dasovich-j_inbox_123

Email Body: As our last day is Friday, November 30th, we would love to toast the good times and special memories that we have shared with you over the past five years. Please join us at Teala's (W. Dallas) on Thursday, November 29th, beginning at 5pm. Looking forward to being with you, Lara and Janel Lara Leibman
Original Subject: Farewell Drinks
Human Annotation: Our last day
See et al., ACL 2017: Friday 30th and day, W. Dallas - November
Our System: Teala's

(d) **Email ID:** arnold-j_inbox_153

Table 5.7: Case study. The sentences extracted by our model are underlined. (a)(b)(c): Our model can generate effective subjects by extracting and rewriting multiple sentences containing salient information. (d): Our model fails to generate reasonable subjects for the novel topic of "farewell" which is not seen during training.

5.5.3 Metric Correlation Analysis

It is important to check if the automatic metric scores can truly reflect the generation quality and serve as valid metrics for subject line generation. Therefore, in Table 5.6, we investigate their correlations with the human evaluation. To this end, we take the average of three human ratings and then calculate Pearson's r and Spearman's ρ between different automatic scores and the average human rating. We also report the inter-rater agreement in the last row by checking the correlation between the third human rating and the average of the other two. We find that the inter-rater agreement is moderate with 0.64 for Pearson's r and 0.58 for Spearman's ρ. We would recommend ESQE because it has the highest correlations while being reference-less.

5.5.4 Case Study

Table 5.7 shows examples of our model outputs. Our model works well by first picking multiple sentences containing information such as named entities and dates and then rewriting them into a succinct subject line preserving the key information. In Example 5.7a, our model extracts sentences with the name of the company and position "KWI President of the Americas". It also captures the importance of the opportunity for this position. Similarly, in Example 5.7b, our model identifies "Western Power Trading" for "filings". In Example 5.7c, our model identifies the date of degree "December 2011" and action item "application". However, we also found our model can fail on emails about novel topics, as in Example 5.7d where the topic is scheduling farewell drinks. Our model only captures the name of the restaurant but not the purpose and topic since it has not seen this kind of email in training.

5.6 Related Work

Past NLP email research has focused on summarization (Muresan et al., 2001; Nenkova and Bagga, 2003; Rambow et al., 2004; Corston-Oliver et al., 2004; Wan and McKeown, 2004; Carenini et al., 2007; Zajic et al., 2008; Carenini et al., 2008; Ulrich et al., 2009), keyword extraction and action detection (Turney, 2000; Bennett and Carbonell, 2005; Dredze et al., 2008; Scerri et al., 2010; Loza et al., 2014; Lahiri et al., 2017; Lin et al., 2018), and classification (Prabhakaran et al., 2014; Prabhakaran and Rambow, 2014; Alkhereyf and Rambow, 2017). However, we could not find any previous work on email subject line generation. The very first study on email summarization is Muresan et al. (2001) who reduce the problem to extracting salient phrases. Later, Nenkova and Bagga (2003), Rambow et al. (2004), Wan and McKeown (2004) deal with the problem of email thread summarization by the sentence extraction approach.

Another related line of research is natural language generation. Our task is most similar to single document summarization because the email subject line can be viewed as a short summary of the email content. Therefore, we use different summarization models as baselines with techniques such as graph-based extraction and compression, sequence-to-sequence neural abstractive summarization with the hierarchical attention, copy, and coverage mechanisms. In addition, RL has become increasingly popular for text generation to optimize the non-differentiable metrics and to reduce the exposure bias in the traditional "teaching forcing" supervised training (Ranzato et al., 2016; Bahdanau et al., 2017; Zhang and Lapata, 2017; Sakaguchi et al., 2017). For example, Narayan et al. (2018b) use RL for ranking sentences in pure extractive summarization. Moreover, most neural summarization work focuses on the single-document setting in the news domain and relies on huge amounts of training data, and much less effort has been made on summarization in other settings or domains with limited amounts of data such as multi-document news summarization (Yasunaga et al., 2017), scientific article summarization incorporating citation information (Yasunaga et al., 2019), and emails.

Furthermore, current methods on news headline generation (Lopyrev, 2015; Tilk and Alumäe, 2017; Kiyono et al., 2017; Tan et al., 2017; Shen et al., 2017) most follow the encoder-decoder model, while our model uses a multi-sentence selection and rewriting framework.

Chapter 6

Improving Low-Resource Cross-lingual Document Retrieval by Reranking with Deep Bilingual Representations

In this chapter, we propose to boost low-resource cross-lingual document retrieval performance with deep bilingual query-document representations. We match queries and documents in both source and target languages with four components, each of which is implemented as a term interaction-based deep neural network with cross-lingual word embeddings as input. By including query likelihood scores as extra features, our model effectively learns to rerank the retrieved documents by using a small number of relevance labels for low-resource language pairs. Due to the shared cross-lingual word embedding space, the model can also be directly applied to another language pair without any training label. Experimental results on the MATERIAL dataset show that our model outperforms the competitive translation-based baselines on English-Swahili, English-Tagalog, and English-Somali cross-lingual information retrieval tasks.

This chapter is based on Zhang et al. (2019a).

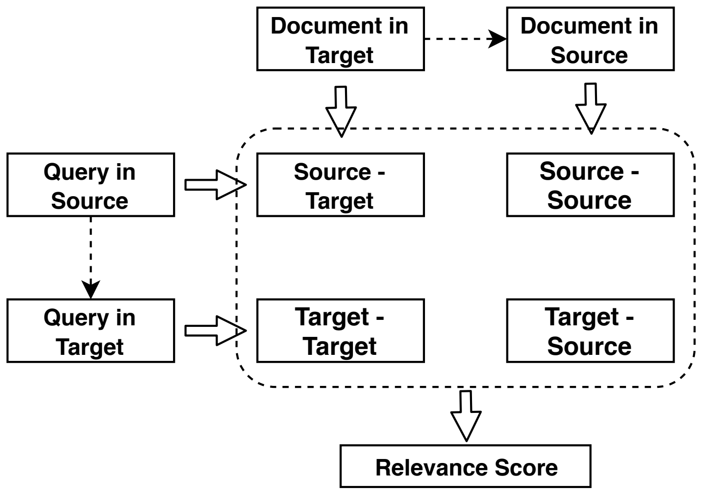


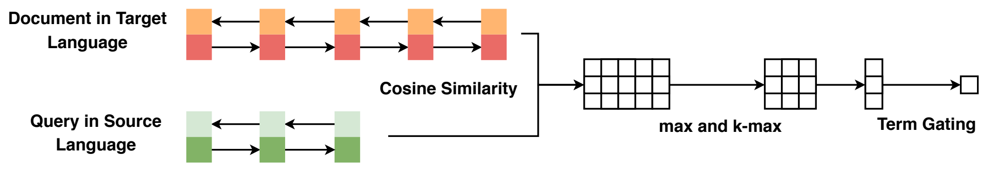
Figure 6.1: Cross-lingual Relevance Ranking with Bilingual Query and Document Representation.

6.1 Introduction

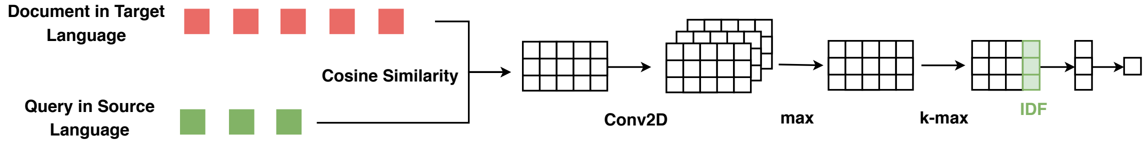
Cross-lingual relevance ranking, or Cross-Lingual Information Retrieval (CLIR), is the task of ranking foreign documents against a user query (Hull and Grefenstette, 1996; Ballesteros and Croft, 1996; Oard and Hackett, 1997; Darwish and Oard, 2003; Oard et al., 2019). As multilingual documents are more accessible, CLIR is increasingly more important whenever the relevant information is in other languages.

Traditional CLIR systems consist of two components: machine translation and monolingual information retrieval. Based on the translation direction, it can be further categorized into the document translation and the query translation approaches (Nie, 2010). In both cases, we first solve the translation problem, and the task is transformed to the monolingual setting. However, while conceptually simple, the performance of this modular approach is fundamentally limited by the quality of machine translation.

Recently, many deep neural IR models have shown promising results on monolingual data sets (Huang et al., 2013; Guo et al., 2016; Pang et al., 2016; Mitra et al., 2016, 2017; Xiong et al., 2017; Hui et al., 2017, 2018; McDonald et al., 2018). They learn a scoring function directly from the relevance label of query-document pairs. However, it is not clear how to use them when documents and queries are not in the same language. Furthermore, those deep neural networks need a large amount of training data. This is expensive to get for low-resource language pairs in our cross-lingual case.



(a) Bilingual POSIT-DRMM. The colored box represents hidden states in bidirectional LSTMs.



(b) Bilingual PACRR-DRMM. The colored box represents cross-lingual word embeddings. Bilingual PACRR is the same except it uses a single MLP at the final stage.

Figure 6.2: Model architecture. We only show the component of the source query with the target document.

In this paper, we propose a cross-lingual deep relevance ranking architecture based on a bilingual view of queries and documents. As shown in Figure 6.1, our model first translates queries and documents and then uses four components to match them in both the source and target language. Each component is implemented as a deep neural network, and the final relevance score combines all components which are jointly trained given the relevance label. We implement this based on state-of-the-art term interaction models because they enable us to make use of cross-lingual embeddings to explicitly encode terms of queries and documents even if they are in different languages. To deal with the small amount of training data, we first perform query likelihood retrieval and include the score as an extra feature in our model. In this way, the model effectively learns to rerank from a small number of relevance labels. Furthermore, since the word embeddings are aligned in the same space, our model can directly transfer to another language pair with no additional training data.

We evaluate our model on the MATERIAL CLIR dataset with three language pairs including English to Swahili, English to Tagalog, and English to Somali. Experimental results demonstrate that our model outperforms other translation-based query likelihood retrieval and monolingual deep relevance ranking approaches.

6.2 Our Method

In cross-lingual document retrieval, given a user query in the source language Q and a document in the target language D, the system computes a relevance score $s(Q, D)$. As shown in Figure 6.1, our model first translates the document as $\hat{D}$ or the query as $\hat{Q}$, and then it uses four separate components to match: (1) source query with target document, (2) source query with source document, (3) target query with source document, (4) target query with target document. The final relevance score combines all components: $s(Q, D) = s(Q, D) + s(Q, \hat{D}) + s(\hat{Q}, \hat{D}) + s(\hat{Q}, D)$

To implement each component, we extend three state-of-the-art *term interaction models*: PACRR (Position-Aware Convolutional Recurrent Relevance Matching) proposed by Hui et al. (2017), POSIT-DRMM (POoled SImilariTy DRMM) and PACRR-DRMM proposed by McDonald et al. (2018). In term interaction models, each query term is scored to a document's terms from the interaction encodings, and scores for different query terms are aggregated to produce the query-document relevance score.

6.2.1 Bilingual POSIT-DRMM

This model is illustrated in Figure 6.2a. We first use bidirectional LSTMs (Hochreiter and Schmidhuber, 1997) to produce the context-sensitive encoding of each query and document term. We also add residual connection to combine the pre-trained term embedding and the LSTM hidden states. For the source query and document term, we can use the pre-trained word embedding in the source language. For the target query and document term, we first align the pre-trained embedding in the target language to the source language and then use this cross-lingual word embedding as the input to LSTM. Thereafter, we produce the document-aware query term encoding by applying max pooling and k-max pooling over the cosine similarity matrix of query and document terms. We then use an MLP to produce term scores, and the relevance score is a weighted sum over all terms in the query with a term gating mechanism.

	EN->SW			EN->TL			EN->SO
# Document	813			844			695
# Document Token (Min/Avg/Max)	34/341/1724			32/404/2501			69/370/2671
Query Set	Q1	Q2	Q3	Q1	Q2	Q3	Q1
# Query	300	400	600	300	400	600	300
# Relevant Pairs	411	489	828	236	576	1018	496

Table 6.1: The MATERIAL dataset statistics. For SW and TL, we use the ANALYSIS document set with Q1 for training, Q2 for dev, and Q3 for test. For transfer learning to SO, we use the DEV document set with Q1. Q1 contains open queries where performers can conduct any automatic or manual exploration while Q2 and Q3 are closed queries where results must be generated with fully automatic systems with no human in the loop.

6.2.2 Bilingual PACRR and Bilingual PACRR-DRMM

These models are shown in Figure 6.2b. We first align the word embeddings in the target language to the source language and build a query-document similarity matrix that encodes the similarity between the query and document term. Depending on the query language and document language, we construct four matrices, $\text{SIM}_{Q,D}$, $\text{SIM}_{Q,\hat{D}}$, $\text{SIM}_{\hat{Q},\hat{D}}$, $\text{SIM}_{\hat{Q},D}$, for each of the four components. Then, we use convolutional neural networks over the similarity matrix to extract n-gram matching features. We then use max-pooling and k-max-pooling to produce the feature matrix where each row is a document-aware encoding of a query term. The final step computes the relevance score: Bilingual PACRR uses an MLP on the whole feature matrix to get the relevance score, while Bilingual PACRR-DRMM first uses an MLP on individual rows to get query term scores and then use a second layer to combine them.

6.3 Related Work

Cross-lingual Information Retrieval. Traditional CLIR approaches include document translation and query translation, and more research efforts are on the latter (Oard and Hackett, 1997; Oard, 1998; McCarley, 1999; Franz et al., 1999). Early methods use the dictionary to translate the user query (Hull and Grefenstette, 1996; Ballesteros and Croft, 1996; Pirkola, 1998). Other methods include the single best SMT query translation (Chin et al., 2014) and the weighted SMT translation alternatives known as the probabilistic structured query (PSQ) (Darwish and Oard, 2003; Ture et al., 2012). Recently, Bai et al. (2010) and Sokolov et al. (2013) propose methods to learn the sparse query-document associations from supervised ranking signals on cross-lingual Wikipedia and patent data, respectively. Furthermore, Vulić and Moens (2015) and Litschko et al. (2018) use cross-lingual word embeddings to represent both queries and documents as vectors and perform IR by computing the cosine similarity. Schamoni et al. (2014) and Sasaki et al. (2018) also use an automatic process to build CLIR datasets from Wikipeida articles.

Neural Learning to Rank. Most of neural learning to rank models can be categorized in two groups: representation based (Huang et al., 2013; Shen et al., 2014) and interaction based (Pang et al., 2016; Guo et al., 2016; Hui et al., 2017; Xiong et al., 2017; McDonald et al., 2018). The former builds representations of query and documents independently, and the matching is performed at the final stage. The latter explicitly encodes the interaction between terms to direct capture word-level interaction patterns. For example, the DRMM (Guo et al., 2016) first compares the term embeddings of each pair of terms within the query and the document and then generates fixed-length matching histograms.

	EN->SW				EN->TL			
	MAP	P@20	NDCG@20	AQWV	MAP	P@20	NDCG@20	AQWV
Query Translation and Document Translation with Indri								
Dictionary-Based Query Translation (DBQT)	20.93	4.86	28.65	6.50	20.01	5.42	27.01	5.93
Probabilistic Structured Query (PSQ)	27.16	5.81	36.03	12.56	35.20	8.18	44.04	19.81
Statistical MT (SMT)	26.30	5.28	34.60	13.77	37.31	8.77	46.77	21.90
Neural MT (NMT)	26.54	5.26	34.83	15.70	33.83	8.20	43.17	18.56
Deep Relevance Ranking								
PACRR	24.69	5.24	32.85	11.73	32.53	8.42	41.75	17.48
PACRR-DRMM	22.15	5.14	30.28	8.50	32.59	8.60	42.17	16.59
POSIT-DRMM	23.91	6.04	33.83	12.06	25.16	8.15	34.80	9.28
Deep Relevance Ranking with Extra Features in Section 6.4								
PACRR	27.03	5.34	35.36	14.18	41.43	8.98	49.96	27.46
PACRR-DRMM	25.46	5.50	34.15	12.18	35.61	8.69	45.34	22.70
POSIT-DRMM	26.10	5.26	34.27	14.11	39.35	9.24	48.41	25.01
Ours with Extra Features in Section 6.4: In-Language Training								
Bilingual PACRR	29.64	5.75	38.27	17.87	43.02	9.63	52.27	29.12
Bilingual PACRR-DRMM	26.15	5.84	35.54	12.92	38.29	9.21	47.60	22.94
Bilingual POSIT-DRMM	**30.13**	**6.28**	**39.68**	**18.69**	**43.67**	**9.73**	**52.80**	**29.12**
Bilingual POSIT-DRMM (3-model ensemble)	**31.60**	**6.37**	**41.25**	**20.19**	**45.35**	**9.84**	**54.26**	**31.08**

Table 6.2: Test set result on English to Swahili and English to Tagalog. We report the TREC ad-hoc retrieval evaluation metrics (MAP, P@20, NDCG@20) and the Actual Query Weighted Value (AQWV).

Train: EN->SW + EN->TL, Test: EN->SO			
	MAP	P@20	AQWV
PSQ	17.52	5.45	2.35
SMT	19.04	6.12	4.62
Bilingual POSIT-DRMM	**20.58**	**6.51**	**5.71**
+3-model ensemble	**21.25**	**6.68**	**5.89**

Table 6.3: Zero-shot transfer learning on English to Somali test set.

6.4 Experiments

Training and Inference. We first use the Indri[12] system which uses query likelihood with Dirichlet Smoothing (Zhai and Lafferty, 2004) to pre-select the documents from the collection. To build the training dataset, for each positive example in the returned list, we randomly sample one negative example from the documents returned by Indri. The model is then trained with a binary cross-entropy loss. On validation or testing set, we use our prediction scores to rerank the documents returned by Indri.

Extra Features. Following the previous work (Severyn and Moschitti, 2015; Mohan et al., 2017; McDonald et al., 2018), we compute the final relevance score by a linear model to

12. www.lemurproject.org/indri.php

combine the model output with the following set of extra features: (1) the Indri score with the language modeling approach to information retrieval. (2) the percentage of query terms with an exact match in the document, including the regular percentage and IDF weighted percentage. (3) the percentage of query term bigrams matches in the document.

Cross-lingual Word Embeddings. We apply the supervised iterative Procrustes approach (Xing et al., 2015; Conneau et al., 2018) to align two pretrained mono-lingual fastText (Bojanowski et al., 2016) word embeddings using the MUSE implementation[13]. To build the bilingual dictionary, we use the translation pages of Wiktionary[14]. For Swahili, we build a training dictionary for 5301 words and a testing dictionary for 1326 words. For Tagalog, the training dictionary and testing dictionary contains 7088 and 1773 words, respectively. For Somali, the corresponding number is 7633 and 1909. We then learn the cross-lingual word embeddings from Swahili to English, from Tagalog to English, and from Somali to English. Therefore, all three languages are in the same word embedding space.

Data Sets and Evaluation Metrics. Our experiments are evaluated on the MATERIAL[15] program as summarized in Table 6.1. It consists of three language pairs with English queries on Swahili (EN->SW), Tagalog (EN->TL), Somali documents (EN->SO).

We use the TREC ad-hoc retrieval evaluation script[16] to compute Precision@20, Mean Average Precision (MAP), Normalized Discounted Cumulative Gain@20 (NDCG@20). We also report the Actual Query Weighted Value (AQWV) (NIST, 2017), a set-based metric with penalty for both missing relevant and returning irrelevant documents. We use $\beta = 40.0$ and find the best global fixed cutoff over all queries.

Baselines. For traditional CLIR approaches, we use query translation and document translation with the Indri system. For query translation, we use Dictionary-Based Query Trans-

13. github.com/facebookresearch/MUSE

14. https://www.wiktionary.org/

15. www.iarpa.gov/index.php/research-programs/material

16. https://trec.nist.gov/trec_eval/

lation (DBQT) and Probabilistic Structured Query (PSQ). For document translation, we use Statistical Machine Translation (SMT) and Neural Machine Translation (NMT). For SMT, we use the moses system (Koehn et al., 2007) with word alignments using mGiza and 5-gram KenLM language model (Heafield, 2011). For NMT, we use sequence-to-sequence model with attention (Bahdanau et al., 2015; Miceli Barone et al., 2017) implemented in Marian (Junczys-Dowmunt et al., 2018).

For deep relevance ranking baselines, we investigate recent state-of-the-art models including PACRR, PACRR-DRMM, and POSIT-DRMM. These models and our methods all use an SMT-based document translation as input.

Implementation Details. For POSIT-DRMM and Bilingual POSIT-DRMM, we use the k-max-pooling with $k = 5$ and 0.3 dropout of the BiLSTM output. For PACRR, PACRR-DRMM and their bilingual counterparts, we use convolutional filter sizes with [1,2,3], and each filter size has 32 filters. We use $k = 2$ in the k-max-pooling. The loss function is minimized using the Adam optimizer (Kingma and Ba, 2015a) with the training batch size as 32. We monitor the MAP performance on the development set after each epoch of training to select the model which is used on the test data.

6.4.1 Results and Discussion

Table 6.2 shows the result on EN->SW and EN->TL where we train and test on the same language pair.

Performance of Baselines. For query translation, PSQ is better than DBQT because PSQ uses a weighted alternative to translate query terms and does not limit to the fixed translation from the dictionary as in DBQT. For document translation, we find that both SMT and NMT have a similar performance which is close to PSQ. The effectiveness of different approaches depends on the language pair (PSQ for EN->SW and SMT for EN->TL), which is a similar finding with McCarley (1999) and Franz et al. (1999). In our experiments with deep relevance ranking models, we all use SMT and PSQ because they have strong performances in both language pairs and it is fair to compare.

Effect of Extra Features and Bilingual Representation. While deep relevance ranking can achieve decent performance, the extra features are critical to achieve better results. Because the extra features include the Indri score, the deep neural model essentially learns to rerank the document by effectively using a small number of training examples. Furthermore, our models with bilingual representations achieve better results in both language pairs, giving additional 1-3 MAP improvements over their counterparts. To compare language pairs, EN->TL has larger improvements over EN->SW. This is because EN->TL has better query translation, document translation, and query likelihood retrieval results from the baselines, and thus it enjoys more benefits from our model. We also found POSIT-DRMM works better than the other two, suggesting term-gating is useful especially when the query translation can provide more alternatives. We then perform ensembling of POSIT-DRMM to further improve the results.

Zero-Shot Transfer Learning. Table 6.3 shows the result for a zero-shot transfer learning setting where we train on EN->SW + EN->TL and directly test on EN->SO without using any Somali relevance labels. This transfer learning delivers a 1-3 MAP improvement over PSQ and SMT. This presents a promising approach to boost performance by utilizing relevance labels from other language pairs.

Chapter 7

Editing-Based SQL Query Generation for Cross-Domain Context-Dependent Questions

In this chapter, we focus on the cross-domain context-dependent text-to-SQL generation task. Based on the observation that adjacent natural language questions are often linguistically dependent and their corresponding SQL queries tend to overlap, we utilize the interaction history by editing the previous predicted query to improve the generation quality. Our editing mechanism views SQL as sequences and reuses generation results at the token level in a simple manner. It is flexible to change individual tokens and robust to error propagation. Furthermore, to deal with complex table structures in different domains, we employ an utterance-table encoder and a table-aware decoder to incorporate the context of the user utterance and the table schema. We evaluate our approach on the SParC dataset and demonstrate the benefit of editing compared with the state-of-the-art baselines which generate SQL from scratch.

This chapter is based on Zhang et al. (2019b).

7.1 Introduction

Generating SQL queries from user utterances is an important task to help end users acquire information from databases. In a real-world application, users often access information in a multi-turn interaction with the system by asking a sequence of related questions. As the interaction proceeds, the user often makes reference to the relevant mentions in the history or omits previously conveyed information assuming it is known to the system.

Therefore, in the context-dependent scenario, the contextual history is crucial to understand the follow-up questions from users, and the system often needs to reproduce partial sequences generated in previous turns. Recently, Suhr et al. (2018) proposes a context-dependent text-to-SQL model including an interaction-level encoder and an attention mechanism over previous utterances. To reuse what has been generated, they propose to copy complete segments from the previous query. While their model is successful to reason about explicit and implicit references, it does not need to explicitly address different database schemas because the ATIS contains only the flight-booking domain. Furthermore, the model is confined to copy whole segments which are extracted by a rule-based procedure, limiting its capacity to utilize the previous query when only one or a few tokens are changed in the segment.

	Context	Cross-Domain	Interaction (train / dev / test)	Question	Turn	Database	Table	Q. Length	Q. Vocab
Spider	✗	✓	11,840 (8,659 / 1,034 / 2,147)	11,840	1.0	200	1020	13.4	4,818
ATIS	✓	✗	1,658 (1,148 / 380 / 130)	11,653	7.0	1	27	10.2	1,582
SParC	✓	✓	4,298 (3,034 / 422 / 842)	12,726	3.0	200	1020	8.1	3,794

Table 7.1: Dataset Statistics.

	WHERE	AGG	GROUP	ORDER	HAVING	SET	JOIN	Nested
Spider	55.2	51.7	24.0	21.5	6.7	5.8	42.9	15.7
ATIS	100	16.6	0.3	0	0	0	96.6	96.6
SParC	42.8	39.8	20.1	17.0	4.7	3.5	35.5	5.7

Table 7.2: % of SQL queries that contain a particular SQL component.

To exploit the correlation between sequentially generated queries and generalize the system to different domains, in this paper, we study an editing-based approach for cross-

Database: student dormitory containing 5 tables.

Goal: Find the first and last names of the students who are living in the dorms that have a TV Lounge as an amenity.

Q1: How many dorms have a TV Lounge?

S1:
```
SELECT COUNT(*) FROM dorm AS T1 JOIN has_amenity AS T2 ON T1.dormid = T2.dormid JOIN
dorm_amenity AS T3
   ON T2.amenid = T3.amenid WHERE T3.amenity_name = 'TV Lounge'
```

Q2: What is the total capacity of these dorms?

S2:
```
SELECT SUM(T1.student_capacity) FROM dorm AS T1 JOIN has_amenity AS T2 ON T1.dormid =
T2.dormid JOIN
   dorm_amenity AS T3 ON T2.amenid = T3.amenid WHERE T3.amenity_name = 'TV Lounge'
```

Q3: How many students are living there?

S3:
```
SELECT COUNT(*) FROM student AS T1 JOIN lives_in AS T2 ON T1.stuid = T2.stuid WHERE
T2.dormid
         IN (SELECT T3.dormid FROM has_amenity AS T3 JOIN dorm_amenity AS T4 ON T3.amenid =
T4.amenid
   WHERE T4.amenity_name = 'TV Lounge')
```

Q4: Please show their first and last names.

S4:
```
SELECT T1.fname, T1.lname FROM student AS T1 JOIN lives_in AS T2 ON T1.stuid = T2.stuid
WHERE T2.dormid
         IN (SELECT T3.dormid FROM has_amenity AS T3 JOIN dorm_amenity AS T4 ON T3.amenid =
T4.amenid
   WHERE T4.amenity_name = 'TV Lounge')
```

Table 7.3: SParC example.

domain context-dependent text-to-SQL generation task. We propose query generation by editing the query in the previous turn. To this end, we first encode the previous query as a sequence of tokens, and the decoder computes a switch to change it at the token level. This sequence editing mechanism models token-level changes and is thus robust to error propagation. Furthermore, to capture the user utterance and the complex database schemas in different domains, we use an utterance-table encoder based on BERT to jointly encode the user utterance and column headers with co-attention, and adopt a table-aware decoder to perform SQL generation with attentions over both the user utterance and column headers.

We evaluate our model on SParC (Yu et al., 2019b), a new large-scale dataset for cross-domain semantic parsing in context consisting of coherent question sequences annotated with SQL queries over 200 databases in 138 domains. Experiment results show that by generating from the previous query, our model delivers an improvement of 7% question match accuracy and 11% interaction match accuracy over the previous state-of-the-art. Further analysis shows that our editing approach is more robust to error propagation than copying segments, and the improvement becomes more significant if the basic text-to-SQL generation accuracy (without editing) improves.

7.2 Cross-Domain Context-Depencent Semantic Parsing

7.2.1 Datasets

We use SParC [17] (Yu et al., 2019b), a large-scale cross-domain context-dependent semantic parsing dataset with SQL labels, as our main evaluation benchmark. A SParC example is shown in Table 7.3. We also report performance on ATIS (Hemphill et al., 1990; Dahl et al., 1994a) for direct comparison to Suhr et al. (2018). In addition, we evaluate the cross-domain context-independent text-to-SQL ability of our model on Spider[18] (Yu et al., 2018c), which SParC is built on.

We summarize and compare the data statistics in Table 7.1 and Table 7.2. While the ATIS dataset has been extensively studied, it is limited to a particular domain. By contrast, SParC is both context-dependent and cross-domain. Each interaction in SParC is constructed using a question in Spider as the interaction goal, where the annotator asks inter-related questions to obtain information that completes the goal. SParC contains interactions over 200 databases and it follows the same database split as Spider where each database appears only in one of train, dev and test sets. In summary, SParC introduces new challenges to context-dependent text-to-SQL because it (1) contains more complex context dependencies, (2) has greater semantic coverage, and (3) adopts a cross-domain task setting.

17. https://yale-lily.github.io/sparc

18. https://yale-lily.github.io/spider

7.2.2 Task Formulation

Let X denote a natural language utterance and Y denote the corresponding SQL query. Context-independent semantic parsing considers individual (X, Y) pairs and maps X to Y. In context-dependent semantic parsing, we consider an interaction I consisting of n utterance-query pairs in a sequence:

$$I = [(X_i, Y_i)]_{i=1}^{n}$$

At each turn t, the goal is to generate Y_t given the current utterance X_t and the interaction history

$$[(X_1, Y_1), (X_2, Y_2), \ldots, (X_{t-1}, Y_{t-1})]$$

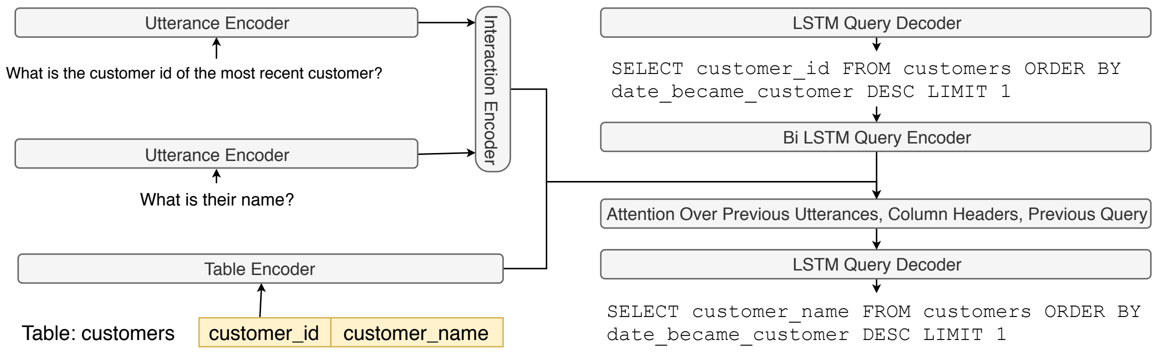
Furthermore, in the cross-domain setting, each interaction is grounded to a different database. Therefore, the model is also given the schema of the current database as an input. We consider relational databases with multiple tables, and each table contains multiple column headers:

$$T = [c_1, c_2, \ldots, c_l, \ldots, c_m]$$

where m is the number of column headers, and each c_l consists of multiple words including its table name and column name (§ 7.3.1).

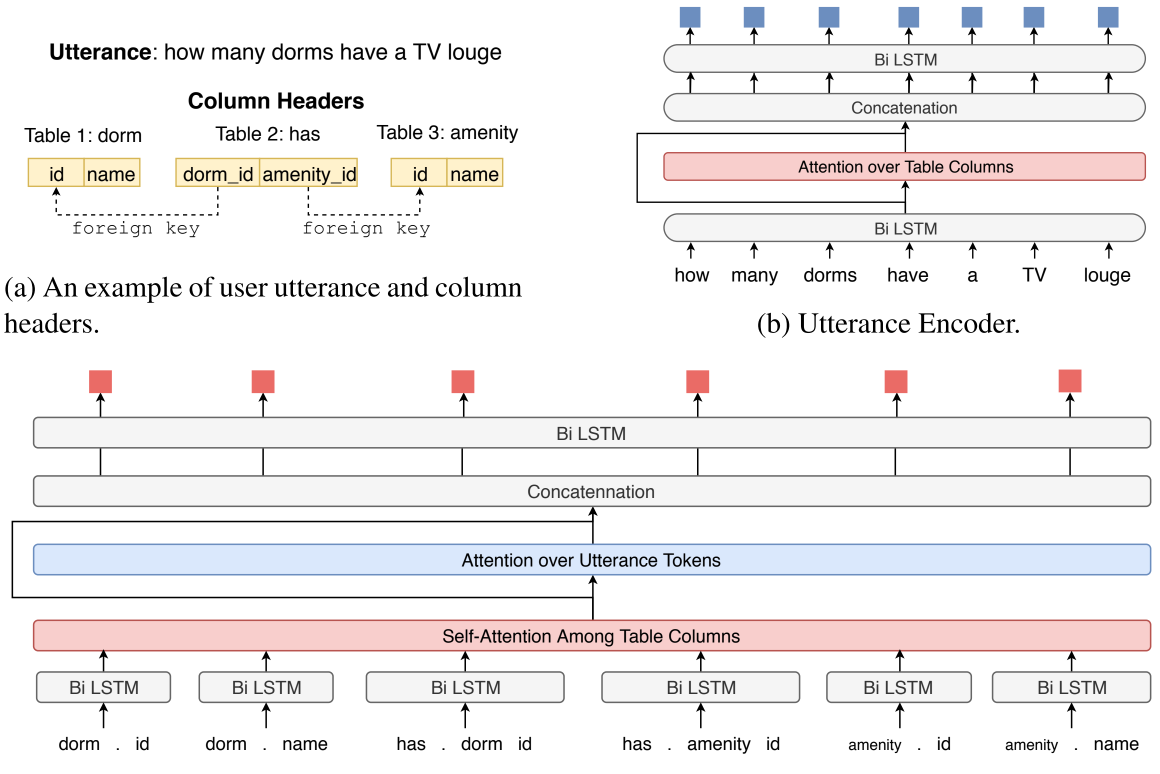
7.3 Methodology

We employ an encoder-decoder architecture with attention mechanisms (Sutskever et al., 2014; Luong et al., 2015) as illustrated in Figure 7.1. The framework consists of (1) an utterance-table encoder to explicitly encode the user utterance and table schema at each turn, (2) A turn attention incorporating the recent history for decoding, (3) a table-aware decoder taking into account the context of the utterance, the table schema, and the previ-



Utterance Encoder
What is the customer id of the most recent customer?
Utterance Encoder
What is their name?
Table Encoder
Table: customers
customer_id
customer_name
Interaction Encoder
LSTM Query Decoder
SELECT customer_id FROM customers ORDER BY date_became_customer DESC LIMIT 1
Bi LSTM Query Encoder
Attention Over Previous Utterances, Column Headers, Previous Query
LSTM Query Decoder
SELECT customer_name FROM customers ORDER BY date_became_customer DESC LIMIT 1

Figure 7.1: Model architecture of editing the previous query with attentions to the user utterances, the table schema, and the previously generated query.



(a) An example of user utterance and column headers.

(b) Utterance Encoder.

(c) Table Encoder.

Figure 7.2: Utterance-Table Encoder for the example in (a).

ously generated query to make editing decisions.

7.3.1 Utterance-Table Encoder

An effective encoder captures the meaning of user utterances, the structure of table schema, and the relationship between the two. To this end, we build an utterance-table encoder with co-attention between the two as illustrated in Figure 7.2.

Figure 7.2b shows the utterance encoder. For the user utterance at each turn, we first use a bi-LSTM to encode utterance tokens. The bi-LSTM hidden state is fed into a dot-product attention layer (Luong et al., 2015) over the column header embeddings. For each utterance token embedding, we get an attention weighted average of the column header embeddings to obtain the most relevant columns (Dong and Lapata, 2018). We then concatenate the bi-LSTM hidden state and the column attention vector, and use a second layer bi-LSTM to generate the utterance token embedding $\mathbf{h}^E$.

Figure 7.2c shows the table encoder. For each column header, we concatenate its table name and its column name separated by a special dot token (i.e., `table name . column name`). Each column header is processed by a bi-LSTM layer. To better capture the internal structure of the table schemas (e.g., foreign key), we then employ a self-attention (Vaswani et al., 2017) among all column headers. We then use an attention layer to capture the relationship between the utterance and the table schema. We concatenate the self-attention vector and the utterance attention vector, and use a second layer bi-LSTM to generate the column header embedding $\mathbf{h}^C$.

Note that the two embeddings depend on each other due to the co-attention, and thus the column header representation changes across different utterances in a single interaction.

Utterance-Table BERT Embedding. We consider two options as the input to the first layer bi-LSTM. The first choice is the pretrained word embedding. Second, we also consider the contextualized word embedding based on BERT (Devlin et al., 2019). To be specific, we follow Hwang et al. (2019) to concatenate the user utterance and all the column headers in a single sequence separated by the `[SEP]` token:

$$\texttt{[CLS]}, X_i, \texttt{[SEP]}, c_1, \texttt{[SEP]}, \ldots, c_m, \texttt{[SEP]}$$

This sequence is fed into the pretrained BERT model whose hidden states at the last layer is used as the input embedding.

7.3.2 Interaction Encoder with Turn Attention

To capture the information across different utterances, we use an interaction-level encoder (Suhr et al., 2018) on top of the utterance-level encoder. At each turn, we use the hidden state at the last time step from the utterance-level encoder as the utterance encoding. This is the input to a uni-directional LSTM interaction encoder:

$$\begin{aligned}
\mathbf{h}_i^U &= \mathbf{h}_{i,|X_i|}^E \\
\mathbf{h}_{i+1}^I &= \text{LSTM}^I(\mathbf{h}_i^U, \mathbf{h}_i^I)
\end{aligned}$$

The hidden state of this interaction encoder $\mathbf{h}^I$ encodes the history as the interaction proceeds.

Turn Attention When issuing the current utterance, the user may omit or explicitly refer to the previously mentioned information. To this end, we adopt the turn attention mechanism to capture correlation between the current utterance and the utterance(s) at specific turn(s). At the current turn t, we compute the turn attention by the dot-product attention between the current utterance and previous utterances in the history, and then add the weighted average of previous utterance embeddings to the current utterance embedding:

$$\begin{aligned}
s_i &= \mathbf{h}_t^U \mathbf{W}_{\text{turn-att}} \mathbf{h}_i^U \\
\alpha^{\text{turn}} &= \operatorname{softmax}(s) \\
\mathbf{c}_t^{\text{turn}} &= \mathbf{h}_t^U + \sum_{i=1}^{t-1} \alpha_i^{\text{turn}} \times \mathbf{h}_i^U
\end{aligned} \tag{7.1}$$

The $\mathbf{c}_t^{\text{turn}}$ summarizes the context information and the current user query and will be used as the initial decoder state as described in the following.

7.3.3 Table-aware Decoder

We use an LSTM decoder with attention to generate SQL queries by incorporating the interaction history, the current user utterance, and the table schema.

Denote the decoding step as k, we provide the decoder input as a concatenation of the embedding of SQL query token $\mathbf{q}_k$ and a context vector $\mathbf{c}_k$:

$$\mathbf{h}^D_{k+1} = \text{LSTM}^D([\mathbf{q}_k; \mathbf{c}_k], \mathbf{h}^D_k)$$

where $\mathbf{h}^D$ is the hidden state of the decoder LSTM^D, and the hidden state $\mathbf{h}^D_0$ is initialized by $\mathbf{c}^{\text{turn}}_t$. When the query token is a SQL keyword, $\mathbf{q}_k$ is a learned embedding; when it is a column header, we use the column header embedding given by the table-utterance encoder as $\mathbf{q}_k$. The context vector $\mathbf{c}_k$ is described below.

Context Vector with the Table and User Utterance. The context vector consists of attentions to both the table and the user utterance. First, at each step k, the decoder computes the attention between the decoder hidden state and the column header embedding.

$$\begin{aligned} s_l &= \mathbf{h}^D_k \mathbf{W}_{\text{column-att}} \mathbf{h}^C_l \\ \alpha^{\text{column}} &= \text{softmax}(s) \\ \mathbf{c}^{\text{column}}_k &= \sum_l \alpha^{\text{column}}_l \times \mathbf{h}^C_l \end{aligned} \tag{7.2}$$

where l is the index of column headers and $\mathbf{h}^C_l$ is its embedding. Second, it also computes the attention between the decoder hidden state and the utterance token embeddings:

$$\begin{aligned} s_{i,j} &= \mathbf{h}^D_k \mathbf{W}_{\text{utterance-att}} \mathbf{h}^E_{i,j} \\ \alpha^{\text{utterance}} &= \text{softmax}(s) \\ \mathbf{c}^{\text{token}}_k &= \sum_{i,j} \alpha^{\text{utterance}}_{i,j} \times \mathbf{h}^E_{i,j} \end{aligned} \tag{7.3}$$

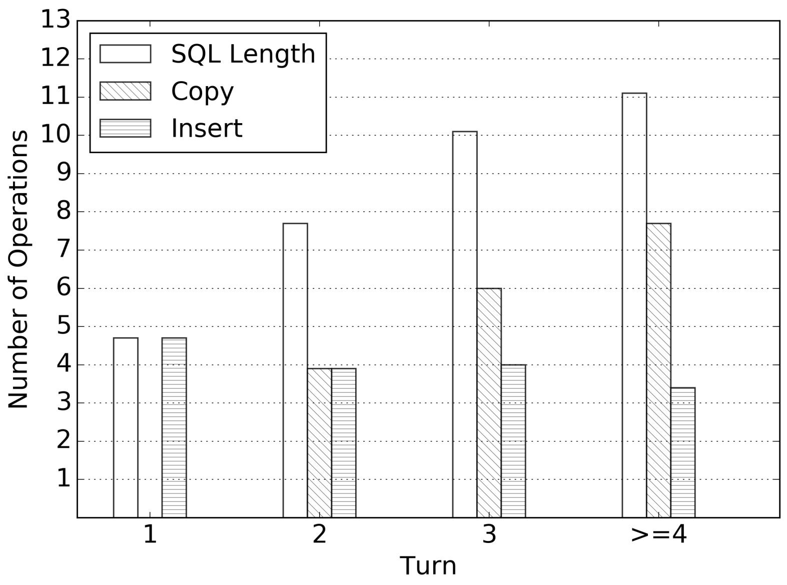


Figure 7.3: The number of operations at different turns.

where i is the turn index, j is the token index, and $\mathbf{h}^E_{i,j}$ is the token embedding for the j-th token of i-th utterance. The context vector $\mathbf{c}_k$ is a concatenation of the two:

$$\mathbf{c}_k = [\mathbf{c}_k^{\text{column}}; \mathbf{c}_k^{\text{token}}]$$

Output Distribution. In the output layer, our decoder chooses to generate a SQL keyword (e.g., `SELECT`, `WHERE`, `GROUP BY`, `ORDER BY`) or a column header. This is critical for the cross-domain setting where the table schema changes across different examples. To achieve this, we use separate layers to score SQL keywords and column headers, and finally use the softmax operation to generate the output probability distribution:

$$\begin{aligned}
\mathbf{o}_k &= \tanh([\mathbf{h}_k^D; \mathbf{c}_k]\mathbf{W}_o) \\
\mathbf{m}^{\text{SQL}} &= \mathbf{o}_k \mathbf{W}_{\text{SQL}} + \mathbf{b}_{\text{SQL}} \\
\mathbf{m}^{\text{column}} &= \mathbf{o}_k \mathbf{W}_{\text{column}} \mathbf{h}^C \\
P(y_k) &= \text{softmax}([\mathbf{m}^{\text{SQL}}; \mathbf{m}^{\text{column}}])
\end{aligned} \tag{7.4}$$

7.3.4 Query Editing Mechanism

In an interaction with the system, the user often asks a sequence of closely related questions to complete the final query goal. Therefore, the query generated for the current turn often overlaps significantly with the previous ones.

To empirically verify the usefulness of leveraging the previous query, we consider the process of generating the current query by applying copy and insert operations to the previous query[19]. Figure 7.3 shows the SQL query length and the number of copy and insert operations at different turns. As the interaction proceeds, the user question becomes more complicated as it requires longer SQL query to answer. However, more query tokens overlap with the previous query, and thus the number of new tokens remains small at the third turn and beyond.

Based on this observation, we extend our table-ware decoder with a query editing mechanism. We first encode the previous query using another bi-LSTM, and its hidden states are the query token embeddings $\mathbf{h}^Q_{i,j'}$ (i.e., the j'-th token of the i-th query). We then extend the context vector with the attention to the previous query:

$$\mathbf{c}_k = [\mathbf{c}_k^{\text{column}}; \mathbf{c}_k^{\text{token}}; \mathbf{c}_k^{\text{query}}]$$

where $\mathbf{c}_k^{\text{query}}$ is produced by an attention to query tokens $\mathbf{h}^Q_{i,j'}$ in the same form as Equation 7.3.

At each decoding step, we predict a switch p_{copy} to decide if we need copy from the previous query or insert a new token.

$$\begin{aligned} p_{\text{copy}} &= \sigma(\mathbf{c}_k \mathbf{W}_{\text{copy}} + \mathbf{b}_{\text{copy}}) \\ p_{\text{insert}} &= 1 - p_{\text{copy}} \end{aligned} \tag{7.5}$$

Then, we use a separate layer to score the query tokens at turn $t-1$, and the output distri-

19. We use a diffing algorithm from https://github.com/paulgb/simplediff

bution is modified as the following to take into account the editing probability:

$$
\begin{aligned}
P_{\text{prev\_SQL}} &= \mathrm{softmax}(\mathbf{o}_k \mathbf{W}_{\text{prev\_SQL}} \mathbf{h}^Q_{t-1}) \\
\mathbf{m}^{\text{SQL}} &= \mathbf{o}_k \mathbf{W}_{\text{SQL}} + \mathbf{b}_{\text{SQL}} \\
\mathbf{m}^{\text{column}} &= \mathbf{o}_k \mathbf{W}_{\text{column}} \mathbf{h}^C \\
P_{\text{SQL} \bigcup \text{column}} &= \mathrm{softmax}([\mathbf{m}^{\text{SQL}}; \mathbf{m}^{\text{column}}]) \\
P(y_k) &= p_{\text{copy}} \cdot P_{\text{prev\_SQL}}(y_k \in \text{prev\_SQL}) \\
&+ p_{\text{insert}} \cdot P_{\text{SQL} \bigcup \text{column}}(y_k \in \text{SQL} \bigcup \text{column})
\end{aligned}
\tag{7.6}
$$

While the copy mechanism has been introduced by Gu et al. (2016) and See et al. (2017), they focus on summarization or response generation applications by copying from the source sentences. By contrast, our focus is on editing the previously generated query while incorporating the context of user utterances and table schemas.

7.4 Related Work

Semantic parsing is the task of mapping natural language sentences into formal representations. It has been studied for decades including using linguistically-motivated compositional representations, such as logical forms (Zelle and Mooney, 1996; Clarke et al., 2010) and lambda calculus (Zettlemoyer and Collins, 2005; Artzi and Zettlemoyer, 2011), and using executable programs, such as SQL queries (Miller et al., 1996; Zhong et al., 2017) and other general-purpose programming languages (Yin and Neubig, 2017; Iyer et al., 2018). Most of the early studies worked on a few domains and small datasets such as GeoQuery (Zelle and Mooney, 1996) and Overnight (Wang et al., 2015b).

Recently, large and cross-domain text-to-SQL datasets such as WikiSQL (Zhong et al., 2017) and Spider (Yu et al., 2018c) have received an increasing amount of attention as many data-driven neural approaches achieve promising results (Dong and Lapata, 2016; Su and

Yan, 2017; Iyer et al., 2017; Xu et al., 2017; Finegan-Dollak et al., 2018; Yu et al., 2018a; Huang et al., 2018; Dong and Lapata, 2018; Sun et al., 2018; Gur et al., 2018; Guo et al., 2018; Yavuz et al., 2018; Shi et al., 2018). Most of them still focus on *context-independent* semantic parsing by converting single-turn questions into executable queries.

Relatively less effort has been devoted to *context-dependent* semantic parsing on datasets including ATIS (Hemphill et al., 1990; Dahl et al., 1994b), SpaceBook (Vlachos and Clark, 2014), SCONE (Long et al., 2016; Guu et al., 2017; Fried et al., 2018; Suhr and Artzi, 2018; Huang et al., 2019), SequentialQA (Iyyer et al., 2017), SParC (Yu et al., 2019b) and CoSQL (Yu et al., 2019a). On ATIS, Miller et al. (1996) maps utterances to semantic frames which are then mapped to SQL queries; Zettlemoyer and Collins (2009) starts with context-independent Combinatory Categorial Grammar (CCG) parsing and then resolves references to generate lambda-calculus logical forms for sequences of sentences. The most relevant to our work is Suhr et al. (2018), who generate ATIS SQL queries from interactions by incorporating history with an interaction-level encoder and copying segments of previously generated queries. Furthermore, SCONE contains three domains using stack- or list-like elements and most queries include a single binary predicate. SequentialQA is created by decomposing some complicated questions in WikiTableQuestions (Pasupat and Liang, 2015). Since both SCONE and SequentialQA are annotated with only denotations but not query labels, they don't include many questions with rich semantic and contextual types. For example, SequentialQA (Iyyer et al., 2017) requires that the answer to follow-up questions must be a subset of previous answers, and most of the questions can be answered by simple SQL queries with `SELECT` and `WHERE` clauses.

Concurrent with our work, Yu et al. (2019a) introduced CoSQL, a large-scale cross-domain conversational text-to-SQL corpus collected under the Wizard-of-Oz setting. Each dialogue in CoSQL simulates a DB querying scenario with a crowd worker as a user and a college computer science student who is familiar with SQL as an expert. Question-SQL pairs in CoSQL reflect greater diversity in user backgrounds compared to other corpora and involve frequent changes in user intent between pairs or ambiguous questions that require user clarification. These features pose new challenges for text-to-SQL systems.

Our work is also related to recently proposed approaches to code generation by editing (Hayati et al., 2018; Yin et al., 2019; Hashimoto et al., 2018). While they follow the framework of generating code by editing the relevant examples retrieved from training data, we focus on a context-dependent setting where we generate queries from the previous query predicted by the system itself.

7.5 Experimental Results

7.5.1 Metrics

On both Spider and SParC, we use the exact set match accuracy between the gold and the predicted queries [20]. To avoid ordering issues, instead of using simple string matching, Yu et al. (2018c) decompose predicted queries into different SQL clauses such as `SELECT`, `WHERE`, `GROUP BY`, and `ORDER BY` and compute scores for each clause using set matching separately. On SparC, we report two metrics: question match accuracy which is the score average over all questions and interaction match accuracy which is average over all interactions.

7.5.2 Baselines

SParC. We compare with the two baseline models released by Yu et al. (2019b).

(1) Context-dependent Seq2Seq (CD-Seq2Seq): This model is adapted from Suhr et al. (2018). The original model was developed for ATIS and does not take the database schema as input hence cannot generalize well across domains. Yu et al. (2019b) adapt it to perform context-dependent SQL generation in multiple domains by adding a bi-LSTM database

20. More details at https://github.com/taoyds/spider/tree/master/evaluation_examples

schema encoder which takes bag-of-words representations of column headers as input. They also modify the decoder to select between a SQL keyword or a column header.

(2) SyntaxSQL-con: This is adapted from the original context-agnostic SyntaxSQL-Net (Yu et al., 2018b) by using bi-LSTMs to encode the interaction history including the utterance and the associated SQL query response. It also employs a column attention mechanism to compute representations of the previous question and SQL query.

Spider. We compare with the results as reported in Yu et al. (2018b). Furthermore, we also include recent results from Lee (2019) who propose to use recursive decoding procedure, Bogin et al. (2019) introducing graph neural networks for encoding schemas, and Guo et al. (2019) who achieve state-of-the-art performance by using an intermediate representation to bridge natural language questions and SQL queries.

7.5.3 Implementation Details

Our model is implemented in PyTorch (Paszke et al., 2017). We use pretrained 300-dimensional GloVe (Pennington et al., 2014) word embedding. All LSTM layers have 300 hidden size, and we use 1 layer for encoder LSTMs, and 2 layers for decoder LSTMs. We use the ADAM optimizer (Kingma and Ba, 2015b) to minimize the token-level cross-entropy loss with a batch size of 16. Model parameters are randomly initialized from a uniform distribution $U[-0.1, 0.1]$. The main model has an initial learning rate of 0.001 and it will be multiplied by 0.8 if the validation loss increases compared with the previous epoch. When using BERT instead of GloVe, we use the pretrained small uncased BERT model with 768 hidden size[21], and we fine tune it with a separate constant learning rate of 0.00001. The training typically converges in 10 epochs.

21. https://github.com/google-research/bert

	Dev Set	Test Set
SQLNet (Xu et al., 2017)	10.9	12.4
SyntaxSQLNet (Yu et al., 2018b)	18.9	19.7
+data augmentation (Yu et al., 2018b)	24.8	27.2
Lee (2019)	28.5	24.3
GNN (Bogin et al., 2019)	40.7	39.4
IRNet (Guo et al., 2019)	53.2	46.7
IRNet (BERT) (Guo et al., 2019)	61.9	54.7
Ours	36.4	32.9
+ utterance-table BERT Embedding	57.6	53.4

Table 7.4: Spider results on dev set and test set.

	Question Match		Interaction Match	
	Dev	Test	Dev	Test
SyntaxSQL-con (Yu et al., 2019b)	18.5	20.2	4.3	5.2
CD-Seq2Seq (Yu et al., 2019b)*	21.9	23.2	8.1	7.5
+ segment copy (w/ predicted query)	21.7	20.3	9.5	8.1
+ segment copy (w/ gold query)	27.3	26.7	10.0	8.4
Ours	31.4	–	14.7	–
+ query attention and sequence editing (w/ predicted query)	33.0	–	16.4	–
+ query attention and sequence editing (w/ gold query)	40.6	–	17.3	–
Ours + utterance-table BERT Embedding	40.4	–	18.1	–
+ query attention (w/ predicted query)	42.7	–	21.6	–
+ query attention and sequence editing (w/ predicted query)	**47.2**	**47.9**	**29.5**	**25.3**
+ query attention and sequence editing (w/ gold query)	53.4	54.5	29.2	25.0

Table 7.5: SParC results. For our models, we only report test set results of our best model on the dev set. *We improve the CD-Seq2Seq performance over Yu et al. (2019b) by separating and parsing the column names (e.g., stu_fname → student first name) and using the schema-specific output vocabulary during decoding.

	Dev Set			Test Set		
	Query	Relaxed	Strict	Query	Relaxed	Strict
FULL (Suhr et al., 2018)	$37.5_{\pm 0.9}$	$63.0_{\pm 0.7}$	$62.5_{\pm 0.9}$	$43.6_{\pm 1.0}$	$69.3_{\pm 0.8}$	$69.2_{\pm 0.8}$
Ours	36.2	60.5	60.0	43.9	68.5	68.1

Table 7.6: ATIS results on dev set and test set.

7.5.4 Overall Results

Spider. Table 7.4 shows the results on Spider dataset. Since each question is standalone, we don't use interaction-level decoder or query editing. Our method can achieve the performance of 36.4% on dev set and 32.9% on test set, serving as a strong model for the context-independent cross-domain text-to-SQL generation. This demonstrates the effec-

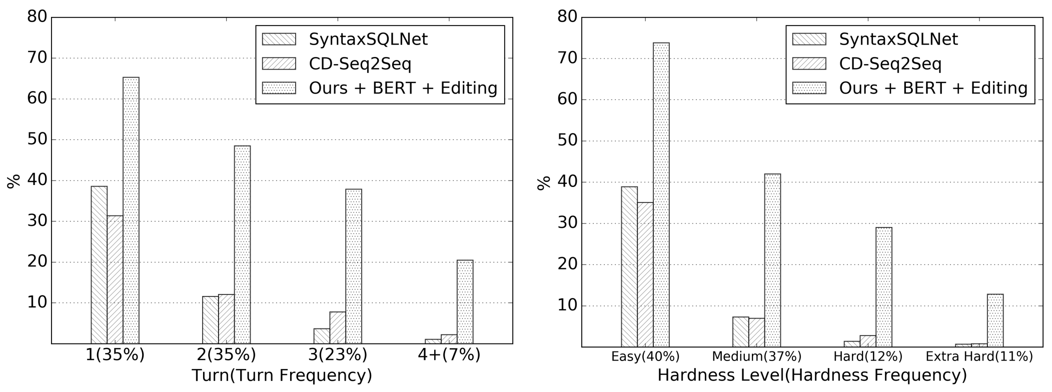


Figure 7.4: Performance split by different turns (Left) and hardness levels (Right) on SParC dev set.

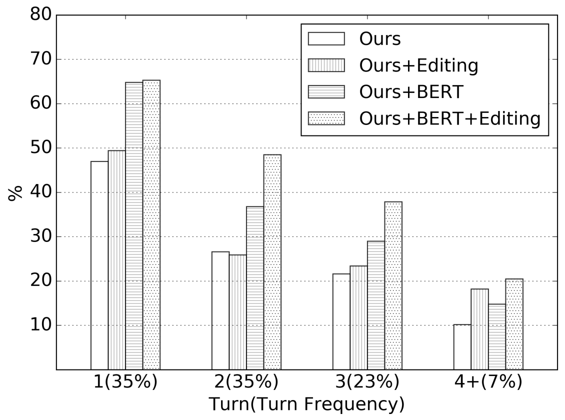


Figure 7.5: Effect of query editing at different turns on SParC dev set.

tiveness of our utterance-table encoder and table-aware decoder to handle the semantics of user utterances and the complexity of table schemas to generate complex SQL queries in unseen domains.

Furthermore, adding the utterance-table BERT embedding gives significant improvement, achieving 57.6% on dev set and 53.4% on test set, which is comparable to the state-of-the-art results from IRNet with BERT. We attribute our BERT model's high performance to (1) the empirically powerful text understanding ability of pretrained BERT model and (2) the early interaction between utterances and column headers when they are concatenated in a single sequence as the BERT input.

SParC. Table 7.5 shows the results on SParC dataset. Similar to Spider, our model without previous query as input already outperforms SyntaxSQL-con, achieving 31.4% question matching accuracy and 14.7% interaction matching accuracy. In addition, compared with CD-Seq2Seq, our model enjoys the benefits of the table-utterance encoder, turn attention, and the joint consideration of utterances and table schemas during the decoding stage. This boosts the performance by 10% question accuracy and 6% interaction accuracy.

Furthermore, we also investigate the effect of copying segment. We use the same segment copy procedure as Suhr et al. (2018): first deterministically extract segments from the previous query and encode each segment using an LSTM, then generate a segment by computing its output probability based on its segment encoding. However, since the segment extraction from Suhr et al. (2018) is exclusively designed for the ATIS dataset, we implement our own segment extraction procedure by extracting `SELECT`, `FROM`, `GROUP BY`, `ORDER BY` clauses as well as different conditions in `WHERE` clauses. In this way, 3.9 segments can be extracted per SQL on average. We found that adding segment copying to CD-Seq2Seq gives a slightly lower performance on question matching and a small gain on interaction matching, while using segments extracted from the gold query can have much higher results. This demonstrates that segment copy is vulnerable to error propagation. In addition, it can only copy whole segments hence has difficulty capturing the changes of only one or a few tokens in the query.

To better understand how models perform as the interaction proceeds, Figure 7.4 (Left) shows the performance split by turns on the dev set. The questions asked in later turns are more difficult to answer given longer context history. While the baselines have lower performance as the turn number increases, our model still maintains 38%-48% accuracy for turn 2 and 3, and 20% at turn 4 or beyond. Similarly, Figure 7.4 (Right) shows the performance split by hardness levels with the frequency of examples. This also demonstrates our model is more competitive in answering hard and extra hard questions.

ATIS. We also report our model performance on ATIS in Table 7.6. Our model achieves 36.2% dev and 43.9% test string accuracy, comparable to Suhr et al. (2018). On ATIS, we only apply our editing mechanism and reuse their utterance encoder instead of the BERT utterance-table encoder, because ATIS is single domain.

7.5.5 Effect of Query Editing

We further investigate the effect of our query editing mechanism. To this end, we apply editing from both the gold query and the predicted query on our model with or without the utterance-table BERT embedding. We also perform an ablation study to validate the contribution of query attention and sequence editing separately.

As shown in Table 7.5, editing the gold query consistently improves both question match and interaction match accuracy. This shows the editing approach is indeed helpful to improve the generation quality when the previous query is the oracle.

Using the predicted query is a more realistic setting, and in this case, the model is affected by error propagation due to the incorrect queries produced by itself. For the model without the utterance-table BERT embedding, using the predicted query only gives around 1.5% improvement. As shown in Figure 7.5, this is because the editing mechanism is more helpful for turn 4 which is a small fraction of all question examples. For the model with the utterance-table BERT embedding, the query generation accuracy at each turn is significantly improved, thus reducing the error propagation effect. In this case, the editing approach delivers consistent improvements of 7% increase on question matching accuracy and 11% increase on interaction matching accuracy. Figure 7.5 also shows that query editing with BERT benefits all turns.

Finally, as an ablation study, Table 7.5 also reports the result with only query attention (use predicted query) on the dev set. This improves over our vanilla BERT model without query attention and achieves 42.7% question and 21.6% interaction matching accuracy. With query editing, our best model further improves to 47.2% question and 29.5% interaction matching accuracy. This demonstrates the effectiveness of our query attention and query editing separately, both of which are essential to make use of the previous query.

Chapter 8

Conclusions and Future Work

In this thesis, we recognize the importance of contexts of language. We propose to address the challenges of comprehension and leverage of contexts for language understanding, generation, and grounding. We present several deep neural network based models to understand the meaning of language in various domains and contexts and to generate language responses for information request and summarization and grounded structure query representations for human-computer communication.

To enable machines to understand language at different levels of granularity, Chapter 2, Chapter 3, and Chapter 4 design deep neural networks for text units. Chapter 2 presents an end-to-end neural network for language understanding at levels of mentions and entities by performing coreference resolution in documents. It considers all text spans as potential mentions and links an antecedent for each possible mention using a biaffine attention model, and the network is trained end-to-end by jointly optimizing the mention detection accuracy and the mention clustering log-likelihood. Then, in Chapter 3, we describe a general-purpose text classification neural model at levels of sentences and documents. It processes pretrained word embeddings via Long Short-Term Memory networks and subsequently extracts features with convolution operators. In this way, it captures both the dependency information within each sentence and relationships across sentences in the same document. In Chapter 4, we step further to the level of dialogs by studying the problem of addressee and response selection in multi-turn multi-party conversations. To tackle the challenge of conversation with multiple speakers in different roles, we propose the Speaker Interaction Recurrent Neural Network with a dialog encoder to update speaker embeddings in a role-sensitive way.

In Chapter 5, we then study the language generation in text summarization domains. To be specific, we propose the task of subject line generation, which aims to automatically produce email subjects given the email body. We build Annotated Enron Subject Line Corpus, the first dataset for this task, and propose both an evaluation metric and a data-efficient model. We first build a neural network for Email Subject Quality Estimator (ESQE) to score the quality of an email subject given the email body, where we show that ESQE has a higher correlation with human evaluation than metrics based on n-gram matching. We then combine extractive and abstractive approaches to summarize messages to short subjects with a high compression ratio by optimizing quality estimation scores via reinforcement learning.

Chapter 5 presents a model for retrieving information relevant to a users query from documents in different low-resource languages. We propose a deep relevance ranking model to combine different translations by using cross-lingual word embeddings in a low-resource setting. To deal with small amounts of training data, we first perform query likelihood retrieval and include the score as an extra feature in the model.

Finally, Chapter 7 investigates the semantic parsing in contexts of multi-turn user utterances by grounding to SQL query for relational databases. Based on the observation that adjacent natural language questions are often linguistically dependent and their corresponding SQL queries tend to overlap, we utilize the interaction history by editing the previous predicted query to improve the generation quality. To generate SQL queries across different domains, we employ an utterance-table encoder and a table-aware decoder to incorporate the context of the user utterance and the table schema.

8.1 Future Work

Even though the past few years have seen tremendous progress in understanding and generating natural language, we have only scratched the surface of what can be achieved in developing intelligent systems. Built upon my past work, I plan to embrace important open challenges by exploring the following new directions.

Grounding Language to Interactive User Actions. Mobile phone applications have been widely used in daily life. To accomplish a job on the mobile phone, the user interacts with an application via a sequence of low-level user interface actions, such as click an element, swipe to check the rest of a list, and type a string. Due to their diversity in content and design, mobile phone applications provide an open-domain learning environment with a rich source of real-world knowledge. We are interested in mapping natural language instructions to sequences of user interface actions. In particular, we consider a task setting where the agent only receives a high-level user goal in natural language without step-by-step instructions or human demonstration, and it needs to perform a sequence of user interface actions to accomplish the user goal. This would automate the task completion that currently requires complicated human participation, greatly promoting the efficiency and accessibility for mobile application users. The complex representation of a mobile phone application contains both structured properties (e.g., the internal tree representation of a screen and the spatial relations among elements) and unstructured elements (e.g., text and image icons). Moreover, since we do not provide any human demonstration or step-by-step annotation, the agent needs to reason about the semantics of a high-level user goal and the context of screens, and then learns to perform the correct sequence of actions. Therefore, this presents a challenging natural language grounding and navigation problem in this diverse and open-domain platform.

Multilingual Transfer Learning for Low-resource Languages. Deep neural networks, despite their success to achieve promising results in various natural language processing tasks, still heavily rely on large amounts of in-domain labeled training data. However, only a few languages have large amounts of labeled data, and generalization in low-resource scenarios is still an open challenge. I would like to develop multilingual transfer learning techniques that can leverage annotations in high-resource languages to boost the performance of low-resource languages. I am particularly interested in models that (1) require minimal cross-lingual supervision, (2) leverage knowledge from multiple source languages, and (3) quickly adapt to the target language and task.

Natural Language for Interpretability and Accessibility. While deep learning has become the de facto approach to build intelligent systems, the improvement of performance often comes at a cost of interpretability. Complex neural networks permit easy architectural and operational variations for state-of-the-art accuracy, yet they provide little transparency about their inner decision-making mechanisms. I believe natural language can promote interpretable AI: language is not only the means of communication between humans, but it also offers a medium for the intelligent system to explain and rationalize its solutions. To this end, I would like to empower the intelligent systems with abilities to automatically extract or generate human-readable language explanations to justify their predictions or actions. Furthermore, I also plan to use natural language to improve the accessibility of data and AI. I hope natural language can serve as a bridge between humans and machines to help users access data more efficiently, recognize the objects in images and graphic user interfaces for visually impaired people, converse and interact with machines to accomplish daily tasks, etc.

Controllable and Personalized Text Generation. While deep learning models have shown promising results in text generation tasks such as summarization, translation, and dialog response generation, progress remains to be made towards controllable and personalized text generation. In particular, I would like develop more controllable models such that (1) the generated text stay faithful to the conditioned text input, (2) we can manipulate and transfer the output text attributes (such as formal v.s. informal style, positive v.s. negative sentiment), (3) we should remove social bias and abusive content. Furthermore, I would like also to incorporate personal information such as gender, age, social context, and knowledge background to generate text suitable to individual users.

Bibliography

Sakhar Alkhereyf and Owen Rambow. Work hard, play hard: Email classification on the avocado and enron corpora. In *Proceedings of TextGraphs-11: the Workshop on Graph-based Methods for Natural Language Processing*, 2017.

Yoav Artzi and Luke Zettlemoyer. Bootstrapping semantic parsers from conversations. In *Proceedings of the conference on empirical methods in natural language processing*, pages 421–432. Association for Computational Linguistics, 2011.

Amit Bagga and Breck Baldwin. Algorithms for scoring coreference chains. In *The first international conference on language resources and evaluation workshop on linguistics coreference*, 1998.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*, 2014.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. In *ICLR*, 2015.

Dzmitry Bahdanau, Philemon Brakel, Kelvin Xu, Anirudh Goyal, Ryan Lowe, Joelle Pineau, Aaron Courville, and Yoshua Bengio. An actor-critic algorithm for sequence prediction. In *ICLR*, 2017.

Bing Bai, Jason Weston, David Grangier, Ronan Collobert, Kunihiko Sadamasa, Yanjun

Qi, Olivier Chapelle, and Kilian Weinberger. Learning to rank with (a lot of) word features. *Information retrieval*, 13(3):291–314, 2010.

Lisa Ballesteros and Bruce Croft. Dictionary methods for cross-lingual information retrieval. In *International Conference on Database and Expert Systems Applications*, pages 791–801. Springer, 1996.

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166, 1994.

Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Janvin. A neural probabilistic language model. *The Journal of Machine Learning Research*, 3:1137–1155, 2003.

Eric Bengtson and Dan Roth. Understanding the value of features for coreference resolution. In *EMNLP*, 2008.

Paul N Bennett and Jaime Carbonell. Detecting action-items in e-mail. In *SIGIR*, 2005.

Anders Björkelund and Jonas Kuhn. Learning structured perceptrons for coreference resolution with latent antecedents and non-local features. In *ACL*, 2014.

Ben Bogin, Matt Gardner, and Jonathan Berant. Representing schema structure with graph neural networks for text-to-sql parsing. In *ACL*, 2019.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. Enriching word vectors with subword information. *arXiv preprint arXiv:1607.04606*, 2016.

Antoine Bordes and Jason Weston. Learning end-to-end goal-oriented dialog. In *ICLR*, 2017.

Giuseppe Carenini, Raymond T Ng, and Xiaodong Zhou. Summarizing email conversations with clue words. In *WWW*, 2007.

Giuseppe Carenini, Raymond T Ng, and Xiaodong Zhou. Summarizing emails with conversational cohesion and subjectivity. *ACL*, 2008.

Po-Chun Chen, Ta-Chung Chi, Shang-Yu Su, and Yun-Nung Chen. Dynamic time-aware attention to speaker roles and contexts for spoken language understanding. In *ASRU*, 2017.

Yen-Chun Chen and Mohit Bansal. Fast abstractive summarization with reinforce-selected sentence rewriting. In *ACL*, 2018.

Jianpeng Cheng and Mirella Lapata. Neural summarization by extracting sentences and words. In *ACL*, 2016.

Ta-Chung Chi, Po-Chun Chen, Shang-Yu Su, and Yun-Nung Chen. Speaker role contextual modeling for language understanding and dialogue policy learning. In *IJCNLP*, 2017.

Jeffrey Chin, Maureen Heymans, Alexandre Kojoukhov, Jocelyn Lin, and Hui Tan. Cross-language information retrieval, August 5 2014. US Patent 8,799,307.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. Learning phrase representations using rnn encoder–decoder for statistical machine translation. In *EMNLP*, 2014.

Jinho D Choi and Martha Palmer. Guidelines for the clear style constituent to dependency conversion. Technical report, Technical Report 01-12, University of Colorado at Boulder, 2012.

Junyoung Chung, Caglar Gulcehre, KyungHyun Cho, and Yoshua Bengio. Empirical evaluation of gated recurrent neural networks on sequence modeling. *NIPS 2014 Deep Learning and Representation Learning Workshop*, 2014.

Kevin Clark and Christopher D. Manning. Entity-centric coreference resolution with model stacking. In *ACL*, 2015.

Kevin Clark and Christopher D. Manning. Deep reinforcement learning for mention-ranking coreference models. In *EMNLP*, 2016a.

Kevin Clark and Christopher D. Manning. Improving coreference resolution by learning entity-level distributed representations. In *ACL*, 2016b.

James Clarke, Dan Goldwasser, Ming-Wei Chang, and Dan Roth. Driving semantic parsing from the world's response. In *Proceedings of the fourteenth conference on computational natural language learning*, pages 18–27. Association for Computational Linguistics, 2010.

Ronan Collobert and Jason Weston. A unified architecture for natural language processing: Deep neural networks with multitask learning. In *Proceedings of ICML*, pages 160–167. ACM, 2008.

Alexis Conneau, Guillaume Lample, Marc'Aurelio Ranzato, Ludovic Denoyer, and Hervé Jégou. Word translation without parallel data. In *ICLR*, 2018.

Simon Corston-Oliver, Eric Ringger, Michael Gamon, and Richard Campbell. Task-focused summarization of email. *Text Summarization Branches Out*, 2004.

Deborah A. Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. Expanding the scope of the atis task: The atis-3 corpus. In *Proceedings of the Workshop on Human Language Technology*, HLT '94, Stroudsburg, PA, USA, 1994a. Association for Computational Linguistics.

Deborah A Dahl, Madeleine Bates, Michael Brown, William Fisher, Kate Hunicke-Smith, David Pallett, Christine Pao, Alexander Rudnicky, and Elizabeth Shriberg. Expanding the scope of the atis task: The atis-3 corpus. In *Proceedings of the workshop on Human Language Technology*, pages 43–48. Association for Computational Linguistics, 1994b.

George E Dahl, Ryan P Adams, and Hugo Larochelle. Training restricted boltzmann machines on word observations. *arXiv preprint arXiv:1202.5695*, 2012.

Andrew M Dai and Quoc V Le. Semi-supervised sequence learning. *arXiv preprint arXiv:1511.01432*, 2015.

Kareem Darwish and Douglas W Oard. Probabilistic structured query methods. In *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval*, pages 338–344. ACM, 2003.

Hal Daumé III and Daniel Marcu. A large-scale exploration of effective global features for a joint entity detection and tracking model. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, 2005.

Pascal Denis and Jason Baldridge. Joint determination of anaphoricity and coreference resolution using integer programming. In *NAACL*, 2007a.

Pascal Denis and Jason Baldridge. A ranking approach to pronoun resolution. In *IJCAI*, 2007b.

Michael Denkowski and Alon Lavie. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the ninth workshop on statistical machine translation*, 2014.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019.

Li Dong and Mirella Lapata. Language to logical form with neural attention. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*, 2016.

Li Dong and Mirella Lapata. Coarse-to-fine decoding for neural semantic parsing. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018.

Cícero Nogueira dos Santos and Bianca Zadrozny. Learning character-level representations for part-of-speech tagging. In *Proceedings of the 31st International Conference on Machine Learning (ICML), JMLR: W&CP volume 32*, 2014.

Timothy Dozat and Christopher D Manning. Deep biaffine attention for neural dependency parsing. In *ICLR*, 2017.

Mark Dredze, Hanna M Wallach, Danny Puller, and Fernando Pereira. Generating summary keywords for emails using topics. In *Proceedings of the 13th international conference on Intelligent user interfaces*, pages 199–206. ACM, 2008.

Greg Durrett and Dan Klein. Easy victories and uphill battles in coreference resolution. In *EMNLP*, 2013.

Greg Durrett and Dan Klein. A joint model for entity analysis: Coreference, typing, and linking. *Transactions of the Association for Computational Linguistics*, 2:477–490, 2014.

Günes Erkan and Dragomir R Radev. Lexrank: Graph-based lexical centrality as salience in text summarization. *journal of artificial intelligence research*, 22:457–479, 2004.

Eraldo Rezende Fernandes, Cícero Nogueira Dos Santos, and Ruy Luiz Milidiú. Latent structure perceptron with feature induction for unrestricted coreference resolution. In *Joint Conference on EMNLP and CoNLL-Shared Task*, 2012.

Eraldo Rezende Fernandes, Cícero Nogueira dos Santos, and Ruy Luiz Milidiú. Latent trees for coreference resolution. *Computational Linguistics*, 2014.

Katja Filippova. Multi-sentence compression: Finding shortest paths in word graphs. In *COLING*, 2010.

Catherine Finegan-Dollak, Reed Coke, Rui Zhang, Xiangyi Ye, and Dragomir Radev. Effects of creativity and cluster tightness on short text clustering performance. In *ACL*, 2016.

Catherine Finegan-Dollak, Jonathan K. Kummerfeld, Li Zhang, Karthik Ramanathan, Sesh Sadasivam, Rui Zhang, and Dragomir Radev. Improving text-to-sql evaluation methodology. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018.

Martin Franz, J Scott McCarley, and Salim Roukos. Ad hoc and multilingual information retrieval at ibm. *NIST special publication SP*, pages 157–168, 1999.

Daniel Fried, Jacob Andreas, and Dan Klein. Unified pragmatic models for generating and following instructions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, 2018.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016.

Daya Guo, Yibo Sun, Duyu Tang, Nan Duan, Jian Yin, Hong Chi, James Cao, Peng Chen, and Ming Zhou. Question generation from sql queries improves neural semantic parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W Bruce Croft. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 55–64. ACM, 2016.

Jiaqi Guo, Zecheng Zhan, Yan Gao, Yan Xiao, Jian-Guang Lou, Ting Liu, and Dongmei Zhang. Towards complex text-to-sql in cross-domain database with intermediate representation. In *ACL*, 2019.

Izzeddin Gur, Semih Yavuz, Yu Su, and Xifeng Yan. Dialsql: Dialogue based structured query generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018.

Kelvin Guu, Panupong Pasupat, Evan Liu, and Percy Liang. From language to programs: Bridging reinforcement learning and maximum marginal likelihood. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017.

Aria Haghighi and Dan Klein. Simple coreference resolution with rich syntactic and semantic features. In *EMNLP*, 2009.

Tatsunori B Hashimoto, Kelvin Guu, Yonatan Oren, and Percy S Liang. A retrieve-and-edit framework for predicting structured outputs. In *Advances in Neural Information Processing Systems*, pages 10052–10062, 2018.

Shirley Anugrah Hayati, Raphael Olivier, Pravalika Avvaru, Pengcheng Yin, Anthony Tomasic, and Graham Neubig. Retrieval-based neural code generation. In *EMNLP*, 2018.

Kenneth Heafield. Kenlm: Faster and smaller language model queries. In *Proceedings of the sixth workshop on statistical machine translation*, pages 187–197. Association for Computational Linguistics, 2011.

Irene Heim. *The semantics of definite and indefinite noun phrases*. PhD thesis, University of Massachusetts, Amherst, 1982.

Charles T Hemphill, John J Godfrey, and George R Doddington. The atis spoken language systems pilot corpus. In *Speech and Natural Language: Proceedings of a Workshop Held at Hidden Valley, Pennsylvania, June 24-27, 1990*, 1990.

James Henderson, Oliver Lemon, and Kallirroi Georgila. Hybrid reinforcement/supervised learning of dialogue policies from fixed data sets. *Computational Linguistics*, 34(4): 487–511, 2008.

Matthew Henderson, Blaise Thomson, and Jason Williams. The second dialog state tracking challenge. In *SIGDIAL*, 2014a.

Matthew Henderson, Blaise Thomson, and Steve Young. Word-based dialog state tracking with recurrent neural networks. In *SIGDIAL*, 2014b.

Geoffrey E Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan R Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

Sepp Hochreiter. The vanishing gradient problem during learning recurrent neural nets and problem solutions. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 6(02):107–116, 1998.

Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Eduard Hovy, Mitchell Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. Ontonotes: the 90% solution. In *NAACL*, 2006.

Wan-Ting Hsu, Chieh-Kai Lin, Ming-Ying Lee, Kerui Min, Jing Tang, and Min Sun. A unified model for extractive and abstractive summarization using inconsistency loss. In *ACL*, 2018.

Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. Convolutional neural network architectures for matching natural language sentences. In *NIPS*, 2014.

Hsin-Yuan Huang, Eunsol Choi, and Wen-tau Yih. Flowqa: Grasping flow in history for conversational machine comprehension. In *ICLR*, 2019.

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. Learning deep structured semantic models for web search using clickthrough data. In *Proceedings of the 22nd ACM international conference on Conference on information & knowledge management*, pages 2333–2338. ACM, 2013.

Po-Sen Huang, Chenglong Wang, Rishabh Singh, Wen-tau Yih, and Xiaodong He. Natural language to structured query generation via meta-learning. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, 2018.

Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. Pacrr: A position-aware neural ir model for relevance matching. *arXiv preprint arXiv:1704.03940*, 2017.

Kai Hui, Andrew Yates, Klaus Berberich, and Gerard de Melo. Co-pacrr: A context-aware neural ir model for ad-hoc retrieval. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 279–287. ACM, 2018.

David A Hull and Gregory Grefenstette. Querying across languages: a dictionary-based approach to multilingual information retrieval. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–57. ACM, 1996.

Wonseok Hwang, Jinyeung Yim, Seunghyun Park, and Minjoon Seo. A comprehensive exploration on wikisql with table-aware word contextualization. *arXiv preprint arXiv:1902.01069*, 2019.

Ozan Irsoy and Claire Cardie. Deep recursive neural networks for compositionality in language. In *Proceedings of NIPS, 2014*, pages 2096–2104, 2014.

Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, Jayant Krishnamurthy, and Luke Zettlemoyer. Learning a neural semantic parser from user feedback. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017.

Srinivasan Iyer, Ioannis Konstas, Alvin Cheung, and Luke Zettlemoyer. Mapping language to code in programmatic context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

Mohit Iyyer, Varun Manjunatha, Jordan Boyd-Graber, and Hal Daumé III. Deep unordered composition rivals syntactic methods for text classification. In *Proceedings of ACL-IJCNLP*, volume 1, pages 1681–1691, 2015.

Mohit Iyyer, Wen-tau Yih, and Ming-Wei Chang. Search-based neural structured learning for sequential question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017.

Zongcheng Ji, Zhengdong Lu, and Hang Li. An information retrieval approach to short text conversation. *arXiv preprint arXiv:1408.6988*, 2014.

Natasa Jovanović, Rieks op den Akker, and Anton Nijholt. Addressee identification in face-to-face meetings. In *EACL*, 2006.

Marcin Junczys-Dowmunt, Roman Grundkiewicz, Tomasz Dwojak, Hieu Hoang, Kenneth Heafield, Tom Neckermann, Frank Seide, Ulrich Germann, Alham Fikri Aji, Nikolay Bogoychev, André F. T. Martins, and Alexandra Birch. Marian: Fast neural machine translation in C++. In *Proceedings of ACL 2018, System Demonstrations*, pages 116–121, Melbourne, Australia, July 2018. Association for Computational Linguistics.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. A convolutional neural network for modelling sentences. *arXiv preprint arXiv:1404.2188*, 2014.

Hans Kamp. A theory of truth and semantic representation. *Formal semantics-the essential readings*, pages 189–222, 1981.

Yaser Keneshloo, Tian Shi, Chandan K Reddy, and Naren Ramakrishnan. Deep reinforcement learning for sequence to sequence models. *arXiv preprint arXiv:1805.09461*, 2018.

Yoon Kim. Convolutional neural networks for sentence classification. In *EMNLP*, 2014a.

Yoon Kim. Convolutional neural networks for sentence classification. In *Proceedings of EMNLP*, pages 1746–1751, Doha, Qatar, October 2014b. URL http://www.aclweb.org/anthology/D14-1181.

Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *International Conference for Learning Representations (ICLR)*, 2015a.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015b.

Shun Kiyono, Sho Takase, Jun Suzuki, Naoaki Okazaki, Kentaro Inui, and Masaaki Nagata. Source-side prediction for neural headline generation. *arXiv preprint arXiv:1712.08302*, 2017.

Bryan Klimt and Yiming Yang. The enron corpus: A new dataset for email classification research. In *ECML*, 2004.

Philipp Koehn. Statistical significance tests for machine translation evaluation. In *EMNLP*, 2004.

Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al.

Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*, pages 177–180. Association for Computational Linguistics, 2007.

Shibamouli Lahiri, Rada Mihalcea, and P-H Lai. Keyword extraction from emails. *Natural Language Engineering*, 23(2), 2017.

Thomas K Landauer and Susan T Dumais. A solution to plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological review*, 104(2):211, 1997.

Quoc V Le and Tomas Mikolov. Distributed representations of sentences and documents. *arXiv preprint arXiv:1405.4053*, 2014.

Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.

Dongjun Lee. Recursive and clause-wise decoding for complex and cross-domain text-to-sql generation. In *CoRR*, 2019.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. Stanford's multi-pass sieve coreference resolution system at the conll-2011 shared task. In *Proceedings of the fifteenth conference on computational natural language learning: Shared task*, pages 28–34, 2011.

Kenton Lee, Luheng He, Mike Lewis, and Luke Zettlemoyer. End-to-end neural coreference resolution. In *EMNLP*, 2017.

David Lewis. A problem about permission. In *Essays in honour of Jaakko Hintikka*, pages 163–175. Springer, 1979.

Jiwei Li, Minh-Thang Luong, and Dan Jurafsky. A hierarchical neural autoencoder for paragraphs and documents. *arXiv preprint arXiv:1506.01057*, 2015.

Jiwei Li, Michel Galley, Chris Brockett, Georgios Spithourakis, Jianfeng Gao, and Bill Dolan. A persona-based neural conversation model. In *ACL*, 2016.

X. Li and D. Roth. Learning question classifiers. In *COLING*, pages 556–562, 2002. URL http://cogcomp.cs.illinois.edu/papers/qc-coling02.pdf.

Chin-Yew Lin. Rouge: A package for automatic evaluation of summaries. *Text Summarization Branches Out*, 2004.

Chu-Cheng Lin, Dongyeop Kang, Michael Gamon, Madian Khabsa, Ahmed Hassan Awadallah, and Patrick Pantel. Actionable email intent modeling with reparametrized rnns. In *AAAI*, 2018.

Robert Litschko, Goran Glavaš, Simone Paolo Ponzetto, and Ivan Vulić. Unsupervised cross-lingual information retrieval using monolingual data only. In *SIGIR*, 2018.

Reginald Long, Panupong Pasupat, and Percy Liang. Simpler context-dependent logical forms via model projections. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2016.

Konstantin Lopyrev. Generating news headlines with recurrent neural networks. *arXiv preprint arXiv:1512.01712*, 2015.

Ryan Lowe, Nissan Pow, Iulian Serban, and Joelle Pineau. The Ubuntu Dialogue Corpus: A large dataset for research in unstructured multi-turn dialogue systems. In *SIGDIAL*, 2015.

Ryan Lowe, Michael Noseworthy, Iulian Vlad Serban, Nicolas Angelard-Gontier, Yoshua Bengio, and Joelle Pineau. Towards an automatic turing test: Learning to evaluate dialogue responses. In *ACL*, 2017.

Vanessa Loza, Shibamouli Lahiri, Rada Mihalcea, and Po-Hsiang Lai. Building a dataset for summarization and keyword extraction from emails. In *LREC*, 2014.

Zhengdong Lu and Hang Li. A deep architecture for matching short texts. In *NIPS*, 2013.

Xiaoqiang Luo. On coreference resolution performance metrics. In *EMNLP*, 2005.

Thang Luong, Hieu Pham, and Christopher D. Manning. Effective approaches to attention-based neural machine translation. In *EMNLP*, 2015.

Mingbo Ma, Liang Huang, Bing Xiang, and Bowen Zhou. Dependency-based convolutional neural networks for sentence embedding. In *Proceedings of ACL-IJCNLP*, volume 2, page 174, 2015.

Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of ACL-HLT*, pages 142–150, Portland, Oregon, USA, June 2011. URL http://www.aclweb.org/anthology/P11-1015.

Sebastian Martschat and Michael Strube. Latent structures for coreference resolution. *Transactions of the Association for Computational Linguistics*, 3:405–418, 2015.

J Scott McCarley. Should we translate the documents or the queries in cross-language information retrieval? In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 208–214. Association for Computational Linguistics, 1999.

Ryan McDonald, Georgios-Ioannis Brokos, and Ion Androutsopoulos. Deep relevance ranking using enhanced document-query interactions. *arXiv preprint arXiv:1809.01682*, 2018.

Hongyuan Mei, Mohit Bansal, and Matthew R. Walter. Coherent dialogue with attention-based language models. In *AAAI*, 2017.

Zhao Meng, Lili Mou, and Zhi Jin. Towards neural speaker modeling in multi-party conversation: The task, dataset, and models. *arXiv preprint arXiv:1708.03152*, 2017.

Grégoire Mesnil, Yann Dauphin, Kaisheng Yao, Yoshua Bengio, Li Deng, Dilek Hakkani-Tur, Xiaodong He, Larry Heck, Gokhan Tur, Dong Yu, et al. Using recurrent neural networks for slot filling in spoken language understanding. *Audio, Speech, and Language Processing, IEEE/ACM Transactions on*, 23(3):530–539, 2015.

Antonio Valerio Miceli Barone, Jindřich Helcl, Rico Sennrich, Barry Haddow, and Alexandra Birch. Deep architectures for neural machine translation. In *Proceedings of the Second Conference on Machine Translation*, Copenhagen, Denmark, 2017. Association for Computational Linguistics.

Rada Mihalcea and Paul Tarau. Textrank: Bringing order into text. In *EMNLP*, 2004.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Proceedings of NIPS*, pages 3111–3119, 2013.

Scott Miller, David Stallard, Robert Bobrow, and Richard Schwartz. A fully statistical approach to natural language interfaces. In *Proceedings of the 34th Annual Meeting of the Association for Computational Linguistics*, 1996.

Bhaskar Mitra, Eric Nalisnick, Nick Craswell, and Rich Caruana. A dual embedding space model for document ranking. *arXiv preprint arXiv:1602.01137*, 2016.

Bhaskar Mitra, Fernando Diaz, and Nick Craswell. Learning to match using local and distributed representations of text for web search. In *Proceedings of the 26th International Conference on World Wide Web*, pages 1291–1299. International World Wide Web Conferences Steering Committee, 2017.

Sunil Mohan, Nicolas Fiorini, Sun Kim, and Zhiyong Lu. Deep learning for biomedical information retrieval: learning textual relevance from click logs. *BioNLP 2017*, pages 222–231, 2017.

Nafise Sadat Moosavi and Michael Strube. Lexical features in coreference resolution: To be used with caution. In *ACL*, 2017.

Smaranda Muresan, Evelyne Tzoukermann, and Judith L Klavans. Combining linguistic and machine learning techniques for email summarization. In *CoNLL*, 2001.

Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *ICML*, 2010.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Don't give me the details, just the summary! Topic-aware convolutional neural networks for extreme summarization. In *EMNLP*, 2018a.

Shashi Narayan, Shay B. Cohen, and Mirella Lapata. Ranking sentences for extractive summarization with reinforcement learning. In *NAACL*, 2018b.

Ani Nenkova and Amit Bagga. Facilitating email thread access by extractive summary generation. *RANLP*, 2003.

Vincent Ng. Supervised noun phrase coreference research: The first fifteen years. In *ACL*, 2010.

Vincent Ng and Claire Cardie. Improving machine learning approaches to coreference resolution. In *ACL*, 2002.

Jian-Yun Nie. Cross-language information retrieval. *Synthesis Lectures on Human Language Technologies*, 3(1):1–125, 2010.

NIST. The Official Original Derivation of AQWV, 2017. URL https://www.nist.gov/sites/default/files/documents/2017/10/26/aqwv_derivation.pdf.

Douglas W Oard. A comparative study of query and document translation for cross-language information retrieval. In *Conference of the Association for Machine Translation in the Americas*, pages 472–483. Springer, 1998.

Douglas W Oard and Paul Hackett. Document translation for cross-language text retrieval at the university of maryland. In *TREC*, pages 687–696. Citeseer, 1997.

Douglas W. Oard, Petra Galuščáková, Kathleen McKeown, Marine Carpuat, Mohamed Elbadrashiny, Ramy Eskander, Kenneth Heafield, Efsun Kayi, Chris Kedzie, Smaranda Muresan, Suraj Nair, Xing Niu, Dragomir Radev, Anton Ragni, Han-Chin Shing, Yan Virin, Weijia Xu, Rui Zhang, Elena Zotkina, Joseph Barrow, and Mark Gales. Surprise languages: Rapid-response cross-language IR. In *EVIA*, 2019.

Rieks op den Akker and David Traum. A comparison of addressee detection methods for multiparty conversations. *Workshop on the Semantics and Pragmatics of Dialogue*, 2009.

Hiroki Ouchi and Yuta Tsuboi. Addressee and response selection for multi-party conversation. In *EMNLP*, 2016.

Bo Pang and Lillian Lee. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of ACL*, page 271. Association for Computational Linguistics, 2004.

Bo Pang and Lillian Lee. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of ACL*, pages 115–124. Association for Computational Linguistics, 2005.

Liang Pang, Yanyan Lan, Jiafeng Guo, Jun Xu, and Xueqi Cheng. A study of matchpyramid models on ad-hoc retrieval. *arXiv preprint arXiv:1606.04648*, 2016.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. On the difficulty of training recurrent neural networks. In *ICML*, 2013.

Panupong Pasupat and Percy Liang. Compositional semantic parsing on semi-structured tables. In *Proceedings of the 53rd Annual Meeting of the Association for Computational*

Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), 2015.

Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. In *NIPS 2017 Workshop Autodiff*, 2017.

Romain Paulus, Caiming Xiong, and Richard Socher. A deep reinforced model for abstractive summarization. In *ICLR*, 2018.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. Glove: Global vectors for word representation. *Proceedings of EMNLP*, 12:1532–1543, 2014.

Ari Pirkola. The effects of query structure and dictionary setups in dictionary-based cross-language information retrieval. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 55–63. ACM, 1998.

Paul Portner. Imperatives and modals. *Natural Language Semantics*, 15(4):351–383, 2007.

Vinodkumar Prabhakaran and Owen Rambow. Predicting power relations between participants in written dialog from a single thread. In *ACL*, 2014.

Vinodkumar Prabhakaran, Emily E Reid, and Owen Rambow. Gender and power: How gender and gender environment affect manifestations of power. In *EMNLP*, 2014.

Sameer Pradhan, Alessandro Moschitti, Nianwen Xue, Olga Uryupina, and Yuchen Zhang. Conll-2012 shared task: Modeling multilingual unrestricted coreference in ontonotes. In *Joint Conference on EMNLP and CoNLL-Shared Task*, 2012.

Owen Rambow, Lokesh Shrestha, John Chen, and Chirsty Lauridsen. Summarizing email threads. In *NAACL*, 2004.

Marc'Aurelio Ranzato, Sumit Chopra, Michael Auli, and Wojciech Zaremba. Sequence level training with recurrent neural networks. In *ICLR*, 2016.

Alan Ritter, Colin Cherry, and William B Dolan. Data-driven response generation in social media. In *EMNLP*, 2011.

Alexander M. Rush, Sumit Chopra, and Jason Weston. A neural attention model for abstractive sentence summarization. In *EMNLP*, 2015.

Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. Grammatical error correction with neural reinforcement learning. In *IJCNLP*, 2017.

Shota Sasaki, Shuo Sun, Shigehiko Schamoni, Kevin Duh, and Kentaro Inui. Cross-lingual learning-to-rank with shared representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, volume 2, pages 458–463, 2018.

Simon Scerri, Gerhard Gossen, Brian Davis, and Siegfried Handschuh. Classifying action items for semantic email. In *LREC*, 2010.

Shigehiko Schamoni, Felix Hieber, Artem Sokolov, and Stefan Riezler. Learning translational and knowledge-based similarities from relevance rankings for cross-language retrieval. In *Proceedings of the 52 Annual Meeting of the Association for Computational Linguistics (ACL)*, 2014.

Abigail See, Peter J. Liu, and Christopher D. Manning. Get to the point: Summarization with pointer-generator networks. In *ACL*, 2017.

Iulian V Serban, Alessandro Sordoni, Yoshua Bengio, Aaron Courville, and Joelle Pineau. Building end-to-end dialogue systems using generative hierarchical neural network models. In *AAAI*, 2016.

Aliaksei Severyn and Alessandro Moschitti. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 373–382. ACM, 2015.

Guokan Shang, Wensi Ding, Zekun Zhang, Antoine Tixier, Polykarpos Meladianos, Michalis Vazirgiannis, and Jean-Pierre Lorré. Unsupervised abstractive meeting summarization with multi-sentence compression and budgeted submodular maximization. In *ACL*, 2018.

Lifeng Shang, Zhengdong Lu, and Hang Li. Neural responding machine for short-text conversation. In *ACL*, 2015.

Shi-Qi Shen, Yan-Kai Lin, Cun-Chao Tu, Yu Zhao, Zhi-Yuan Liu, Mao-Song Sun, et al. Recent advances on neural headline generation. *Journal of Computer Science and Technology*, 2017.

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*, pages 373–374. ACM, 2014.

Tianze Shi, Kedar Tatwawadi, Kaushik Chakrabarti, Yi Mao, Oleksandr Polozov, and Weizhu Chen. Incsql: Training incremental text-to-sql parsers with non-deterministic oracles. *arXiv preprint arXiv:1809.05054*, 2018.

Hiroki Shimanaka, Tomoyuki Kajiwara, and Mamoru Komachi. Metric for automatic machine translation evaluation based on universal sentence representations. In *NAACL: Student Research Workshop*, 2018.

Joao Silva, Luísa Coheur, Ana Cristina Mendes, and Andreas Wichert. From symbolic to sub-symbolic information in question classification. *Artificial Intelligence Review*, 35 (2):137–154, 2011.

Satinder P Singh, Michael J Kearns, Diane J Litman, and Marilyn A Walker. Reinforcement learning for spoken dialogue systems. In *NIPS*, 1999.

Richard Socher, Brody Huval, Christopher D Manning, and Andrew Y Ng. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of EMNLP-CoNLL*, pages 1201–1211. Association for Computational Linguistics, 2012.

Richard Socher, Alex Perelygin, Jean Y Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*, volume 1631, page 1642. Citeseer, 2013.

Artem Sokolov, Laura Jehl, Felix Hieber, and Stefan Riezler. Boosting cross-language retrieval by learning bilingual phrase associations from relevance rankings. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1688–1699, 2013.

Wee Meng Soon, Hwee Tou Ng, and Daniel Chung Yong Lim. A machine learning approach to coreference resolution of noun phrases. *Computational linguistics*, 27(4):521–544, 2001.

Robert C Stalnaker. Assertion. In *Pragmatics*, pages 315–332. Brill, 1978.

Yu Su and Xifeng Yan. Cross-domain semantic parsing via paraphrasing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, 2017.

Alane Suhr and Yoav Artzi. Situated mapping of sequential instructions to actions with single-step reward observation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018.

Alane Suhr, Srinivasan Iyer, and Yoav Artzi. Learning to map context-dependent sentences to executable formal queries. In *Proceedings of the 2018 Conference of the North*

American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers), 2018.

Yibo Sun, Duyu Tang, Nan Duan, Jianshu Ji, Guihong Cao, Xiaocheng Feng, Bing Qin, Ting Liu, and Ming Zhou. Semantic parsing with syntax- and table-aware sql generation. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2018.

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. Sequence to sequence learning with neural networks. In *NIPS*, 2014.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075*, 2015.

Jiwei Tan, Xiaojun Wan, and Jianguo Xiao. From neural sentence summarization to headline generation: a coarse-to-fine approach. In *IJCAI*, 2017.

Ottokar Tilk and Tanel Alumäe. Low-resource neural headline generation. *arXiv preprint arXiv:1707.09769*, 2017.

Ferhan Ture, Jimmy Lin, and Douglas Oard. Combining statistical translation techniques for cross-language information retrieval. *Proceedings of COLING 2012*, pages 2685–2702, 2012.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. Word representations: a simple and general method for semi-supervised learning. In *ACL*, 2010.

Peter D Turney. Learning algorithms for keyphrase extraction. *Information retrieval*, 2(4): 303–336, 2000.

Jan Ulrich, Giuseppe Carenini, Gabriel Murray, and Raymond T Ng. Regression-based summarization of email conversations. In *ICWSM*, 2009.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. A model-theoretic coreference scoring scheme. In *Proceedings of the 6th conference on Message understanding*, 1995.

Oriol Vinyals and Quoc Le. A neural conversational model. *ICML Deep Learning Workshop*, 2015.

Oriol Vinyals, Alexander Toshev, Samy Bengio, and Dumitru Erhan. Show and tell: A neural image caption generator. *arXiv preprint arXiv:1411.4555*, 2014.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. Pointer networks. In *NIPS*, 2015.

Andreas Vlachos and Stephen Clark. A new corpus and imitation learning framework for context-dependent semantic parsing. *Transactions of the Association for Computational Linguistics*, 2014.

Ivan Vulić and Marie-Francine Moens. Monolingual and cross-lingual information retrieval models based on (bilingual) word embeddings. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 363–372. ACM, 2015.

Alexander Waibel, Toshiyuki Hanazawa, Geoffrey Hinton, Kiyohiro Shikano, and Kevin J Lang. Phoneme recognition using time-delay neural networks. *Acoustics, Speech and Signal Processing, IEEE Transactions on*, 37(3):328–339, 1989.

Stephen Wan and Kathy McKeown. Generating overview summaries of ongoing email thread discussions. In *COLING*, 2004.

Mingxuan Wang, Zhengdong Lu, Hang Li, and Qun Liu. Syntax-based deep matching of short texts. In *IJCAI*, 2015a.

Sida Wang and Christopher D Manning. Baselines and bigrams: Simple, good sentiment and topic classification. In *Proceedings of ACL: Short Papers-Volume 2*, pages 90–94. Association for Computational Linguistics, 2012.

Yushi Wang, Jonathan Berant, and Percy Liang. Building a semantic parser overnight. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 2015b.

Tsung-Hsien Wen, Milica Gasic, Nikola Mrksic, Pei-Hao Su, David Vandyke, and Steve Young. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. *arXiv preprint arXiv:1508.01745*, 2015a.

Tsung-Hsien Wen, Milica Gašić, Nikola Mrkšić, Pei-Hao Su, David Vandyke, and Steve Young. Semantically conditioned lstm-based natural language generation for spoken dialogue systems. In *EMNLP*, 2015b.

Tsung-Hsien Wen, David Vandyke, Nikola Mrksic, Milica Gasic, Lina M Rojas-Barahona, Pei-Hao Su, Stefan Ultes, and Steve Young. A network-based end-to-end trainable task-oriented dialogue system. *arXiv preprint arXiv:1604.04562*, 2016.

Jason Williams, Antoine Raux, and Matthew Henderson. The dialog state tracking challenge series: A review. *Dialogue & Discourse*, 7(3):4–33, 2016.

Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning*, 8(3-4):229–256, 1992.

Sam Wiseman, Alexander M Rush, and Stuart M Shieber. Learning global features for coreference resolution. In *NAACL*, 2016.

Sam Joshua Wiseman, Alexander Matthew Rush, Stuart Merrill Shieber, and Jason Weston. Learning anaphoricity and antecedent ranking features for coreference resolution. In *ACL*, 2015.

Chao Xing, Dong Wang, Chao Liu, and Yiye Lin. Normalized word embedding and orthogonal transform for bilingual word translation. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1006–1011, 2015.

Chenyan Xiong, Zhuyun Dai, Jamie Callan, Zhiyuan Liu, and Russell Power. End-to-end neural ad-hoc ranking with kernel pooling. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 55–64. ACM, 2017.

Xiaojun Xu, Chang Liu, and Dawn Song. Sqlnet: Generating structured queries from natural language without reinforcement learning. *arXiv preprint arXiv:1711.04436*, 2017.

Seth Yalcin. Introductory notes on dynamic semantics. *Routledge Companion to the Philosophy of Language*, pages 253–79, 2013.

Kaisheng Yao, Baolin Peng, Yu Zhang, Dong Yu, Geoffrey Zweig, and Yangyang Shi. Spoken language understanding using long short-term memory neural networks. In *Spoken Language Technology Workshop (SLT), 2014 IEEE*, pages 189–194. IEEE, 2014.

Michihiro Yasunaga, Rui Zhang, Kshitijh Meelu, Ayush Pareek, Krishnan Srinivasan, and Dragomir Radev. Graph-based neural multi-document summarization. In *CoNLL*, 2017.

Michihiro Yasunaga, Jungo Kasai, Rui Zhang, Alexander R Fabbri, Irene Li, Dan Friedman, and Dragomir R Radev. Scisummnet: A large annotated corpus and content-impact models for scientific paper summarization with citation networks. In *AAAI*, 2019.

Semih Yavuz, Izzeddin Gur, Yu Su, and Xifeng Yan. What it takes to achieve 100% condition accuracy on wikisql. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018.

Pengcheng Yin and Graham Neubig. A syntactic neural model for general-purpose code generation. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017.

Pengcheng Yin, Graham Neubig, Miltiadis Allamanis, Marc Brockschmidt, and Alexander L Gaunt. Learning to represent edits. In *ICLR*, 2019.

Wenpeng Yin and Hinrich Schütze. Multichannel variable-size convolution for sentence classification. In *Proceedings of CoNLL*, pages 204–214, Beijing, China, July 2015. URL http://www.aclweb.org/anthology/K15-1021.

Steve Young, Milica Gasic, Blaise Thomson, and John D Williams. Pomdp-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5):1160–1179, 2013.

Tao Yu, Zifan Li, Zilin Zhang, Rui Zhang, and Dragomir Radev. Typesql: Knowledge-based type-aware neural text-to-sql generation. In *Proceedings of NAACL*. Association for Computational Linguistics, 2018a.

Tao Yu, Michihiro Yasunaga, Kai Yang, Rui Zhang, Dongxu Wang, Zifan Li, and Dragomir Radev. Syntaxsqlnet: Syntax tree networks for complex and cross-domain text-to-sql task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018b.

Tao Yu, Rui Zhang, Kai Yang, Michihiro Yasunaga, Dongxu Wang, Zifan Li, James Ma, Irene Li, Qingning Yao, Shanelle Roman, Zilin Zhang, and Dragomir Radev. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 2018c.

Tao Yu, Rui Zhang, He Yang Er, Suyi Li, Eric Xue, Bo Pang, Xi Victoria Lin, Yi Chern Tan, Tianze Shi, Zihan Li, Youxuan Jiang, Michihiro Yasunaga, Sungrok Shim, Tao Chen, Alexander Fabbri, Zifan Li, Luyao Chen, Yuwen Zhang, Shreya Dixit, Vincent Zhang, Caiming Xiong, Richard Socher, Walter Lasecki, and Dragomir Radev. Cosql: A conversational text-to-sql challenge towards cross-domain natural language interfaces to databases. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing*. Association for Computational Linguistics, 2019a.

Tao Yu, Rui Zhang, Michihiro Yasunaga, Yi Chern Tan, Xi Victoria Lin, Suyi Li, Heyang Er, Irene Li, Bo Pang, Tao Chen, Emily Ji, Shreya Dixit, David Proctor, Sungrok Shim, Jonathan Kraft, Vincent Zhang, Caiming Xiong, Richard Socher, and Dragomir Radev. Sparc: Cross-domain semantic parsing in context. In *CoRR*, 2019b.

David M Zajic, Bonnie J Dorr, and Jimmy Lin. Single-document and multi-document summarization techniques for email threads using sentence compression. *Information Processing & Management*, 44(4), 2008.

Matthew D Zeiler. Adadelta: An adaptive learning rate method. *arXiv preprint arXiv:1212.5701*, 2012.

John M. Zelle and Raymond J. Mooney. Learning to parse database queries using inductive logic programming. In *AAAI/IAAI*, pages 1050–1055, Portland, OR, August 1996. AAAI Press/MIT Press.

Luke S. Zettlemoyer and Michael Collins. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. *UAI*, 2005.

Luke S Zettlemoyer and Michael Collins. Learning context-dependent mappings from sentences to logical form. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language*

Processing of the AFNLP: Volume 2-Volume 2, pages 976–984. Association for Computational Linguistics, 2009.

Chengxiang Zhai and John Lafferty. A study of smoothing methods for language models applied to information retrieval. *ACM Transactions on Information Systems (TOIS)*, 22 (2):179–214, 2004.

Rui Zhang and Joel Tetreault. This email could save your life: Introducing the task of email subject line generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 446–456, Florence, Italy, July 2019. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P19-1043.

Rui Zhang, Honglak Lee, and Dragomir R. Radev. Dependency sensitive convolutional neural networks for modeling sentences and documents. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1512–1521, San Diego, California, June 2016. Association for Computational Linguistics. URL http://www.aclweb.org/anthology/N16-1177.

Rui Zhang, Honglak Lee, Lazaros Polymenakos, and Dragomir R. Radev. Addressee and response selection in multi-party conversations with speaker interaction rnns. In *Proceedings of AAAI-2018*, 2018a.

Rui Zhang, Cicero Nogueira dos Santos, Michihiro Yasunaga, Bing Xiang, and Dragomir Radev. Neural coreference resolution with deep biaffine attention by joint mention detection and mention clustering. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 102–107. Association for Computational Linguistics, 2018b. URL http://aclweb.org/anthology/P18-2017.

Rui Zhang, Caitlin Westerfield, Sungrok Shim, Garrett Bingham, Alexander Fabbri, William Hu, Neha Verma, and Dragomir Radev. Improving low-resource cross-lingual document retrieval by reranking with deep bilingual representations. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3173–3179, Florence, Italy, July 2019a. Association for Computational Linguistics. URL https://www.aclweb.org/anthology/P19-1306.

Rui Zhang, Tao Yu, Heyang Er, Sungrok Shim, Eric Xue, Xi Victoria Lin, Tianze Shi, Caiming Xiong, Richard Socher, and Dragomir Radev. Editing-based SQL query generation for cross-domain context-dependent questions. 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)*, pages 5337–5348, Hong Kong, China, November 2019b. Association for Computational Linguistics. doi: 10.18653/v1/D19-1537. URL https://www.aclweb.org/anthology/D19-1537.

Xingxing Zhang and Mirella Lapata. Sentence simplification with deep reinforcement learning. In *EMNLP*, 2017.

Victor Zhong, Caiming Xiong, and Richard Socher. Seq2SQL: Generating structured queries from natural language using reinforcement learning. *CoRR*, abs/1709.00103, 2017.

Xiangyang Zhou, Daxiang Dong, Hua Wu, Shiqi Zhao, Dianhai Yu, Hao Tian, Xuan Liu, and Rui Yan. Multi-view response selection for human-computer conversation. In *EMNLP*, 2016.