University of California
Santa Barbara

DeepAssist: Deep Knowledge Grounding for Factual and Conversational Natural Language Interfaces

A dissertation submitted in partial satisfaction
of the requirements for the degree

Doctor of Philosophy
in
Computer Science

by

Semih Yavuz

Committee in charge:

Professor Xifeng Yan, Chair
Professor Amr El-Abbadi
Professor William Yang Wang

September 2019

The Dissertation of Semih Yavuz is approved.

Professor Amr El-Abbadi

Professor William Yang Wang

Professor Xifeng Yan, Committee Chair

September 2019

DeepAssist: Deep Knowledge Grounding for

Factual and Conversational

Natural Language Interfaces

by

Semih Yavuz

To my parents and Aybike, for their unconditional love and endless support.

Acknowledgements

I would like to thank my advisor, Xifeng Yan, for his support and expertise. It was his foresight that got me into the area of deep learning and natural language processing, which I enjoy and ambitiously conduct research on. I am sincerely grateful for his generous guidance and support over the five years of a quite intensive PhD life. He taught and always encouraged me to be curious, genuine, and resilient towards research. I will try to adhere to these core values for the rest of my professional and personal life.

I am also quite grateful to my committee members, Amr El-Abbadi and William Yang Wang for their invaluable feedbacks throughout my graduate studies.

I would like to thank my mentors; Dilek Hakkani-Tur, Arvind Neelekantan, Chung-Cheng Chiu, Patrick Nguyen, Yonghui Wu, Abhinav Rastogi, Olya Gurevich, Abdullah Akce, and all my collaborators; Ben Goodrich, Daniel Duckworth, Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Colin Raffel, Chinnadhurai Sankar, Guan-lin Chao, Mudhakar Srivatsa, Jindong Chen, Ian Lane, Izzeddin Gur, Yu Su, Honglei Liu, Huan Sun, Keqian Li, Hanwen Zha, Shiyang Li.

I would like to express my deepest gratitude to my wife, Aybike, for her endless support and unconditional love. She has always been there to share both the happy and stressful moments of the PhD life. I also thank my parents and sister for their constant love and support.

The research in this dissertation is funded in part by the Army Research Laboratory under cooperative agreements W911NF09-2-0053, NSF IIS 1528175, and NSF CCF 1548848, and Google AI.

Curriculum Vitæ

Semih Yavuz

Education

2014 - 2019	Ph.D. in Computer Science, University of California, Santa Barbara.
2008 - 2013	B.S. in Mathematics, Bilkent University.
2011 - 2012	Education Abroad Program, University of California, Los Angeles.

Experience

11/2018 - 6/2019	Student Researcher, Google Brain, Mountain View.
6/2018 - 9/2018	Research Intern, Google AI, Mountain View.
6/2017 - 9/2017	Research Intern, Google Brain, Mountain View.
6/2016 - 9/2016	Software Engineering Intern, Apple Siri, San Francisco.
6/2015 - 9/2015	Software Engineering Intern, Google Search, Mountain View.
9/2015 - 7/2019	Research Assistant, University of California, Santa Barbara.
9/2014 - 6/2015	Teaching Assistant, University of California, Santa Barbara.
6/2012 - 8/2012	Summer Undergraduate Research Fellow (SURF), California Institute of Technology.

Selected Publications

SIGDIAL 2019	Semih Yavuz, Abhinav Rastogi, Guan-Lin Chao and Dilek Hakkani-Tur. DeepCopy: Grounded Response Generation with Hierarchical Pointer Networks.
SIGDIAL 2019	Guan-Lin Chao, Abhinav Rastogi, Semih Yavuz, Dilek Hakkani-Tur, Jindong Chen and Ian Lane. Learning Question-Guided Video Representation for Multi-Turn Video Question Answering.
ACL 2019	Naveen Arivazhagan, Colin Cherry, Wolfgang Macherey, Chung-Cheng Chiu, Semih Yavuz, Ruoming Pang, Wei Li, Colin Raffel. Monotonic Infinite Lookback Attention for Simultaneous Machine Translation.
EMNLP 2019	Bill Byrne, Karthik Krishnamoorthi, Chinnadhurai Sankar, Arvind Neelakantan, Ben Goodrich, Daniel Duckworth, Semih Yavuz, Amit Dubey, Kyu-Young Kim and Andy Cedilnik. TaskMaster Dialog Corpus: Toward a Realistic and Diverse Dataset.
ICDM 2019	Keqian Li, Shiyang Li, Semih Yavuz, Hanwen Zha, Yu Su, and Xifeng Yan. HierCon: Hierarchical Organization of Technical Documents based on Concepts.

EMNLP 2018	Semih Yavuz, Chung-Cheng Chiu, Patrick Nguyen, Yonghui Wu. CaLCS: Continuously Approximating Longest Common Subsequence for Sequence Level Optimization.
EMNLP 2018	Semih Yavuz, Izzeddin Gur, Yu Su, Xifeng Yan. What It Takes to Achieve 100% Condition Accuracy on WikiSQL.
ACL 2018	Izzeddin Gur, Semih Yavuz, Yu Su, Xifeng Yan. DialSQL: Dialogue Based Structured Query Generation.
NAACL 2018	Yu Su, Honglei, Liu, Semih Yavuz, Izzeddin Gur, Huan Sun, Xifeng Yan. Global Relation Embedding for Relation Extraction.
EMNLP 2017	Semih Yavuz, Izzeddin Gur, Yu Su, Xifeng Yan. Recovering Question Answering Errors via Query Revision.
EMNLP 2016	Semih Yavuz, Izzeddin Gur, Yu Su, Mudhakar Srivatsa, Xifeng Yan. Improving Semantic Parsing via Answer Type Inference.
ITW 2013	Batuhan Karagoz, Semih Yavuz, Tracey Ho, Michelle Effros. Capacity Region of multi-resolution streaming in peer-to-peer networks.

Awards and Honors

2019	SIGDIAL 2019 Travel Award, ISCA
2018	Best Paper Award, 2nd Conversational AI Workshop at NeurIPS
2013	CSE Department Fellowship, University of California San Diego
2013	Outstanding Graduate Award, awarded to the top graduating student of Faculty of Science
2012	Summer Undergraduate Research Fellowship (SURF), California Institute of Technology
2008 - 2013	TUBITAK Fellow, The Scientific and Technological Research Council of Turkey
2008 - 2013	Full Scholarship, awarded by Bilkent University for the entire period of undergraduate study
2008	Bronze Medalist, International Mathematical Olympiad (IMO)
2008	Silver Medalist, Balkan Mathematical Olympiad (BMO)
2008	Gold Medalist, International Silk Road Mathematical Olympiad
2007	Bronze Medalist, International Mathematical Olympiad (IMO)
2007	Bronze Medalist, Balkan Mathematical Olympiad (BMO)
2006	Bronze Medalist, National Mathematical Olympiad
2005	Gold Medalist, National Akdeniz University Mathematical Olympiad
2004	Silver Medalist, National Junior Mathematical Olympiad

Abstract

DeepAssist: Deep Knowledge Grounding for
Factual and Conversational
Natural Language Interfaces

by

Semih Yavuz

Enabling humans to use natural language to interact with computers towards achieving certain goals such accessing factual information, finding restaurants, holding engaging conversations with an AI agent has been one of the central goals of Artificial Intelligence. Many people use natural language interfaces (NLIs) such as Siri, Google Assistant, and Alexa in their daily life. Furthermore, there are several more equally promising, but less explored domains such as health, law, customer service, etc. Hence, developing more reliable, capable, and extendible NLIs has the potential to lead to the next generation human computer interaction technologies. Although several promising results have been achieved in both academia and industry, there are still several challenges to tackle towards realizing such NLIs in full fledge. This thesis tackles the problem of building reliable NLIs by addressing the central challenges confronted with.

The contributions of this thesis are presented in three main parts. The first part is covered by the first three chapters and focuses on factual (single-turn) NLIs that aim to generate a concise answer for factual user queries such as "Who is the president of Canada?". We discuss how our proposed approaches for answer type inference and query revision of factual queries over a large knowledge base can help improve the performance of the state-of-the-art NLIs for factoid question answering task. In the third chapter, we present an in-depth analysis towards better understanding the kinds of language

understanding capabilities required to solve current benchmarks.

In the second part, we investigate conversational NLIs that can generate responses to user queries in a multi-turn fashion. With the goal of making these responses more informative and engaging for users, the main contribution of this study is to introduce principled neural architecture that can generate responses grounded on a relevant external knowledge by hierarchical attention and copy mechanisms.

Learning to generate coherent and engaging natural language from data is one of the most crucial capabilities for the next generation NLIs that can speak with users. In the last part of this thesis, we focus on a more foundational line of research where we propose and discuss a novel training objective for conditional language generation models.

Contents

Chapter 1

Introduction

Over the past decades, the amount of digital data has been exponentially growing. Large amount of worlds information is now digitally available and stored in various kinds of digital forms including structured databases, semi-structured web tables, and unstructured text articles. As a simple example, while the number of English Wikipedia articles was around 20,000 in 2002, this number is now close to 6,000,000 based on the last publicly available report by Wikipedia. Although the growth of digital data is a quite beneficial step for the development of modern society, massive scale and heterogeneity of the data prevent non-expert users to effectively access the information for problem solving and decision making. This becomes an even more serious problem and puts a huge barrier between non-expert users and data especially in the scenarios where even accessing the information requires learning specific query languages and programming, hence an extensive and costly training. As a step towards removing this huge barrier between non-expert users and data, this thesis investigates natural language interfaces (NLIs), a research direction that aims to enhance user experience by enabling humans to use natural language to interact with computers and data.

Enabling humans to use natural language to interact with computers towards achiev-

ing certain goals such accessing factual information, booking flights or finding restaurants, holding engaging conversations with an AI agent has been one of the most central and elusive goals of Artificial Intelligence research. Many people use natural language interfaces (NLIs) such as Siri, Google Assistant, and Alexa in their daily life. Furthermore, there are many more equally promising and impactful, but relatively less explored domains such as health, real estate, customer service, etc. Hence, developing more reliable, capable, and extendible natural language interfaces has the potential to lead to the next generation human computer interaction technologies through applications like question answering and dialogue systems. Although several promising results have been achieved in both academia and industry, there are still several challenges and open research problems including population, representation, and better incorporation of knowledge towards realizing such natural language interfaces in full fledge. This thesis tackles the problem of knowledge grounding for building reliable natural language interfaces by addressing the central challenges induced.

In this thesis, we study *deep knowledge grounding for natural language interfaces*, a class of natural language grounding models built on top of deep neural networks. These neural models have proven to perform better than traditional rule-based or handcrafted feature-based models on various modern language understanding benchmarks. The contributions of this thesis are presented in three main parts. As introduced in more detail below, the first two parts are more application-oriented while the last part is more foundational line of study.

1.1 Factual Natural Language Interfaces

In Chapters 2-4, we focus on factual (single-turn) NLIs that aim to generate a concise answer for factual user queries such as "Who is the president of United States?". Our

common goal throughout these chapters is to better understand the challenges posed by this problem, identify key bottlenecks of the existing approaches, and propose viable solutions that bridge these gaps leading to more accurate and reliable models. In particular, in the second and third chapters, we discuss how our proposed approaches for answer type inference and query revision of factual queries over a large knowledge base (KB) can help improve the performance of the best available NLIs for factoid question answering task. In the fourth chapter, on the other hand, we begin our study with a comprehensive analysis over a recent benchmark to provide a more holistic overview of the kinds of language understanding capabilities required for solving natural language to structured query translation. Subsequently, we propose several alternative approaches to overcome the identified challenges along with a discussion on the extent to which some of these bottlenecks for successful translation are addressed.

In Chapter 2, we show the possibility of inferring the answer type before solving a factoid question and leveraging the type information to improve semantic parsing. By replacing the topic entity in a question with its type, we are able to generate an abstract form of the question, whose answer corresponds to the answer type of the original question. A bidirectional LSTM model is built to train over the abstract form of questions and infer their answer types. Using the predicted type information to rerank the logical forms returned by AgendaIL [1], one of the leading semantic parsers, we are able to improve the F1-score from 49.7% to 52.6% on the WebQuestions benchmark.

In Chapter 3, we start by observing that existing factoid QA systems often lack a post-inspection component that can help models recover from their own mistakes. Hence, in this study [2], we propose to cross-check the corresponding KB relations behind the predicted answers and identify potential inconsistencies. Instead of developing a new model that accepts evidences collected from these relations, we choose to plug them back to the original questions directly and check if the revised question makes sense or

not. A bidirectional LSTM is applied to encode revised questions. We develop a scoring mechanism over the revised question encodings to refine the predictions of a base QA system. This approach can improve the F_1 score of STAGG [3], one of the leading QA systems, from 52.5% to 53.9% on WEBQUESTIONS benchmark.

In Chapter 4, we present an in-depth analysis [4] towards better understanding the depth and kinds of language understanding capabilities required to solve current benchmark NLI tasks. More precisely, we investigate WikiSQL benchmark, a newly released dataset for studying the natural language sequence to SQL translation problem. The SQL queries in WikiSQL are simple: Each involves one relation and does not have any join operation. Despite of its simplicity, none of the publicly reported structured query generation models can achieve a satisfactory enough performance to be reliably put in a practical use. In this study, we ask two questions, "Why is the accuracy still low for such simple queries?" and "What does it take to achieve 100% accuracy on WikiSQL?" To limit the scope of our study, we focus on the `WHERE` clause in SQL. The answers will help us gain insights about the directions we should explore in order to further improve the translation accuracy. We will then investigate alternative solutions to realize the potential ceiling performance on WikiSQL. Our proposed solution can reach up to 88.6% condition accuracy on the WikiSQL dataset.

1.2 Conversational Natural Language Interfaces

A large portion of tasks that humans may desire to complete through an interaction with an agent requires the capability of holding coherent and engaging enough conversation with users. Some more concrete examples of tasks in this setting include booking flights, finding restaurants, having a casual or intellectual conversation with an AI agent. To this end, in the second part of this thesis, we turn our attention to conversational

natural language interfaces that can generate responses to user queries in a multi-turn fashion. More specifically, we focus on addressing commonly observed key problems of model generated responses lacking factual information, hence resulting in short and dull turns. With the goal of making these responses more informative and engaging for users, the main contribution of this study is to introduce principled neural architecture that can generate responses grounded on a relevant external knowledge by hierarchical attention and copy mechanisms.

Recent advances in neural sequence-to-sequence models have led to promising results for several language generation-based tasks, including dialogue response generation, summarization, and machine translation. However, these models are known to have several problems, especially in the context of chit-chat based dialogue systems: they tend to generate short and dull responses that are often too generic. Furthermore, these models do not ground conversational responses on knowledge and facts, resulting in turns that are not accurate, informative and engaging for the users. In Chapter 5, we propose and experiment with a series of response generation models that aim to serve in the general scenario where in addition to the dialogue context, relevant unstructured external knowledge in the form of text is also assumed to be available for models to harness. Our proposed approach extends pointer-generator networks [5] by allowing the decoder to hierarchically attend and copy from external knowledge in addition to the dialogue context. We empirically show the effectiveness of the proposed model compared to several baselines including [6, 7] through both automatic evaluation metrics and human evaluation on ConvAi2 dataset.

1.3 Novel Training Objective for Improved Neural Conditional Language Generation

Learning to generate coherent and engaging natural language from data is one of the most crucial capabilities for the next generation NLIs that can speak with users. This capability becomes particularly important for the conversational settings, where the user experience relies completely on their interaction with the computer using the natural language. Hence, the quality of model generated response in natural language plays a significant role in determining whether these models can be put in a real use. Contributions to advance the field of conditional language generation have also great impacts on several other downstream tasks such machine translation, text summarization, and image captioning, which can eventually become a part of a more advanced NLI system equipped with such capabilities. Although neural sequence-to-sequence models are shown to be very promising approaches for conditional text generation, they are reported to still suffer from the discrepancy between their training and inference time objectives that leads to particular drawbacks. Toward addressing these shortcomings, in Chapter 6 of this thesis, we focus on a more foundational line of research where we propose and discuss a novel training objective for neural conditional language generation models.

Maximum-likelihood estimation (MLE) is one of the most widely used approaches for training structured prediction models for text-generation based natural language processing applications. However, besides *exposure bias*, models trained with MLE suffer from *wrong objective* problem where they are trained to maximize the word-level correct next step prediction, but are evaluated with respect to sequence-level discrete metrics such as ROUGE and BLEU. Several variants of policy-gradient methods address some of these problems by optimizing for final discrete evaluation metrics and showing improvements

over MLE training for downstream tasks like text summarization and machine translation. However, policy-gradient methods suffers from high sample variance, making the training process very difficult and unstable. In this study, we present an alternative direction towards mitigating this problem by introducing a new objective (CaLcs) based on a differentiable surrogate of longest common subsequence (LCS) measure that captures sequence-level structure similarity. Experimental results on abstractive summarization and machine translation validate the effectiveness of the proposed approach.

Chapter 2

Improving Semantic Parsing via Answer Type Inference

2.1 Introduction

Large scale knowledge bases (KB) like Freebase [8], DBpedia [9], and YAGO [10] that store the world's factual information in a structured fashion have become substantial resources for people to solve questions. KB-based factoid question answering (KB-QA) that attempts to find exact answers to natural language questions has gained much attention recently. KB-QA is a challenging task due to the representation variety between natural language and structural knowledge in KBs.

As one of the promising KB-QA techniques, semantic parsing maps a natural language question into its semantic representation (e.g., logical forms). It uses a logical language with predicates closely related to KB schema, and constructs a dictionary that maps relations to KB predicates. The problem then reduces to generating candidate logical forms, ranking them, and selecting one to derive the final answer.

In this work, we propose an answer type prediction model that can improve the rank-

Ranking	$F1$	# Improved Qs
AgendaIL	49.7	-
w/ Oracle Types@10	57.3	+234
w/ Oracle Types@20	58.7	+282
w/ Oracle Types@50	60.1	+331
w/ Oracle Types@All	60.5	+345

Table 2.1: What if the correct answer type is enforced? On WebQuestions, we remove those with incorrect answer types in the top-k logical forms returned by AgendaIL [1], a leading semantic parsing system, and report the new average F1 score as well as the number of questions with an improved F1 score.

ing of the candidate logical forms generated by semantic parsing. The type of an entity, e.g., `person`, `organization`, `location`, carries very useful information for various downstream natural language processing tasks such as co-reference resolution [11], knowledge base population [12], relation extraction [13, 14], and question answering [15]. Although the potential clues for answer type from the question has been employed in the recent work AgendaIL [1] at the lexical level, Table 2.1 suggests that there is yet a large room for further improvement by explicitly enforcing answer type. Inspired by this observation, we aim to directly predict the KB type of the answer from the question. In contrast to a small set of pre-defined types as used in previous answer type prediction methods (e.g., [16]), KBs could have thousands of fine-grained types. Take "When did Shaq come into the NBA?" as a running example. We aim to predict the KB type of its answer as `SportsLeagueDraft`.[1]

The value of typing answers in a fine granularity can be appreciated from two perspectives: (1) Since each entity in a KB like Freebase has a few types, answer type could help prune answer candidates, (2) since each predicate in the KB has a unique type schema, answer type can help rank logical forms.

The key challenge of using answer types to re-rank logic forms and hence their corresponding answers, is that it shall be done before the answer is found. Otherwise, there

[1]KB type of answer ("1992 NBA Draft") in the context.

is no need to further infer its type. Inspired by the observation that the answer type of a question is invariant as long as the type of the topic entity (`Shaq`) remains the same (`DraftedAthlete`), we define **abstract question** as the question where the topic entity mention is replaced by its corresponding KB type. For the aforementioned example, the best candidate abstract question is "When did `DraftedAthlete` come into the NBA?" and the answer to this question is `SportsLeagueDraft`. Hence, we can reduce the answer type prediction task to abstract question answering.

The first step in our method is question abstraction, in which we generate candidate abstract questions based on the context of question and its candidate topic entities. We build a bidirectional LSTM network over the question that recursively computes vector representations for the past and future contexts of an entity mention. Based on these context representations, we predict the right type of the entity mention. Next, in order to better utilize the syntactic features of the question, we convert the question form into a normal statement form by using dependency tree of the question. For the running example, after performing the conversion, the abstract question becomes "`DraftedAthlete` come when into the NBA?" We then construct a bidirectional LSTM neural network over this final representation of the question and predict the type of the answer. Using the inferred answer type, we are able to improve the result of AgendaIL [1] on WebQuestions [17] from 49.7% to 52.6%.

2.2 Related Work

Freebase QA has been studied from two different perspectives: grounded QA systems that work directly on KBs and general purpose ungrounded QA systems. [18] generates KB agnostic intermediary CCG parses of questions which are grounded afterwards given a KB. [19] uses a vector space embedding approach to measure the semantic similarity

between question and answers. [20], [21] and [3] exploit a graph centric approach where a grounded subgraph query is generated from question and then executed against a KB. In this work, we propose a neural answer type inference method that can be incorporated in existing grounded semantic parsers as a complementary feature to improve ranking of the candidate logical forms.

[1] uses lambda DCS logical language with predicates from Freebase. In their approach, types are included as a part of unary lexicon for building the logical forms from natural language questions. However, no explicit type inference is exploited. We show that such information could indeed be useful for selecting logical forms.

There have been a series of studies investigating the expected answer type of a question in different contexts such as [16], [22], and [23]. Most of these approaches classify the questions into a small set of types. Even when the set of classes is more fine-grained, e.g., 50 classes in [16], they cannot be used for our purpose as it would require nontrivial mapping between these categories and a much larger number of KB types. Furthermore, these methods often rely on a rich set of hand crafted features and external resources.

[24] uses Freebase types to learn the relevance of candidate answers to a given question via an association model. Their model directly ranks the answer candidates by utilizing types, whereas ours ranks the logical forms via predicting answer type. In this sense, we are able to take advantage of both logical form and type inference. [25] exploits answer typing to facilitate knowledge graph search, but their input is graph query instead of natural language question. They predict answer types using additional relevance feedback for graph queries, while our algorithm directly infers answer types from input questions. On the question abstraction side, our work is related to a recent study [26] which classifies entity mentions into 22 types derived from DBpedia. They use a multilayer perceptron over a fixed size window and a recurrent neural network for the representations of context and entity mention, respectively.

2.3 Background

The knowledge base we work with consists of triples in subject-predicate-object form. It can be represented as $\mathcal{K} = \{(e_1, p, e_2) : \ e_1, e_2 \in \mathcal{E}, \ p \in \mathcal{P}\}$, where $\mathcal{E}$ denotes the set of entities (e.g., `ShaquilleOneal`), and $\mathcal{P}$ denotes the set of binary predicates (e.g., `Drafted`). A knowledge base in this format can be visualized as a graph where entities are nodes, and predicates are directed edges between entities. Freebase is used in this work as the knowledge base. It has more than 41M entities, 596M facts, and 24K types.

Types are an integral part of the Freebase schema. Each entity e in Freebase has a set of categories (types) it belongs to, and this information can be obtained by checking the out-going predicates (`Type.Object.Type`) from e. For example, `ShaquilleOneal` has 20 Freebase types including `Person`, `BasketballPlayer`, `DraftedAthlete`, `Celebrity`, and `FilmActor`. For a specific question involving `ShaquilleOneal`, among these types, only a few will be relevant.

Each predicate in Freebase is from a subject entity to an object entity, and has a type signature. It has a unique expected types for its subject and object, independent of the individual subject and object entities themselves. For example, the predicate `People.Person.Profession` expects its subject to be of `Person` type and its object to be of `Profession` type.

2.4 Question Abstraction

The type of the topic entity rather than the entity itself is essential for inferring the answer type, which is invariant as the topic entity changes within the same class. For example, independent of which NBA player (with `DraftedAthlete` type) is the topic entity of this question "When did Shaq come into the NBA", the type of the answer is

always going to be `SportsLeagueDraft` in Freebase. Predicting this distinct type among the large number of candidate types in Freebase is a challenging task. We propose a two-step solution for this problem. In the first step, we compute a confidence score for each possible KB type for a given topic entity using a bidirectional LSTM network. The second step prunes candidate types using the entity type information in Freebase.

2.4.1 Formulation

Given a natural language question and its topic entity mention, question abstraction is to predict types of the mention in the question context. Formally, let $q = (x_1, x_2, \ldots, x_L)$ denote the question, m be the topic entity mention in q, and $T = \{t_1, t_2, \ldots, t_K\}$ the set of all types in KB. Given q and m, we compute a probability distribution $o \in \mathbb{R}^{K\times 1}$ over T, where o_k denotes the likelihood of t_k being the correct type of m in q.

2.4.2 Scoring Topic Entity Types with LSTM

Model. We formulate question abstraction as a classification problem. A bidirectional LSTM network is built over q whose output is computed from the nodes that correspond to the words of m. Fig. 2.1 illustrates the model for the question "When did Shaq come into the NBA?"

Let $u(x) \in \mathbb{R}^{D\times 1}$ denote the vector space embedding of word x. Forward and backward outputs $\overrightarrow{h}_l, \overleftarrow{h}_l \in \mathbb{R}^{D_h\times 1}$ of bidirectional LSTM are recursively computed by

$$\overrightarrow{h}_l, \overrightarrow{c}_l = LSTM(u(x_l), \overrightarrow{h}_{l-1}, \overrightarrow{c}_{l-1}) \tag{2.1}$$

$$\overleftarrow{h}_l, \overleftarrow{c}_l = LSTM(u(x_l), \overleftarrow{h}_{l+1}, \overleftarrow{c}_{l+1}) \tag{2.2}$$

as described in [27], where $\overrightarrow{c_l}, \overleftarrow{c_l} \in \mathbb{R}^{D_h\times 1}$ stand for LSTM cell states.

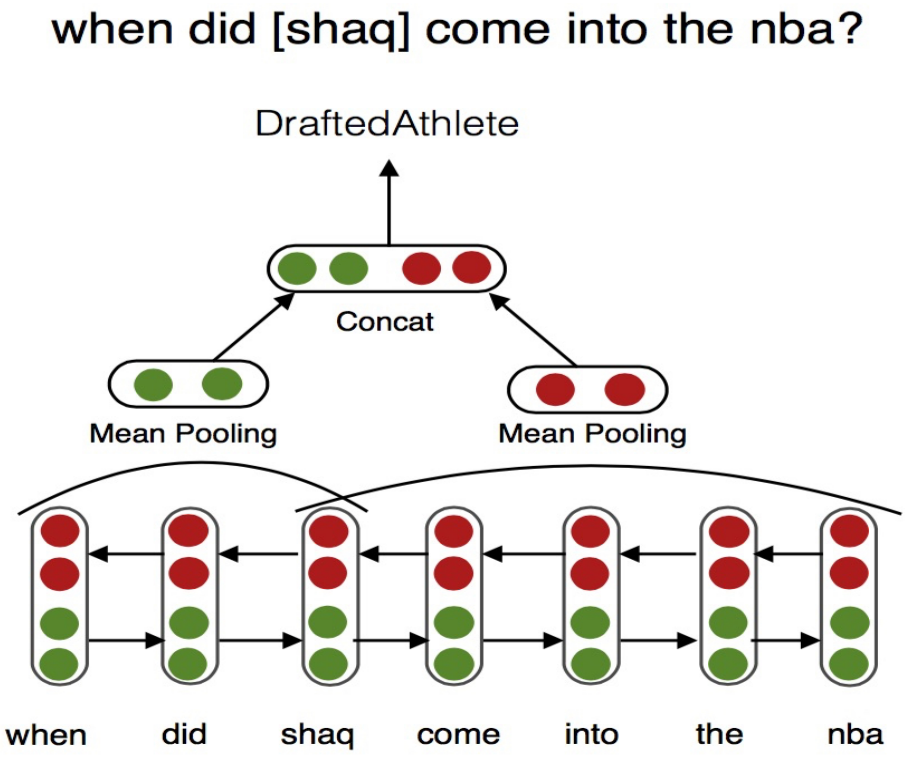

Figure 2.1: Bi-directional LSTM model for question abstraction. Green circles represent the forward sequence's hidden vectors, while the red circles denote the backward sequence's. `shaq` (the topic entity mention) is the single output node of the network.

To encode the context of m to the final output, we apply an AVERAGE pooling layer when computing the output. For each output node $r \in [i, j]$ (i and j correspond to the starting and ending indices of m in q), we compute final forward and backward outputs by

$$\overrightarrow{v_r} = AVG(\overrightarrow{h_1}, \ldots, \overrightarrow{h_r}) \tag{2.3}$$

$$\overleftarrow{v_r} = AVG(\overleftarrow{h_r}, \ldots, \overleftarrow{h_n}), \tag{2.4}$$

where AVG stands for average pooling.

We take the average of outputs at each output node

$$\overrightarrow{v} = AVG(\overrightarrow{v_i}, \ldots, \overrightarrow{v_j}) \tag{2.5}$$

$$\overleftarrow{v} = AVG(\overleftarrow{v_i}, \ldots, \overleftarrow{v_j}) \tag{2.6}$$

as the forward and backward outputs of the whole network. The final representation v of the network is obtained by concatenating $\overrightarrow{v}$ and $\overleftarrow{v}$.

For question q, the probability distribution o over types is computed by

$$s(q) = W_{hy} v \tag{2.7}$$

$$o(q) = softmax(s(q)), \tag{2.8}$$

where $W_{hy} \in \mathbb{R}^{K \times (2D_h)}$ since v is the concatenation of two vectors of dimension D_h, where D_h is the hidden vector dimension.

Objective Function and Learning. Given an input question q with a topic entity mention m, LSTM network computes the probability distribution $o(q) \in \mathbb{R}^{K \times 1}$ as in (2.8). Let $y(q) \in \mathbb{R}^{K \times 1}$ denote the true target distribution over T for q, where $y_k(q) = 1/n$ if t_k is a correct type, $y_k(q) = 0$ otherwise, and n is the number of correct types. We use the cross-entropy loss function between $y(q)$ and $o(q)$, and define the objective function over all training data as

$$J(\theta) = -\sum_{q} \sum_{k=1}^{K} y_k(q) \log o_k(q) + \frac{\lambda}{2} \|\theta\|^2,$$

where λ denotes the regularization parameter, and θ represents the set of all model parameters to be learned. We use stochastic gradient descent with RMSProp [28] for minimizing the objective function.

2.4.3 Pruning

Let T_e represent the set of KB types for entity e. We define the set of candidate types for entity mention m as

$$C_m = \bigcup_e T_e,$$

where e is a possible match of m in KB. We only need to score the types in C_m. Once the hidden representation v is computed by LSTM, we use submatrix $W_{hy}[C_m]$ that consists of rows of W_{hy} corresponding to the types in C_m as the scoring matrix in (2.7). This returns the final scores for candidate types in C_m.

2.5 Conversion to Statement Form

The objective of the conversion is to canonicalize question form into declarative statement (subject-relation-object) form. We use a simple pattern-based method that relies on dependency tree[2] [29]. It decides whether the sub-trees of the root need reordering based on their dependency relations[3].

Before obtaining the dependency tree, we retrieve named entity (NER) tags of the question tokens. We replace a group of question tokens corresponding a named entity with a special token, ENTITY, to simplify the parse tree. In Figure 2.2, the question is first transformed to "what boarding school did ENTITY go to?" Each question is represented by the root's dependency relations to its sub-trees in the original order, e.g., (`dep`, `aux`, `nsubj`, `nmod`). We cluster all these sequences and detect the patterns that appear at least 5 times in the training data. These patterns are then manually mapped to their corresponding conversion (pattern vs. mapping in Figure 2.2).

[2]We use Stanford CoreNLP dependency parser
[3]http://universaldependencies.org

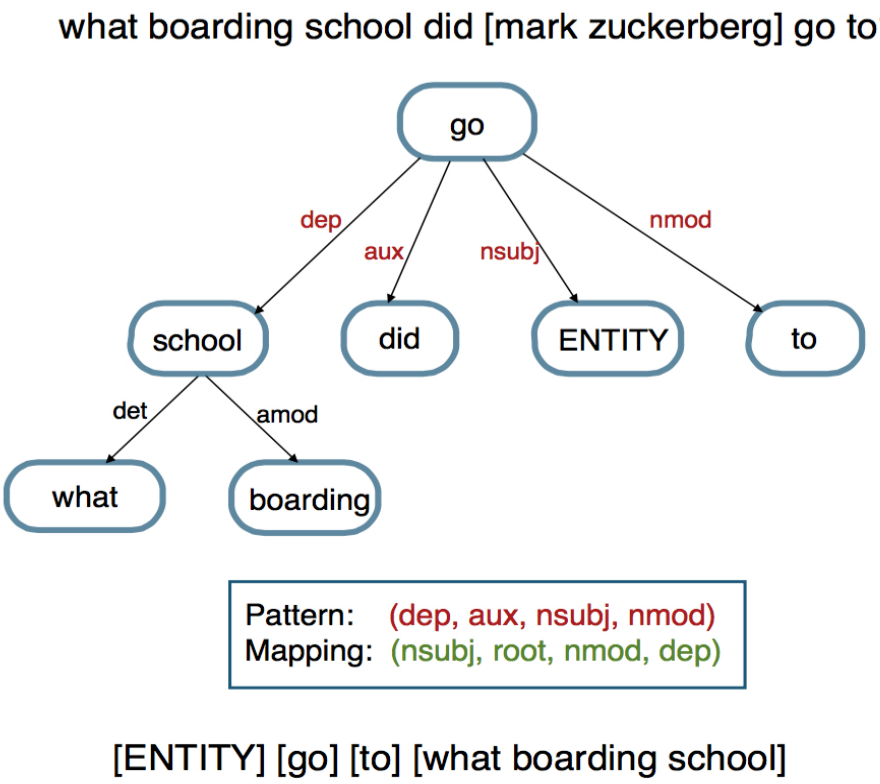

Figure 2.2: Conversion: red relations form the input pattern

Once the recomposition order of the sub-trees is determined by the conversion mapping, we finalize the reordering of the question tokens by keeping the order of words within the sub-trees same as the original order in the question. The example in Figure 2.2 becomes "ENTITY go to what boarding school" with its corresponding sub-tree conversion mapping (`nsubj`, `root`, `nmod`, `dep`). If no mapping is created for a pattern, we keep the order of the words exactly as they occur in the original question form.

The motivation behind conversion is to overcome the potential semantic confusion stemming from varities in syntactic structures. To exemplify, consider two hypothetical questions "who plays X in Y?" and "who does Z play in Y?", where X is a `FilmCharacter`, Y is a `Film`, and Z is a `FilmActor`, with answer types `FilmActor` and `FilmCharacter`, respectively. With conversion, we aim to transform second question into "Z play who in

Pattern	Conversion
(cop, nsubj) who was anakin skywalker?	(nsubj, root, cop) anakin skywalker was who
(dobj, aux, nsubj) what language does australians speak?	(nsubj, root, dobj) australians speak what language
(dobj, aux, nsubj, nmod) what did edward jenner do for a living?	(nsubj, root, dobj, nmod) edward jenner do what for a living
(nsubj, dobj) who played bilbo baggins?	(nsubj, root, dobj) who played bilbo baggins
(advmod, aux, nsubj) where did benjamin franklin died?	(nsubj, root, advmod) benjamin franklin died where

Table 2.2: Top-5 most common patterns with mappings.

Y", while leaving the first one as it is. Noting that the order of words affects the output of our answer type inference network, our intuition is to let the model distinguish better between such questions using their syntactic structure in this way.

2.6 Answer Type Prediction

Given a reordered question with topic entity mention m, and a topic entity type $t_e \in T$, our task is to predict a probability distribution $o \in \mathbb{R}^{K\times 1}$ over the answer types.

A topic entity type $t_e \in T$ is described as a set of words, $\{x_i\}$. Let $u(x_i) \in \mathbb{R}^{D\times 1}$ represent the vector space embedding of x_i, the representation of t_e is computed by the average encoding,

$$u(t_e) = \frac{1}{|\{x_i\}|} \sum_{x_i} u(x_i). \tag{2.9}$$

As the first step, we replace the words of entity mention m with topic entity type t_e, and obtain a new input word sequence r. t_e is treated as one word and encoded by Eq. 2.9. We construct a bidirectional LSTM network over this input sequence r, whose output node corresponds to the question word. The output of the network is

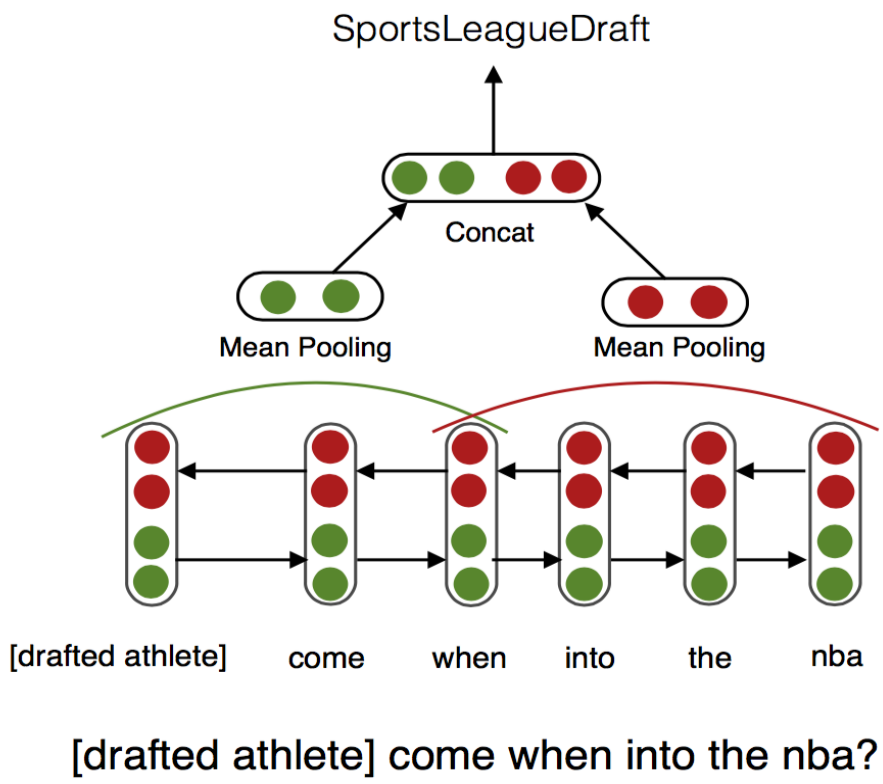

Figure 2.3: Bi-directional LSTM model over the final representation of the question. Green and red circles are corresponding to forward and backward hidden vectors, respectively. The output node is `when`.

a probability distribution over types denoting the likelihood of being the answer type. Figure 2.3 shows how the network is constructed for the running example. The same average pooling described in Section 2.4.2 is applied to obtain the final forward and backward output vectors $\overrightarrow{v}$ and $\overleftarrow{v}$ from the output node (this time, single output node) of network. The final output vector v for prediction is obtained by concatenating $\overrightarrow{v}$, and $\overleftarrow{v}$. The distribution o is computed by a standard softmax layer. The learning is performed by the same cross-entropy loss and objective function described in Section 2.4.2.

2.7 Reranking by Answer Type

We now describe how to rerank logical forms based on our answer type predictions.

Reranking Model. Let $l_1, l_2, \ldots, l_N$ be the logical forms generated for question q by a semantic parser, e.g., AgendaIL. Each logical form has a score from the semantic parser. Meanwhile, our answer type prediction model generates a score for the answer type of each logical form. Therefore, we can represent each logical form l_i using a pair of scores: the score from semantic parser and the score from our type prediction model. Suppose we know which logical forms are "correct", using the two scores as input, we train a logistic regression model with cross-entropy loss to learn a binary classifier for predicting the correct logical forms. We rerank the top-k logical forms using their probability computed by the trained logistic regression model, and select the one with the highest probability. Finally, we run the selected logical form against KB to retrieve the answer. We select the optimal value of k from $[1, N]$ using the training data. For AgendaIL on WebQuestions, we find that $k = 80$ gives the best result.

Training Data Selection. We now discuss which logical forms are "correct", i.e., how to select the positive examples to train the logistic regression model. Because a question can have more than one answer, we use the $F1$ score, the harmonic mean of *precision* and *recall*, to evaluate logical forms. We select all the logical forms with $F1 > 0$ as the set of positive examples. However, taking all the logical forms with $F1 = 0$ as negative examples will not work well. Even though the $F1$ score of a logical form is 0, its answer type could still be correct. Therefore, we use the following trick: If there is a positive example with answer type t, we do not treat any other logical form with answer type t as negative example. The logical forms having $F1 = 0$, with the aforementioned exception, are then selected as the final set of negative examples. Our empirical study shows this trick works well.

2.8 Experiments

In this section, we describe the datasets, model training, and experimental results.

2.8.1 Dataset and Evaluation Metrics

Datasets. To evaluate our method, we use the WebQuestions dataset [17], which contains 5,810 questions crawled via Google Suggest API. The answers to these questions are annotated from Freebase using Amazon Mechanical Turk. The data is split into training and test sets of size 3,778 and 2,032 questions, respectively. This dataset has been popularly used in question answering and semantic parsing.

The SimpleQuestions [30] contains 108,442 questions written in natural language by English-speaking human annotators. This dataset is a collection of question/Freebase-fact pairs rather than question/answer pairs. The data[4] is split and provided as training(75,910), test(21,687), and validation(10,845) sets. Each question is mapped to the subject, relation, and object of the corresponding Freebase fact. This dataset is only used for training the question abstraction model.

Training Data Preparation. Since WebQuestions only provides question-answer pairs along with annotated topic entities, we need to figure out the type information, which can be used as training data. We obtain *simulated* types as follows: We retrieve 1-hop and 2-hop predicates r from/to annotated topic entity e in Freebase. For each relation r, we query $(e, r, ?)$ and $(?, r, e)$ against Freebase and retrieve the candidate answers r_a. The $F1$ value of each candidate answer r_a is computed with respect to the annotated answer. The subject and object types of the relation r with the highest $F1$ value is selected as the *simulated* type for the topic entity and the answer. When there are multiple such relations, we obtain multiple *simulated* types for topic entity and answer,

[4] http://fb.ai/babi.

one from each relation. We treat each of them as correct with equal probability.

Candidate Logical Forms for Evaluation. To obtain candidate logical forms, we train AgendaIL [1] on WebQuestions with beam size 200 using the publicly available code[5] by the authors.

Evaluation Metric. We report average $F1$ score of the reranked logical forms using the predicted answer types as the main evaluation metric. It is a common performance measure in question answering as questions might have multiple answers.

2.8.2 Experimental Setup

We use 50 dimensional word embeddings, which are initialized by the 50 dimensional pre-trained word vectors[6] from GloVe [31], and updated in the training process. Hyper-parameters are tuned on the development set. The size of the LSTM hidden layer is set at 50. We use RMSProp [28] with a learning rate of 0.005 and mini-batch size of 32 for the optimization. We use a dropout layer with probability 0.5 for regularization. We implemented the LSTM networks using *Theano* [32].

Identifying Topic Entity. We use Stanford NER tagger [29] to identify topic entity span for both training and test data. For entity linking, annotated mention span is mapped to a ranked list of candidate Freebase entities using Freebase Search API for the test data. For the training data, we use the gold Freebase topic entity linkings of each question provided by WebQuestions, coming from its question generation process.

Question Abstraction. We first pre-train the LSTM model described in Section 2.4.2 on the SimpleQuestions dataset. Then, we update the pre-trained model on the training portion of WebQuestions data where the *simulated* topic entity types are used as true labels. We use the detected topic entity mentions to obtain candidate matching

[5]https://github.com/percyliang/sempre
[6]http://nlp.stanford.edu/projects/glove/

Model	F1
[17]	35.7
[33]	33.0
[34]	39.9
[35]	37.5
[19]	39.2
[36]	41.3
[37]	40.8
[38]	44.3
AgendaIl [1]	49.7
Stagg [3]	52.5
[39]	50.3
[40]	**53.3**
Stagg [3] (w/ Freebase API)	48.4
Stagg [3] (w/o ClueWeb)	50.9
[40] (w/o Wikipedia)	47.1
Our Approach (w/o SimpleQuestions)	51.6
Our Approach	**52.6**

Table 2.3: Comparison of our reranking-by-type system with several existing works on WebQuestions.

entities in the KB using Freebase Search API. We use top-3 entities returned for the pruning step of *Question Abstraction* on the test examples.

Answer Type Prediction. We train *Answer Type Prediction* model using the *simulated* topic entity and answer types for each question. We perform the answer type prediction on test data using the predicted topic entity type.

2.8.3 Results

Our main result is presented in Table 3.3. Our system adds 2.9% absolute improvement over AgendaIL, and achieves 52.6% in $F1$ measure. staged-15 achieve 52.5% by leveraging ClueWeb and S-MART [41], an advanced entity linking system. text-ev-16 achieve 53.3% by leveraging Wikipedia and S-MART. If tested without Clueweb/Wikipedia/S-MART, their F1 scores are 48.4% and 47.1%, respectively. When our method is tested

Question	T.E Type Prediction	A.T Prediction	AgendaIL A.T	Gain
who inspired **obama**?	InfluenceNode	InfluenceNode	UsVicePresident	1.0
what are some books that **mark twain** wrote?	Author	WrittenWork	InfluenceNode	0.3
who won the **league cup** in 2002?	SportsAwardType	SportsAwardWinner	SportsLeagueSeason	1.0
what type of government does **france** use?	Country	FormOfGovernment	Government	1.0
where are the **new orleans hornets** moving to?	SportsTeam	SportsFacility	Location	1.0
who did **australia** fight in the first world war?	MilitaryCombatant	MilitaryCombatant	MilitaryCommander	0.4
what guitar does **corey taylor** play?	Musician	MusicalInstrument	Organization	0.33
what region is **turkey** considered?	Location	AdministrativeDivision	Beer	0.93
what country does **rafael nadal** play for?	Athlete	Country	OlympicDiscipline	1.0

Table 2.4: Example questions where our type prediction helps select a better logical form. Gain column shows the F1 gain: the difference between the F1 score of the logical form we select and the top ranked logical form from AgendaIL. **T.E** and **A.T** are abbreviations for topic entity and answer type, respectively.

Questions and Selected Logical Forms
1. what are some books that **mark twain** wrote? AgendaIL: (`MarkTwain - Influence.InfluenceNode.InfluencedBy - ?`) Ours: (`MarkTwain - Book.Author.WorksWritten - ?`)
2. what guitar does **corey taylor** play? AgendaIL: (`? - Organization.Organization.Founders - CoreyTaylor`) Ours: (`CoreyTaylor - Music.GroupMember.InstrumentsPlayed - ?`)
3. what type of government does **france** use? AgendaIL: (`France - Government.GovernmentalJurisdiction.Government - ?`) Ours: (`France - Location.Country.FormOfGovernment - ?`)

Table 2.5: Comparison of selected logical forms for some examples. Logical forms are simplified and canonicalized into (subject - predicate - object) format for better readibility, where `?` corresponds to answer nodes.

without using SimpleQuestions data for pretraining question abstraction module, it attains F1 score of 51.6%.

In Table 2.4, we present some question examples where our method can select a better logical form. Take the question "who did [australia] fight in the first world war?" as an example. Our topic entity type prediction module returns `MilitaryCombatant`, `StatisticalRegion`, and `Kingdom` as the top-3 results for the type of "australia" in this question, which indicates that it exploits the context of this short question successfully. The abstract question is "[military combatant] fight who in the first world war?" for which our system returns `MilitaryCombatant`, `MilitaryConflict`, and `MilitaryCommander` as answer types with probabilities 0.73, 0.25, and 0.005, respectively, `MilitaryCombatant`

Method	F1	Gain	Loss
Base	50.3	69	47
Base + Conv	50.5	96	56
Base + Abs	52.2	184	87
Base + Abs + Conv	52.6	203	93
AgendaIL	49.7	-	-

Table 2.6: Ablation analysis of modules of our method. Gain/Loss columns denote the number of questions where the F1 score of our selected logical form is greater/less than that of the top ranked logical forms from AgendaIL.

is indeed the right answer type. This example shows the effect of abstraction in channeling the context in the most relevant direction to find the right answer type. In Table 2.5, we provide a comparison of the selected logical forms based on AgendaIL rankings and our rankings.

2.8.4 Ablation Analysis

In this section, we evaluate the effect of individual components of our model. Note that the answer type prediction model described in Section 2.6 can work independently from question abstraction and form conversion. We develop the following variants i) Base, ii) Base + Conversion, iii) Base + Abstraction, iv) Base + Abstraction + Conversion, where Base corresponds to a model that infers answer types without employing abstraction or form conversion. We train/test each variant separately. Table 2.6 shows each component contributes and question abstraction does help boost the performance.

Suppose we perform answer type prediction without question abstraction, and feed "[australia] fight who in the first world war?" into the answer type prediction model (Base + Conversion). The predicted answer type is `Location`. Unfortunately, there is neither a 1-hop or 2-hop correct relation from/to `Australia` with the expected type `Location` nor a correct (with positive $F1$) candidate logical form with the answer type `Location`. This shows that through question abstraction, a better logical form is selected

for this question.

To exemplify another benefit of question abstraction, consider the question "where does [marta] play soccer?" The top 3 entity linkings via Freebase Search API for "marta" are `MetropolitanAtlantaRapidTransitAuthority`, `Marta`, and `SantaMarta`, where the correct entity is the second one. Our question abstraction system returns `FootballPlayer` as the top topic entity type prediction that is indeed corresponding to the correct entity. Utilizing the context via question abstraction we are able to recover useful information when the entity linking is uncertain.

Table 2.6 also shows that the conversion to statement form also helps, especially together with *Abstraction*. In the above example, the model without *Conversion* (Base + Abs) predicts the answer type for "where does [football player] play soccer" as `SportsFacility`, whereas the full model, considering `Conversion` as well, finds the answer type for "[football player] play soccer where" as `SportsTeam` which is the better type in this case.

2.8.5 Error Analysis

We present a further analysis of our approach by classifying the type inference errors made on randomly sampled 100 questions. 9% of the errors are due to inference at incorrect granularity (e.g., `City` instead of `Location`). 12% of the errors are the result of incorrect answer labels (hence incorrect answer types) or question ambiguity (e.g., "where is dwight howard now?"). 11% of them are incorrect, but acceptable inferences, e.g., `BookWrittenWork` instead of `BookEdition` for question "what dawkins book to read first?" 39% of the errors are due to the sparsity problem: They are made on questions whose answer type appears less than 5 times in the training data (e.g., `DayOfYear`). The remaining 29% of them are due to incorrect question abstraction. In most of the question abstraction errors, the predicted topic entity type is semantically close to the

correct type. In other cases such as "what did joey jordison play in slipknot?" where we predict `FilmActor` as the topic entity type while `Musician` is the correct one. In these cases, the answer type inference is not able to correct the abstraction error. These 29% of errors also contain the entity linking errors.

2.9 Conclusion

In this section, we present a question answer type inference framework and leverage it to improve semantic parsing. We define the notion of abstract question as the class of questions that can be answered by type instead of entity. Question answer type inference is then reduced to "question abstraction" and "abstract question answering", both of which are formulated as classification problems. Question abstraction is performed by exploiting the topic entity and its context in question via an LSTM network . A separate neural network is trained to exploit the abstraction to make the final question answer type inference. Our method improves the ranking of logical forms returned by AgendaIL on the WEBQUESTIONS dataset. In the future, we would like to investigate how the abstraction and explicit type inference can be incorporated in the early stage of semantic parsing for generating better candidate logical forms.

Chapter 3

Recovering Question Answering Errors via Query Revision

3.1 Introduction

With the recent advances in building large scale knowledge bases (KB) like Freebase [8], DBpedia [9], and YAGO [10] that contain the world's factual information, KB-based question answering receives attention of research efforts in this area. Traditional semantic parsing is one of the most promising approaches that tackles this problem by mapping questions onto logical forms using logical languages CCG [18, 42, 43, 39], DCS [17, 34, 1], or directly query graphs [3] with predicates closely related to KB schema. Recently, neural network based models have been applied to question answering [30, 3, 44, 40].

While these approaches yielded successful results, they often lack a post-inspection component that can help models recover from their own mistakes. Table 3.1 shows the potential improvement we can achieve if such a component exists. Can we leverage textual evidences related to the predicted answers to recover from a prediction error? In this work, we show it is possible.

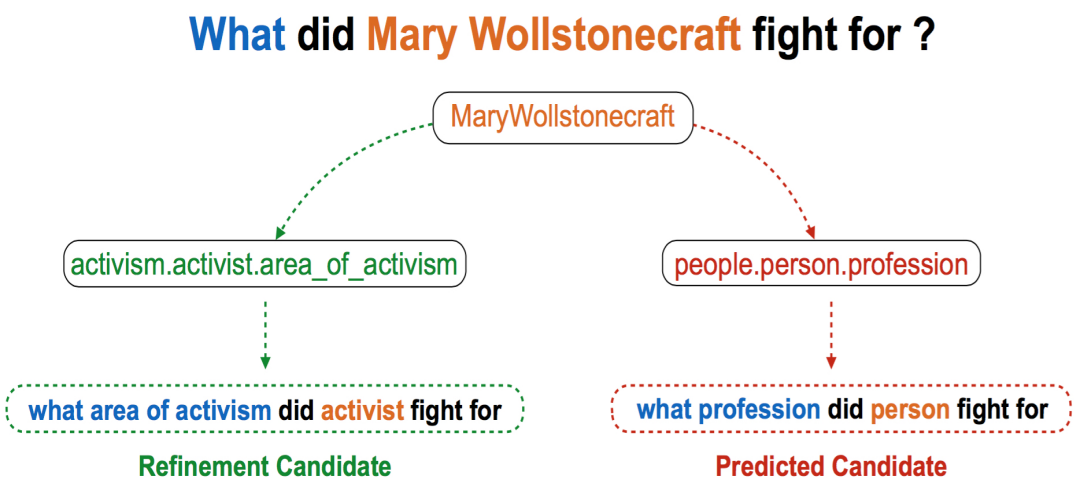

Figure 3.1: Sketch of our approach. Elements in solid round rectangles are KB relation labels. Relation on the left is correct, but the base QA system predicts the one on the right. Dotted rectangles represent **revised questions** with relation labels plugged in. The left revised question looks semantically closer to the original question and itself is more consistent. Hence, it shall be ranked higher than the right one.

Our strategy is to cross-check the corresponding KB relations behind the predicted answers and identify potential inconsistencies. As an intermediate step, we define **question revision** as a tailored transformation of the original question using textual evidences collected from these relations in a knowledge base, and check if the revised questions make sense or not. Figure 3.1 illustrates the idea over an example question "*what did Mary Wollstonecraft fight for ?*" Obviously, "*what [area of activism] did [activist] fight for ?*" looks more consistent over "*what [profession] did [person] fight for ?*" We shall build a model that prefers the former one. This model shall be specialized for comparing the revised questions and checking which one makes better sense, not for answering the revised questions. This strategy differentiates our work from many existing QA studies.

Given a question, we first create its revisions with respect to candidate KB relations. We encode question revisions using a bidirectional LSTM. A scoring mechanism over these encodings is jointly trained with LSTM parameters with the objective that the question revised by a correct KB relation has higher score than that of other candidate KB

Refinement	F_1	# Refined Qs
STAGG	52.5	-
w/ Best Alternative	58.9	639

Table 3.1: What if we know the questions on which the system makes mistakes? Best alternative is computed by replacing the predictions of **incorrectly answered questions** by STAGG with its second top-ranked candidate.

relations by a certain confidence margin. We evaluate our method using STAGG [3] as the base question answering system. Our approach is able to improve the F_1 performance of STAGG [3] from 52.5% to 53.9% on a benchmark dataset WEBQUESTIONS [17]. Certainly, one can develop specialized LSTMs that directly accommodate text evidences without revising questions. Towards exploring this direction as an alternative solution, we have modified QA-LSTM and ATTENTIVE-LSTM [45] accordingly (See Section 3.4). However, their performance are not as good as the question revision approach.

3.2 Question Revisions

We formalize three kinds of question revisions, namely entity-centric, answer-centric, and relation-centric that revise the question with respect to evidences from topic entity type, answer type, and relation description. As illustrated in Figure 3.2, we design revisions to capture generalizations at different granularities while preserving the question structure.

Let s_r (e.g., `Activist`) and o_r (e.g., `ActivismIssue`) denote the subject and object types of a KB relation r (e.g., `AreaOfActivism`), respectively. Let α (`type.object.name`) denote a function returning the textual description of a KB element (e.g., relation, entity, or type). Assuming that a candidate answer set is retrieved by executing a KB relation r from a topic entity in question, we can uniquely identify the types of topic entity and answer for the hypothesis by s_r and o_r, respectively. It is also possible that a chain of

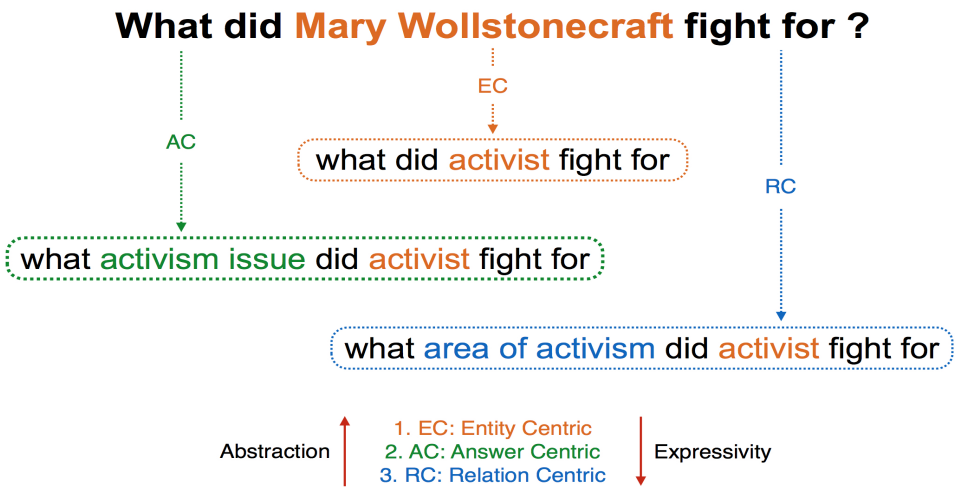

Figure 3.2: Illustration of different question revision strategies on the running example w.r.t KB relation `activism.activist.area_of_activism`.

relations $r = r_1 r_2 \dots r_k$ is used to retrieve an answer set from a topic entity. When $k = 2$, by abuse of notation, we define $s_{r_1 r_2} = s_{r_1}$, $o_{r_1 r_2} = o_{r_2}$, and $\alpha(r_1 r_2) = concat(\alpha(r_1), \alpha(r_2))$.

Let $m : (q, r) \mapsto q'$ denote a mapping from a given question $q = [w_1, w_2, \dots, w_L]$ and a KB relation r to revised question q'. We denote the index span of wh-words (e.g., "what") and topic entity (e.g., "Mary Wollstonecraft") in question q by $[i_s, i_e]$ and $[j_s, j_e]$, respectively.

Entity-Centric (EC). Entity-centric question revision aims a generalization at the entity level. We construct it by replacing topic entity tokens with its type. For the running example, it becomes "*what did [activist] fight for*". Formally, $m_{EC}(q, r) = [w_{[1:j_s-1]}; \alpha(s_r); w_{[j_e+1:L]}]$.

Answer-Centric (AC). It is constructed by augmenting the wh-words of entity-centric question revision with the answer type. The running example is revised to "*[what activism issue] did [activist] fight for*". We formally define it as $m_{AC}(q, r) = [w'_{[1:i_e]}; \alpha(o_r); w'_{[i_e+1:L']}]$, where w'_i's are the tokens of entity-centric question revision $m_{EC}(q, r)$ of length L' with $[i_s, i_e]$ still denoting the index span of wh-words in w'.

Relation-Centric (RC). Here we augment the wh-words with the relation description

instead of answer type. This form of question revision has the most expressive power in distinguishing between the KB relations in question context, but it can suffer more from the training data sparsity. For the running example, it maps to "*[what area of activism] did [activist] fight for*". Formally, it is defined as $m_{RC}(q, r) = [w'_{[1:i_e]}; \alpha(r); w'_{[i_e+1:L']}]$.

3.3 Model

3.3.1 Task Formulation

Given a question q, we first run an existing QA system to answer q. Suppose it returns r as the top predicted relation and r' is a candidate relation that is ranked lower. Our objective is to decide if there is a need to replace r with r'. We formulate this task as finding a scoring function $s : (q, r) \rightarrow \mathbb{R}$ and a confidence margin threshold $t \in \mathbb{R}_{>0}$ such that the function

$$replace(r, r', q) = \begin{cases} 1, \text{ if } s(q, r') - s(q, r) \geq t \\ 0, \text{ otherwise} \end{cases} \tag{3.1}$$

makes the replacement decision.

3.3.2 Encoding Question Revisions

Let $q' = (w'_1, w'_2, \ldots, w'_l)$ denote a question revision. We first encode all the words into a d-dimensional vector space using an embedding matrix. Let $\mathbf{e}_i$ denote the embedding

of word w'_i. To obtain the contextual embeddings for words, we use bi-directional LSTM

$$\overrightarrow{\mathbf{h}}_i = LSTM_{fwd}(\overrightarrow{\mathbf{h}}_{i-1}, \mathbf{e}_i) \tag{3.2}$$

$$\overleftarrow{\mathbf{h}}_i = LSTM_{bwd}(\overleftarrow{\mathbf{h}}_{i+1}, \mathbf{e}_i) \tag{3.3}$$

with $\overrightarrow{\mathbf{h}}_0 = \mathbf{0}$ and $\overleftarrow{\mathbf{h}}_{l+1} = \mathbf{0}$. We combine forward and backward contextual embeddings by $\mathbf{h}_i = \mathbf{concat}(\overrightarrow{\mathbf{h}}_i, \overleftarrow{\mathbf{h}}_i)$. We then generate the final encoding of revised question q' by $\mathbf{enc}(q') = \mathbf{concat}(\mathbf{h}_1, \mathbf{h}_l)$.

3.3.3 Training Objective

Score Function. Given a question revision mapping m, a question q, and a relation r, our scoring function is defined as $s(q, r) = \boldsymbol{w}^T \mathbf{enc}(m(q, r))$ where $\boldsymbol{w}$ is a model parameter that is jointly learnt with the LSTM parameters.

Loss Function. Let $\mathcal{T} = \{(q, a_q)\}$ denote a set of training questions paired with their true answer set. Let $U(q)$ denote the set of all candidate KB relations for question q. Let $f(q, r)$ denote the F_1 value of an answer set obtained by relation r when compared to a_q. For each candidate relation $r \in U(q)$ with a positive F_1 value, we define

$$N(q, r) = \{r' \in U(q) : f(q, r) > f(q, r')\} \tag{3.4}$$

as the set of its negative relations for question q. Similar to a hinge-loss in [19], we define the objective function $J(\boldsymbol{\theta}, \boldsymbol{w}, \boldsymbol{E})$ as

$$\sum_{(q,r,r')} max(0, \delta_\lambda(q, r, r') - s(q, r) + s(q, r')) \tag{3.5}$$

where the sum is taken over all valid $\{(q, r, r')\}$ triplets and the penalty margin is defined

as $\delta_\lambda(q, r, r') = \lambda(f(q, r) - f(q, r'))$.

We use this loss function because: i) it allows us to exploit partially correct answers via F_1 scores, and ii) training with it updates the model parameters towards putting a large margin between the scores of correct (r) and incorrect (r') relations, which is naturally aligned with our prediction refinement objective defined in Equation 3.1.

3.4 Alternative Solutions

Our approach directly integrates additional textual evidences with the question itself, which can be processed by any sequence oriented model, and benefit from its future updates without significant modification. However, we could also design models taking these textual evidences into specific consideration, without even appealing to question revision. We have explored this option and tried two methods that closely follow QA-LSTM and ATTENTIVE-LSTM [45]. The latter model achieves the state-of-the-art for passage-level question answer matching. Unlike our approach, they encode questions and evidences for candidate answers in parallel, and measure the semantic similarity between them using *cosine* distance. The effectiveness of these architectures has been shown in other studies [46, 47, 48, 49] as well.

We adopt these models in our setting as follows: (1) Textual evidences $\alpha(s_r)$ (equiv. of **EC** revision), $\alpha(o_r)$ (equiv. of **AC** revision) or $\alpha(r)$ (equiv. of **RC** revision) of a candidate KB relation r is used in place of a candidate answer a in the original model, (2) We replace the entity mention with a universal #entity# token as in [3] because individual entities are rare and uninformative for semantic similarity, (3) We train the score function $sim(q, r)$ using the objective defined in Eq. 3.5. Further details of the alternative solutions can be found in Appendix A.

3.5 Experiments

In this section, we discuss the details of experiments conducted in this study including data preprocessing, the specifics of implementations, and their results.

3.5.1 Datasets and Preprocessing

Datasets. For evaluation, we use the WebQuestions [17], a benchmark dataset for QA on Freebase. It contains 5,810 questions whose answers are annotated from Freebase using Amazon Mechanical Turk. We also use SimpleQuestions [30], a collection of 108,442 question/Freebase-fact pairs, for training data augmentation in some of our experiments, which is denoted by **+SimpleQ.** in results.

Training Data Preparation. WebQuestions only provides question-answer pairs along with annotated topic entities. We generate candidates $U(q)$ for each question q by retrieving 1-hop and 2-hop KB relations r from annotated topic entity e in Freebase. For each relation r, we query $(e, r, ?)$ against Freebase and retrieve the candidate answers r_a. Then, we compute $f(q, r)$ by comparing the answer set r_a with the annotated answers.

3.5.2 Implementation Details

Proposed Approach. Word embeddings are initialized with pretrained GloVe [31] vectors[1], and updated during the training. We take the dimension of word embeddings and the size of LSTM hidden layer equal and experiment with values in $\{50, 100, 200, 300\}$. We apply dropout regularization on both input and output of LSTM encoder with probability 0.5. We hand tuned penalty margin scalar λ as 1. The model parameters are optimized using Adam [50] with batch size of 32. We implemented our models in *tensorflow* [51].

[1] http://nlp.stanford.edu/projects/glove/

Question Side	Answer Side	Model Name
what did #entity# fight for	activist	ALT.-(equiv **EC**)
what did #entity# fight for	activism issue	ALT.-(equiv **AC**)
what did #entity# fight for	area of activism	ALT.-(equiv **RC**)

Table 3.2: Question (q) and answer (a) sides used for alternative (e.g., ALT.) solutions QA-LSTM and ATTENTIVE-LSTM.

To refine predictions r of a base QA system, we take its second top ranked prediction as the refinement candidate r', and employ $replace(r, r', q)$ in Eq. 3.1. Confidence margin threshold t is tuned by grid search on the training data after the score function is trained. QUESREV-**AC** + **RC** model is obtained by a linear combination of QUESREV-**AC** and QUESREV-**RC**, which is formally defined in Appendix B. To evaluate the alternative solutions for prediction refinement, we apply the same decision mechanism in Eq. 3.1 with the trained $sim(q, r)$ in Section 3.4 as the score function.

We use a dictionary[2] to identify wh-words in a question. We find topic entity spans using Stanford NER tagger [29]. If there are multiple matches, we use the first matching span for both.

Alternative Solutions. Following [45], we use the same bidirectional LSTM for both questions and textual evidences. For the attentive model, we apply the attention mechanism on the question side because our objective is to match textual evidences to the question context unlike the original model. We use average pooling for both models and compute the *general* attention via a bilinear term that has been shown effective in [52].

For the model and training parameters, we follow the strategy described in Section 5.1 with a difference that λ is tuned to be 0.2 in this setting. This intuitively makes sense because the score $sim(q, r)$ is in $[-1, 1]$. In Table 3.2, we further clarify the question and answer sides for the alternative models over the running example.

[2]what, who, where, which, when, how

Model	F_1
[37]	40.8
[38]	44.3
[1]	49.7
STAGG [3]	**52.5**
[39]	50.3
[40]	53.3
[44]	**53.8**
QUESREV on STAGG	**53.9**
Ensemble Model	
STAGG-RANK [53]	54.0
QUESREV on STAGG-RANK	**54.3**

Table 3.3: Comparison of our question revision approach (QUESREV) on STAGG with variety of recent KB-QA works.

3.5.3 Results

Table 3.3 presents the main result of our prediction refinement model using STAGG's results. Our approach improves the performance of a strong base QA system by 1.4% and achieves 53.9% in F_1 measure, which is slightly better than the state-of-the-art KB-QA system [44]. However, it is important to note here that [44] uses DBPedia knowledge base in addition to Freebase and the Wikipedia corpus that we do not utilize. Moreover, applying our approach on the STAGG predictions reranked by [53], referred as STAGG-RANK in Table 3.3, leads to a further improvement over a strong ensemble baseline. These suggest that our system captures orthogonal signals to the ones exploited in the base QA models. Improvements of QUESREV over both STAGG and STAGG-RANK are statistically significant.

In Table 3.4, we present variants of our approach. We observe that **AC** model yields to best refinement results when trained only on WEBQUESTIONS data (e.g., **WebQ.** column). This empirical observation is intuitively expected because it has more generalization power than **RC**, which might make **AC** more robust to the training data

Refinement Model	WebQ.	+ SimpleQ.
QA-LSTM-(equiv **EC**)	51.9	52.5
QA-LSTM-(equiv **AC**)	52.4	52.9
QA-LSTM-(equiv **RC**)	52.6	53.0
ATTENTIVE-LSTM-(equiv **EC**)	52.2	52.6
ATTENTIVE-LSTM-(equiv **AC**)	52.7	53.0
ATTENTIVE-LSTM-(equiv **RC**)	52.9	53.1
QUESREV-**EC**	52.9	52.8
QUESREV-**AC**	**53.5**	53.6
QUESREV-**RC**	53.2	53.8
QUESREV-**AC** + **RC**	53.3	**53.9**

Table 3.4: $\mathbf{F}_1$ performance of variants of our model QUESREV and alternative solutions on base QA system STAGG.

sparsity. This intuition is further justified by observing that augmenting the training data with SIMPLEQUESTIONS improves the performance of **RC** model most as it has more expressive power.

Although both QA-LSTM and ATTENTIVE-LSTM lead to successful prediction refinements on STAGG, question revision approach consistently outperforms both of the alternative solutions. This suggests that our way of incorporating the new textual evidences by naturally blending them in the question context leads to a better mechanism for checking the consistency of KB relations with the question. It is possible to argue that part of the improvements of refinement models over STAGG in Table 3.4 may be due to model ensembling. However, the performance gap between QUESREV and the alternative solutions enables us to isolate this effect for query revision approach.

In Table 3.5, we present qualitative results including example improvements made possible by the proposed QUESREV approach. The proposed post-inspection is able to identify inconsistencies such as profession/fight for, place of birth/executed, river mouth/start. These example improvements further verify our initial motivation for studying question revision approach, which is designed to fix wrong KB relation predictions

Example Predictions and Replacements
1. What position did **vince lombardi** play in college ? **Stagg**: person.education / education.institution (2-hop) - what position did person play in college **QuesRev-EC**: football_player.position_s - what position did american football player play in college
2. What did **mary wollstonecraft** fight for ? **Stagg**: person.profession - what profession did person fight for **QuesRev-AC**: activist.area_of_activism - what activism issue did activist fight for
3. Where was **anne boleyn** executed ? **Stagg**: person.place_of_birth - where place of birth was person executed **QuesRev-RC**: deceased_person.place_of_death - where place of death was deceased person executed
4. Where does the **zambezi river** start ? **Stagg**: river.mouth - where mouth does the river start **QuesRev-RC**: river.origin - where origin does the river start

Table 3.5: Example predictions of STAGG [3] and replacements proposed by variants of QUESREV, followed by their corresponding question revisions. The colors *red* and *blue* indicate wrong and correct, respectively. Domain names of KB relations are dropped for brevity.

through deeper clues obtained by inserting them back in question context.

3.6 Related Work

One of the promising approaches for KB-QA is semantic parsing, which uses logical language CCG [18, 42, 43] or DCS [17] for finding the right grounding of the natural language on knowledge base. Another major line of work [19, 3, 40] exploit vector space embedding approach to directly measure the semantic similarity between questions and candidate answer subgraphs in KB. In this work, we propose a post-inspection step that

can help existing KB-QA systems recover from answer prediction errors.

Our work is conceptually related to traditional query expansion, a well-explored technique [54, 55, 56, 57, 58, 59, 60] in information retrieval area. The intuition behind query expansion is to reformulate the original query to improve retrieval performance. Our approach revises questions using candidate answers already retrieved by a base QA system. Revised questions are then used for reasoning about the corresponding predictions themselves, not for retrieving more candidates. Hence, it is specialized rather as a reasoning component than a retrieval one.

Hypothesis generation steps in [61] and [62] are related to our question revision process. However, hypotheses in these approaches need to be further compared against supporting paragraphs for reasoning. This limits the applicability of them in KB-QA setting due to lack of supporting texts. Our approach modifies the appropriate parts of the question using different KB evidences behind candidate answers that are more informative and generalizable. This enables us to make reasoning about candidate predictions directly via revised questions without relying on any supporting texts.

3.7 Conclusion

We present a prediction refinement approach for question answering over knowledge bases. We introduce question revision as a tailored augmentation of the question via various textual evidences from KB relations. We exploit revised questions as a way to re-examine the consistency of candidate KB relations with the question itself. We show that our method improves the quality of answers produced by STAGG on the WEBQUESTIONS dataset.

Chapter 4

What It Takes to Achieve 100% Condition Accuracy on WikiSQL

4.1 Introduction

A large amount of world's data is stored in relational databases, which require users to master structured query languages such as SQL to query data. It might not be convenient for many users who do not have programming background. Towards removing this huge barrier between non-expert users and data, building reliable natural language interfaces to databases has been a long-standing open problem [63, 64, 65].

Recently, there is a renewed interest in natural language interfaces to databases due to the advance in deep learning and the new release of large-scale annotated data such as WikiSQL [66]. WikiSQL includes a large collection of questions and their corresponding SQL queries. While the queries in WikiSQL are quite simple: Each involves one relation and does not have join operations, none of the publicly reported SQL generation models [66, 67] can achieve an accuracy beyond 62%, which is far from enough for practical use. It is not clear yet what level of parsing capabilities are needed to achieve high

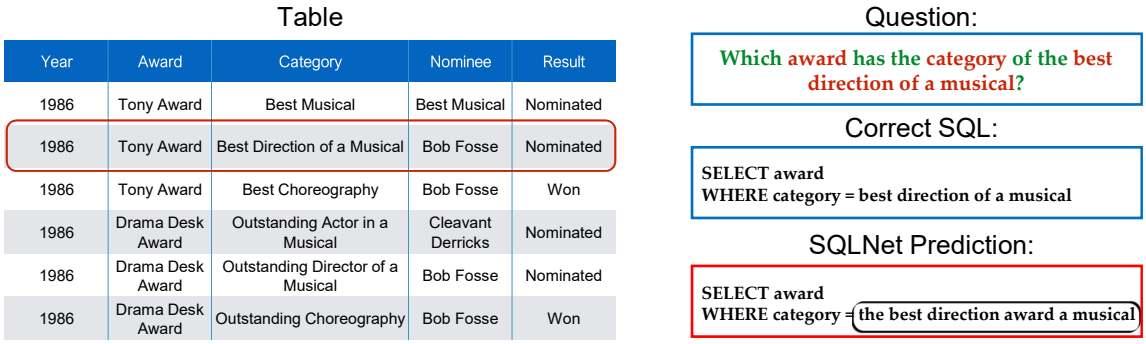

Figure 4.1: An example from WikiSQL. SQLNet makes a wrong prediction on the condition value of the `WHERE` clause.

performance, ideally close to 100% accuracy, on this task.

In this chapter, we aim to figure out the level of language understanding required to perform well on the WikiSQL task. Specifically, we focus on the `WHERE` clause generation, which is the most challenging part of this task as reported in [67]: The accuracies for other clauses like `SELECT` and `AGGREGATE` are over 90% whereas the accuracy for `WHERE` is only around 70%. We aim to conduct the following two studies: (i) Understanding the difficulties of the task and (ii) Investigating alternative solutions to realize the potential ceiling performance.

To this end, we first conduct a careful analysis on a subset of the WikiSQL data to identify the main challenges. This analysis leads to two important observations: (i) ∼17% of the questions are either too ambiguous (hard) or require external knowledge to answer, (ii) ∼68% of the questions can be answered by exact match or simple paraphrasing, however we surprisingly find that the current best system [67] can only get less than 80% accuracy on such simple questions. Deeper analysis on the second observation leads to the conclusion that most of the errors in this category are due to wrongly generated condition values as shown by the example in Figure 4.1. One can resort to soft/hard look-up based approaches over table content as in previous work [68, 69] or user interaction (which is also applicable to the first observation) to more accurately recognize the entities

in questions for better condition value generation.

As a second contribution, we propose to use table content as additional data source to address the aforementioned wrongly generated condition value problem, and investigate solutions to realize the potential performance limit (upper bound) on WikiSQL. Note that neither Seq2SQL [66] nor SQLNet [67] utilize table content. However, we show that it is not straightforward to achieve a high accuracy even in the scenario where table content is available as an optimal external knowledge through our model ablation results and error analysis. We demonstrate that our proposed solutions can reach up to 88.6% `WHERE` condition accuracy, almost matching the performance on `SELECT` and `AGGREGATE`.

4.2 Background

The WikiSQL dataset introduced in [66] is created from a large number of tables extracted from Wikipedia, employing Amazon Mechanical Turk for annotation. An example from the dataset is provided in Figure 4.1: It consists of a table t, a SQL query s, and its corresponding natural language question q. There can be multiple conditions in the `WHERE` clause of a SQL query, each of which consists of a table column, an operator ($=, <, >$, etc.), and a condition value.

Instead of asking annotators to write SQL queries for given questions and tables, the authors [66] facilitate the annotation process by paraphrasing generated questions. This raises concerns that the resulting dataset is limited to only simple queries. We acknowledge this concern. However, it is still a large, valuable dataset towards the goal of building ultimate natural language interfaces to databases. If the existing or newly proposed solutions can not solve this task with high accuracy, how can we advance to more complicated ones? Any insight and solution to this task can help us build more advanced SQL synthesis algorithms in future.

Category	Question	Pseudo SQL Query	Columns
EXACT MATCH	In what state was the **electorate** fowler?	SELECT state WHERE **electorate** EQL fowler	member, party, state, **electorate**, term in office
PARAPHRASE	What was the date of the game after week 5 **against the Houston Oilers?**	SELECT date WHERE week GT 5 AND **opponent EQL houston oilers**	week, date, **opponent**, result, attendance
PARTIAL CLUE	Who had the most points in the game **on March 7**?	SELECT high points WHERE **date EQL march 7**	high points, game, **date**, team, score, ...
EXTERNAL-KNOWLEDGE	Name the callback date **for amway arena**	SELECT callback date WHERE **audition venue EQL amway arena**	**audition venue**, audition city, callback date, ...
AMBIGUOUS	List the branding **for krca-tv**	SELECT branding WHERE **callsign EQL krca-tv**	branding, power (kw), **callsign**, channel, ...

Table 4.1: Representative examples for each category. **EQL** and **GT** correspond to the = and > operators. Columns include the corresponding table's column names. Highlighted parts of the question, pseudo SQL query, and columns are provided to indicate clues for the category.

4.3 WikiSQL Data Analysis

In this section, we aim to deliver a thorough analysis of the WikiSQL dataset. [66, 67] made an assumption that only table schema is available to the model, and table content (i.e., table cell values) is not available. We want to answer the following question: As the dataset creation process involves several heuristics and pre-defined templates to simplify the annotation task, what kind of capabilities does it still require to answer the resulting questions successfully? To this end, we randomly sample 100 examples from the development set of WikiSQL for analysis.

Category	# Questions	Percentage
Exact match	59	54.1%
Paraphrase	16	14.7%
Partial clue	15	13.8%
External knowledge	11	10.1%
Ambiguous	8	7.3%

Table 4.2: Number of examples in different categories.

4.3.1 Categorization

We manually categorize these 100 examples in terms of the capability needed to predict the correct `WHERE` clause as described below. We also provide illustrative examples for each category in Table 4.1. If an example belongs to multiple categories, we include it in all the categories that apply. In Table 4.2, we present the break down of the examples over these categories.

Exact match. For each condition in the SQL query, its column name appears in the question near the neighborhood of its condition value with exactly the same surface form.

Paraphrase. For at least one of the conditions in the SQL query, its column name is paraphrased in the question. So, the inference requires certain paraphrasing capabilities.

Partial clue. For at least one of the conditions in the SQL query, its column name is not explicitly mentioned in the question, not even in paraphrased form. However, there are still partial semantic clues for inference.

External knowledge. For at least one of the conditions in the SQL query, there is no clue in the question to infer its column name. Inferring this column name from the question requires external knowledge regarding the type of its condition value that can be detected from the question.

Ambiguous. For this category, even with the external knowledge it is almost impossible (even for humans) to confidently infer the correct condition column from the question.

Category	Seq2set+CA	Seq2set+CA+WE
Exact match	67.8%	72.9%
Paraphrase	75.0%	68.8%
Partial clue	80.0%	80.0%
External knowledge	54.6%	45.5%
Ambiguous	37.5%	37.5%
Total	67.0%	67.9%

Table 4.3: Accuracy breakdown of SQLNet.

4.3.2 Performance Breakdown of SQLNet

In Table 4.3, we show the performance breakdown of SQLNet [67], the state-of-the-art model for WikiSQL, on the selected 100 examples. It has two variants with comparable performance, so we show both of them. As expected, both variants of SQLNet perform poorly on examples that are either too ambiguous or require external knowledge. However, it is surprising that the accuracy on examples that need only exact matching or simple paraphrasing is also not very high, especially considering the paraphrasing capabilities of deep learning models gained from distributed representations. We find that most of these errors are due to wrongly generated condition values as illustrated in Figure 4.1, where SQLNet fails to even produce a valid phrase. This is indeed important prior knowledge that can be effectively outsourced by resorting to soft/hard look-up based approaches [68, 69] instead of fully relying on models to precisely generate it.

The above observations exhibit an opportunity to incorporate external knowledge in the condition generation process. We opt to use the table content as the knowledge source to address the wrong condition value problem, which is not used in the existing models [66, 67]. Next we will show that it is not trivial to leverage table content.

4.4 Our Solutions

As motivated in the previous sections, our main objective here is to investigate solutions to realize the potential ceiling performance on WikiSQL when using the table content as an additional knowledge source. We first describe an attentional RNN-based model that serves as our baseline, and propose several variants for `WHERE` clause generation, each addressing a specific weakness.

4.4.1 Task Formulation

Given a question q and a table t, our objective is to generate the `WHERE` clause of the SQL query corresponding to the question. In addition, we assume that the table t can be queried while generating the `WHERE` clause. Each condition in the `WHERE` clause is represented as a triple of $(\texttt{COLUMN}, \texttt{OP}, \texttt{VALUE})$. SQLNet [67] generates each of these individual components by first predicting `COLUMN` and `OP`, and then generating `VALUE` using pointer networks [70]. In contrast, we propose a two step approach to tackle this problem in the reverse way, taking advantage of the content of table t as additional knowledge: (i) Generate the condition `VALUE` from the question, and (ii) predict which `COLUMN` and `OP` apply to this `VALUE`.

Candidate Generation

Our objective in candidate generation process is to produce a set of $(\texttt{COLUMN}, \texttt{VALUE})$ pairs from question q and table t, where `VALUE` is an *n-gram* in q and `COLUMN` is a column in t. Similar to prior work [66, 67], we assume that `VALUE` occurs in the question.

As illustrated in Figure 4.2, we first generate the set $\mathcal{N}$ of all *n-grams* from the question. Then, for each candidate $v \in \mathcal{N}$ and each column $c \in t$, we query table t to check whether value v is contained in any row of column c. Then, we create a set

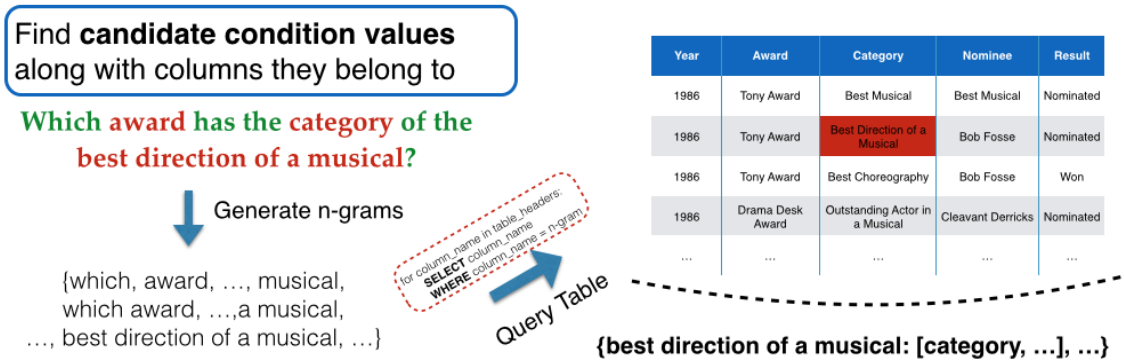

Figure 4.2: Candidate generation.

$\mathcal{C} = \{(c, v) : v \in c\}$ of (`COLUMN`, `VALUE`) pairs as our final candidates for the `WHERE` clause. We note here that `VALUE` may not necessarily be an *n-gram* in question for other potential external knowledge or NLIDB tasks. In such scenarios, we can alternatively resort to soft look-up based approaches like the ones proposed in [68, 30].

Condition Prediction

Given a question q, SQL table t, and a set $\mathcal{C}$ of (`COLUMN`, `VALUE`) pair as candidates for `WHERE` clause, our objective is to learn two mappings:

$$f_{\text{cond}} : (q, c, v) \mapsto \{0, 1\} \tag{4.1}$$

$$f_{\text{op}} : (q, c, v) \mapsto \{=, >, <\} \tag{4.2}$$

where f_{cond} determines whether to select $(c, v) \in \mathcal{C}$ as a condition in `WHERE` clause and f_{op} predicts the operator `OP` $\in \{=, >, <\}$. These two mappings together can fully determine the final `WHERE` clause. Note that there can be multiple conditions in the `WHERE` clause.

4.4.2 Model Overview

We design a neural network model (Figure 4.3) to instantiate the mappings f_{cond} and f_{op}. We first describe the general structure of the model, consisting of the following steps:

Value Context. We define *value context* as the context of a value v, a span in a longer utterance q. Later the question encoder will condition on this information to make predictions. We will investigate different options for value context in Section 4.4.3. For now, we define it as a transformation g_{context} which maps each (q, v) to its value context.

Value Abstraction. Inspired by [53], we define *value abstraction* as a transformation g_{abstract} that replaces the surface form of `VALUE` in the question with a single token `ENTITY`. For the running example in Figure 4.1, applying value abstraction maps the question to "*Which award has the category of the ENTITY?*" It further informs the question encoder regarding the location of the `VALUE` in the question.

Encoding. Given question $q = (q_1, q_2, \ldots, q_m)$, column $c = (c_1, c_2, \ldots, c_n)$, and value $v = (v_1, v_2, \ldots, v_k)$. Before encoding, we first apply the aforementioned textual transformations and obtain value context

$$q' = (q'_1, q'_2, \ldots, q'_l) = g_{\text{context}} \circ g_{\text{abstract}}(q, v).$$

We first encode all the words into a vector space using an embedding matrix $E \in \mathbb{R}^{d \times |\mathcal{V}|}$, where $\mathcal{V}$ denotes the vocabulary and d is the embedding dimension. Let $\mathbf{q'}_i$ denote the embedding of word q'_i. To obtain the contextual embeddings, we use a bi-directional LSTM with hidden unit size h:

$$\overrightarrow{\mathbf{h}^{\text{q}}}_i = LSTM^{\text{q}}_{\text{fwd}}(\overrightarrow{\mathbf{h}^{\text{q}}}_{i-1}, \mathbf{q'}_i) \tag{4.3}$$

$$\overleftarrow{\mathbf{h}^{\text{q}}}_i = LSTM^{\text{q}}_{\text{bwd}}(\overleftarrow{\mathbf{h}^{\text{q}}}_{i+1}, \mathbf{q'}_i) \tag{4.4}$$

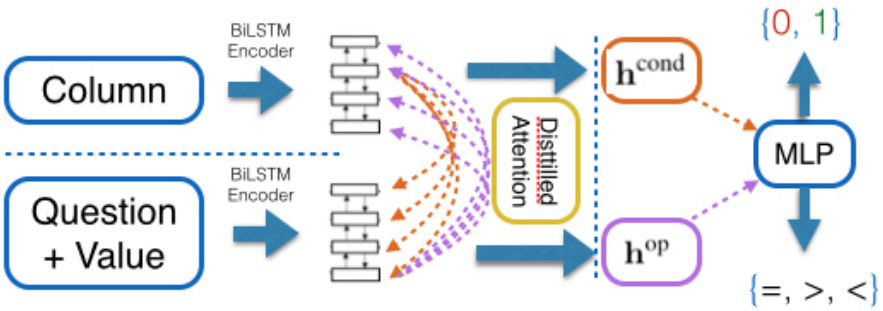

Figure 4.3: Model overview.

with $\overrightarrow{\mathbf{h}}_0 = \mathbf{0}$ and $\overleftarrow{\mathbf{h}}_{l+1} = \mathbf{0}$. We combine forward and backward contextual embeddings for each word in the question to obtain $\mathbf{h}^{\text{q}}_i = \mathbf{concat}(\overrightarrow{\mathbf{h}^{\text{q}}}_i, \overleftarrow{\mathbf{h}^{\text{q}}}_i) \in \mathbb{R}^{2h}$.

Similarly, we obtain the contextual embedding $\mathbf{h}^{\text{col}}_j \in \mathbb{R}^{2h}$ for each word c_j in column c with another bi-directional LSTM but shared word embedding matrix.

Distilled Attention. The objective of this step is to distill the most relevant information from both the value context and the column for the final prediction. We first compute the attention score of each word $c_j \in c$ on the words $q'_i \in q'$:

$$S^{(\text{col}\to\text{q})}_{j,i} = \mathbf{h}^{\text{col}}_j \mathbf{W}^{(\text{col}\to\text{q})} \mathbf{h}^{\text{q}}_i \tag{4.5}$$

$$P^{(\text{col}\to\text{q})}_j = \text{softmax}_i \; S^{(\text{col}\to\text{q})}_{j,i} \tag{4.6}$$

where $\mathbf{W}^{(\text{col}\to\text{q})} \in \mathbb{R}^{2h\times 2h}$ is a model parameter to be learned and $P^{(\text{col}\to\text{q})}_j \in \mathbb{R}^l$ represents an attention probability distribution of word c_j over the words of value context q'. Let $P^{(\text{col}\to\text{q})} \in \mathbb{R}^{n\times l}$ denote the column-wise concatenation of $P^{(\text{col}\to\text{q})}_j$ indicating the unified representation of attention matrix from column words onto value context. Similarly, we compute the attention from value context (question) to column and get $P^{(\text{q}\to\text{col})} \in \mathbb{R}^{l\times n}$.

The intuition behind *distilled attention* is to allow two-way comparison to clean up the attention weights. To exemplify this point better, consider a scenario where a word $c_j \in c$ attends on a word $q'_i \in q'$ with high probability, but q'_i is much more relevant

to another word $c_{j'} \in c$. We leverage reverse attention $P^{(\text{q}\to\text{col})}$ to distill the attention weights of column words on the value context. To this end, we define distilled attention weights as

$$P = (P^{(\text{q}\to\text{col})})^\top \odot P^{(\text{col}\to\text{q})} \tag{4.7}$$

where $P \in \mathbb{R}^{n\times l}$ becomes our final attention weights and $\odot$ is the Hadamard product.

Value Context and Column Representations. Having defined how the encodings and attention weights are computed, we now describe the final value context and column representations. First, we compute a value context vector for each word $c_j \in c$ using the distilled attention weights by

$$\mathbf{h}_j^{(\text{q}\to\text{col})} = \sum_{i=1}^{l} P_{j,i}\ \mathbf{h}_i^{\text{q}} \tag{4.8}$$

and fuse it with the corresponding column context vector by

$$\mathbf{h}_j^{\text{cond}} = \tanh(\mathbf{W}_0^{\text{cond}}\mathbf{h}_j^{\text{col}} + \mathbf{W}_1^{\text{cond}}\mathbf{h}_j^{(\text{q}\to\text{col})}) \tag{4.9}$$

$$\mathbf{h}_j^{\text{op}} = \tanh(\mathbf{W}_0^{\text{op}}\mathbf{h}_j^{\text{col}} + \mathbf{W}_1^{\text{op}}\mathbf{h}_j^{(\text{q}\to\text{col})}) \tag{4.10}$$

where $\mathbf{W}_0^{\text{cond}}, \mathbf{W}_1^{\text{cond}}, \mathbf{W}_0^{\text{op}}, \mathbf{W}_1^{\text{op}} \in \mathbb{R}^{2h\times 2h}$ are trainable model parameters. Finally, we apply a pooling layer to obtain fixed-size representations

$$\mathbf{h}^{\text{cond}} = \frac{1}{n}\sum_j \mathbf{h}_j^{\text{cond}} \text{ and } \mathbf{h}^{\text{op}} = \frac{1}{n}\sum_j \mathbf{h}_j^{\text{op}} \tag{4.11}$$

for *condition* and *operator* predictions.

Prediction. So far we have obtained two different representations $\mathbf{h}^{\text{cond}} \in \mathbb{R}^{2h}$ and $\mathbf{h}^{\text{op}} \in \mathbb{R}^{2h}$ as the latent unified representations of question, column, and value triple

(q, c, v). We than use these representations to make the final predictions:

$$\mathbf{p}^{\text{cond}} = \text{softmax}(\mathbf{U}^{\text{cond}}\ \mathbf{h}^{\text{cond}})\ \in \mathbb{R}^2 \tag{4.12}$$

$$\mathbf{p}^{\text{op}} = \text{softmax}(\mathbf{U}^{\text{op}}\ \mathbf{h}^{\text{op}})\ \in \mathbb{R}^3 \tag{4.13}$$

where $\mathbf{U}^{\text{cond}} \in \mathbb{R}^{2\times 2h}$ and $\mathbf{U}^{\text{op}} \in \mathbb{R}^{3\times 2h}$ are model parameters. The final prediction mappings are then defined as:

$$f_{\text{cond}}(q, c, v) =_i \mathbf{p}_i^{\text{cond}} \tag{4.14}$$

$$f_{\text{op}}(q, c, v) =_i \mathbf{p}_i^{\text{op}} \tag{4.15}$$

Training Objective. Let $\mathcal{T} = \{(q, c, v, l, o)\}$ denote a set of training tuples where $l \in \{0, 1\}$ denotes whether to include a condition with (c, v) as (`COLUMN`, `VALUE`) pair and if so, $o \in \{=, >, <\}$ indicates which `OP` to apply. Our loss function consists of two components:

$$J(\Theta) = \ J_{\text{cond}}(\Theta) + l\ J_{\text{op}}(\Theta) \tag{4.16}$$

where $J_{\text{cond}}(\Theta) = -\log(\mathbf{p}_l^{\text{cond}})$ and $J_{\text{op}}(\Theta) = -\log(\mathbf{p}_o^{\text{op}})$ are simply negative log-likelihood losses for *condition* and *operator* predictions.

Inference. We also employ an inference schema where we make a simple assumption that a condition value v may be a part of *only one* condition. Hence, we group candidate $\{(c, v)\}$ pairs by value v, and create a candidate column set C_v: $\{c\colon (c, v)\}$ for each unique candidate value v. Based on the probabilities $p^{\text{cond}}(c, v)$ computed by the trained model, we select the column $c \in C_v$ with the maximum probability for each candidate value v, hence form the set of (`COLUMN`, `VALUE`) pairs to be included in condition this way.

4.4.3 Model Variants

In this section, we discuss variants of our model with different choices for the value context. Consider the running example in Figure 4.1 and assume hypothetically that *award* is a candidate condition column for value *best direction of a musical*. When we use the whole question as the value context, eliminating *award* from being a condition `COLUMN` for this `VALUE` becomes very challenging for the model as it is not informed enough regarding the finer context of the value.

Base Model:

The base model simply uses identity mapping for value context. More precisely, we use the whole question as the context for the candidate condition `VALUE`. So, $g_{\text{context}}(q, v) = q$.

Window-based model:

The objective of *window-based* model is to get value contexts that can provide more clean context information by leveraging the context window around the candidate value. In this case, we first identify the span $[start, end]$ for value v in the question q. Based on its span and a predetermined context window size w, we define

$$g_{\text{context}}(q, v) = (q_{start-w}, \ldots, q_{start}, \ldots, q_{end}, \ldots, q_{end+w})$$

Parse tree-based model:

We also analyze a simple parse tree based model that hierarchically split up the value context into multiple contexts based on the constituency parse tree of the question. To motivate this, consider the question in Figure 4.4. Window-based value context for 0.5 is *"a large end smaller than ENTITY"* and candidate table columns to apply this value are

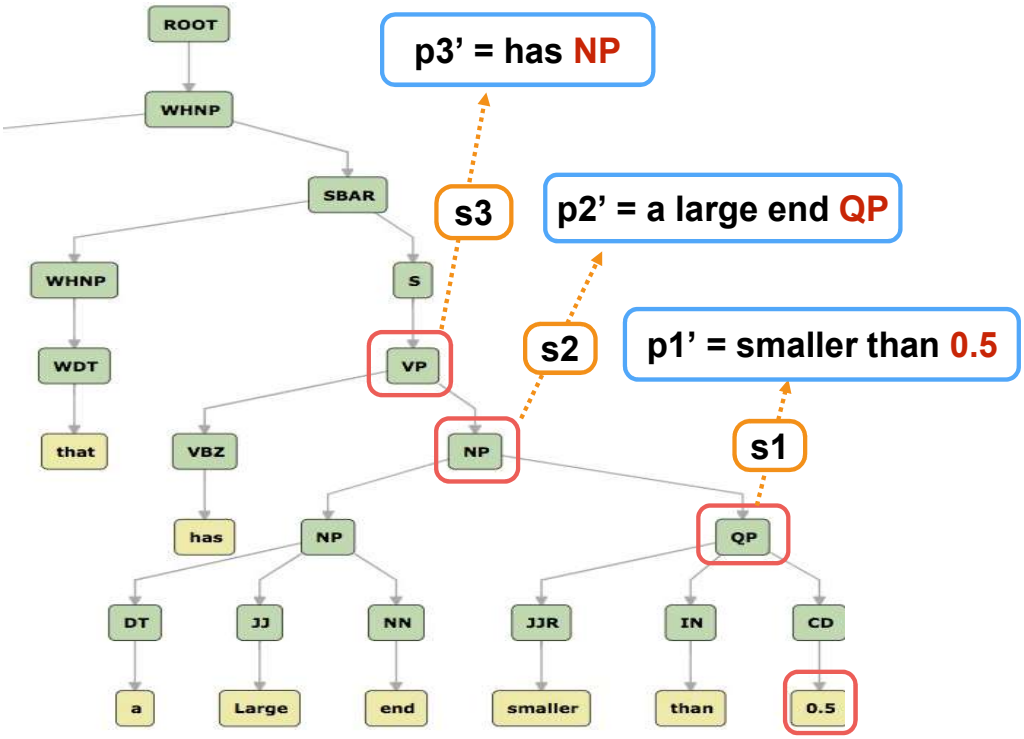

Figure 4.4: Partial view of parse tree for question "*Which taper/ft that has a Large end smaller than 0.5*" illustrating parse tree based value contexts.

"large end" and *"small end"*. Based on this context, a model will likely assign a higher probability to *"large end"* than *"small end"*. The syntactic structure of the question can potentially help reduce such ambiguities.

Parse tree-based value contexts. As highlighted in Figure 4.4, we use nested phrase-level subtrees that contain the candidate `VALUE`. Moreover, to better inform the model regarding the type of phrases, we use phrase level constituent tags of subtrees in two ways: (i) to inform the parent tree with the type of its subtree containing candidate `VALUE`, (ii) to apply a tag-specific affine transformation on phrase representations. To this end, we first select the r-lowest subtrees $s_1 \subset s_2 \subset \ldots \subset s_r$ of the question's parse tree containing `VALUE` along with their corresponding phrase-level constituency tags[1]

[1]We use the Penn treebank annotation conventions that are described in bies95penntreebank at http://languagelog.ldc.upenn.edu/myl/PennTreebank1995.pdf

$t_1, t_2, \ldots, t_r$, respectively. We then iteratively form multiple values contexts as nested phrases of $p'_1, p'_2, \ldots, p'_r$ shown in Figure 4.4 as follows: (i) p'_1 is equal to the phrase formed by the words at the leaf nodes of subtree s_1, and then (ii) iteratively form p'_i as the phrase formed by the words under subtree s_i by replacing its subphrase corresponding to subtree s_{i-1} with its constituency tag t_{i-1} for $i > 1$.

Modifications to General Model. We now describe how the general model defined in Section 4.4.2 is adapted to accommodate the multiple nested value contexts of different types. In this process, we aim to capture two types of important information: (i) syntactic phrase-level types of value contexts, and (ii) their distance to `VALUE`.

We first compute a value context vector $\mathbf{h}_{j,k}^{(\text{q}\to\text{col})} \in \mathbb{R}^{2h}$ for each word $c_j \in c$ and each value context p'_k as in Eq. 4.8 using the same encoding and attention mechanisms. However, we replace the affine transformation layers defined in Eq. 4.9 and 4.10 with tag and distance specific ones as follows:

$$\mathbf{h}_j^{\text{cond}} = \tanh(\mathbf{W}_0^{\text{cond}}\mathbf{h}_j^{\text{col}} + \frac{1}{r}\sum_{k=1}^{r}\mathbf{W}_{k,t_k}^{\text{cond}}\mathbf{h}_{j,k}^{(\text{q}\to\text{col})}) \tag{4.17}$$

$$\mathbf{h}_j^{\text{op}} = \tanh(\mathbf{W}_0^{\text{op}}\mathbf{h}_j^{\text{col}} + \frac{1}{r}\sum_{k=1}^{r}\mathbf{W}_{k,t_k}^{\text{op}}\mathbf{h}_{j,k}^{(\text{q}\to\text{col})}) \tag{4.18}$$

where $\mathbf{W}_0^{\text{cond}}, \mathbf{W}_{k,t_k}^{\text{cond}}, \mathbf{W}_0^{\text{op}}, \mathbf{W}_{k,t_k}^{\text{op}} \in \mathbb{R}^{d\times 2h}$ for $k = 1, 2, \ldots, r$ are model parameters. It is important to note here that k determines the distance of value context to `VALUE` and t_k indicates its tag, effectively making the fusion layers above *tag* and *distance* specific. The rest of the model exactly follows the Eq. from 4.11 through 4.15 with the same training objective and inference schemes.

4.5 Experiments

In this section, we discuss the details of the experiments and present our main findings.

4.5.1 Training Details

For training our neural networks, we only keep the words appearing at least 3 times in the whole training data and the rest of the words are replaced with `UNK` token. Word embeddings are initialized with pretrained GloVe [31] vectors[2], and updated during the training. We take the dimension of word embeddings and the size of LSTM hidden layer both equal to 100. The model parameters are optimized using Adam [50] with batch size of 32 and a decaying learning rate starting with 0.001. We apply gradient clipping to 5 when its norm exceeds this value. We use early stopping based on the model accuracy on the development set. We report our results with a model snapshot achieving the best accuracy on the development set. Our models are implemented in *tensorflow* [51].

For candidate generation, we use *n-gram* size of 10. For window-based model, we experiment with window size of $w \in \{2, 3, 4, 5, 6\}$ and report results with $w = 4$ that performs best on the development set. We use Stanford CoreNLP tool [29] for tokenization and parse-tree generation.

4.5.2 Main Results

In Table 4.4, we present our main results in comparison with the related works. Baseline refers to a baseline for our models where the `WHERE` clause accuracy is computed by assuming that each candidate (`COLUMN`, `VALUE`) pair is included with corresponding `OP` being equality. On the other hand, UpperBound accuracy is computed by assuming f_{cond} and f_{op} makes 100% correct mapping of whether to include a candidate (`COLUMN`, `VALUE`) pair and which `OP` to apply on this condition. In other words, errors in UpperBound exist due to wrong candidate generation.

As shown in Table 4.4, our models surpass the previous results by a large margin as

[2]More specifically, we use 100D vectors from http://nlp.stanford.edu/data/glove.6B.zip

Model	Dev	Test
No External Knowledge		
Seq2Sql [66]	62.1%	60.2%
SqlNet-(Seq2Set + CA)	72.1%	70.0%
SqlNet-(Seq2Set + CA + WE)	74.1%	71.9%
SqlNet*-(Seq2Set + CA)	72.3%	70.9%
SqlNet*-(Seq2Set + CA + WE)	73.8%	71.7%
In Our Scenario		
SqlNet*-(Seq2Set + CA)	82.2%	80.9%
SqlNet*-(Seq2Set + CA + WE)	**83.5**%	**81.8**%
Stamp + RL [71]	77.3%	76.3%
TypeSql + TC (w/ Freebase) [72]	**92.8**%	**87.9**%
Our Baseline	**77.2**%	**77.1**%
SqlMaster	84.8%	83.9%
SqlMaster + VA	86.2%	86.1%
SqlMaster (Window-Based)	87.4%	87.1%
SqlMaster (Window-Based) + VA	87.9%	87.6%
SqlMaster (Tree-Based)	88.7%	88.3%
SqlMaster (Tree-Based) + VA	**88.9**%	**88.6**%
Our UpperBound	**92.1**%	**91.4**%

Table 4.4: `WHERE` clause accuracy. +VA denotes that *Value Abstraction* is applied. SqlNet* refers to our reimplementation of SQLNet. SqlMaster refers to our proposed models. Only TypeSql + TC leverages Freebase on top of table content among the models reported in our scenario. WHERE clause accuracy of TypeSql (w/o Freebase) is not reported.

well as its variants improving upon each other. A portion of these improvements definitely come from assuming and using the table itself as the optimal external knowledge. Acknowledging this fact, we make the following more important conclusions from Table 4.4, 4.5, and 4.6: (i) Comparison of the performance results across scenarios (SqlNet vs. SqlMaster or TypeSQL) reveals that there is a large room for improvement when an external knowledge base is used, (ii) Comparison of our own models with its variants demonstrates that each component/extension incorporated brings a considerable performance improvement, justifying its potential power to be used in other related NLP tasks,

Model	Dev	Test
SqlNet [67]	63.2%	61.3%
DialSql [73]	70.9%	69.0%
TypeSql (w/o Freebase) [72]	66.5%	64.9%
TypeSql + TC (w/ Freebase) [72]	**79.2**%	**75.4**%
SqlMaster (Ours, w/o Freebase)	**73.1**%	**72.4**%

Table 4.5: Full query-match (QM) accuracy. For SqlMaster, we combine our WHERE clause predictions with the SELECT and AGGREGATE clause predictions of SQLNet. QM result for TypeSql + TC (w/o Freebase) is not reported.

Category	SqlNet	SqlMaster
Exact match	72.9%	86.4%
Paraphrase	68.8%	93.8%
Partial clue	80.0%	93.3%
External knowledge	45.5%	90.9%
Ambiguous	37.5%	75.0%
Total	67.9%	88.1%

Table 4.6: Accuracy breakdown of SqlNet compared to SqlMaster over the categories obtained by hand-analysis on the randomly selected examples as explained in Section 4.3.

(iii) Comparing our models with SqlNet and TypeSql+TC within our scenario provides further clues/justifications towards effectiveness of our proposed Tree-Based model, (iv) SqlMaster performs comparably to TypeSql + TC [72] on WHERE condition predictions despite the fact that we do not use Freebase, which is exploited in [72] to identify named entities of certain Freebase types (e.g., *person*, *place*, *country*, *organization*, *sport*), (v) Our SqlMaster (Tree-Based) + VA model achieves 88.6% on test portion of WikiSQL, which almost reaches the upper bound of 91.4%, demonstrating a great promise for future work in this domain, and finally (vi) When the performance of our model is compared to SqlNet over the categories as shown in Table 4.6, we observe that it consistently improves the performance of over all the categories, but most noticeably on External Knowledge and Ambiguous ones which were the main categories inspiring this study and proposed approaches motivated by the analysis in Section 4.3.

Evaluation in Our Scenario. We adapt SQLNet [67] results to our scenario via a post-processing step as follows: For each of their predicted condition in `WHERE` clause, if column c and operator o are both correct, but value v is wrong, then we replace this value with the one in our generated candidates that maps to the column c when the mapping is unique (only one value maps to c).

4.5.3 Error Analysis

In this section, we provide an error analysis of our models to better understand what are the remaining challenges to achieve UpperBound performance. To this end, we analyzed 100 randomly sampled examples from development set on which our best model fails. 41% of these errors are due to our models not being able to perform a good semantic understanding of the value context. 36% of the errors correspond to ambiguous questions that lack sufficient information to disambiguate between correct and wrong column. A good representative example for this category is "*Who was the director of king's speech?*", where our model predicts "*winner and nominees*" column for the condition value "*king's speech*" while the correct column is "*original title*". The remaining 18% and 5% of the errors are caused by sparsity of column names and wrong labelling problems, respectively.

4.6 Related Work

Research on natural language interfaces to databases (NLIDBs) and semantic parsing has spanned several decades. Early rule-based NLIDBs [63, 64, 65] employ carefully designed rules to map natural language questions to formal representations like SQL queries. While having a high precision, rule-based systems are brittle when facing with language variations and usually only admit inputs from a restricted vocabulary. The rise of statistical models [74, 75, 17] and neural network models [3, 76, 77, 66, 67] has

enabled NLIDBs that are more robust to language variations. Such systems allow users to formulate questions with greater flexibility instead of having to probe and adapt to the boundary of rule-based systems.

Along with the advance in modeling is the development of benchmarks for training and testing NLIDB models. Early benchmarks are mostly curated by experts [78, 79]. State-of-the-art models [76] have achieved a high accuracy of 80% to 90% on these benchmarks. In the recent years, a number of large-scale, crowdsourced benchmarks have been constructed with the goal to train and test NLIDBs in a more real-world setting, notably WebQuestions [17] and GraphQuestions [80] for knowledge bases, and WikiSQL [66] for SQL queries to relational databases. The best accuracies on these benchmarks are still far from enough for real use, typically in the range of 20% to 60%.

Besides releasing WikiSQL dataset, the authors of [66] propose an approach (Seq2SQL) to solve this task. Seq2SQL leverages the pointer-networks [70] to generate linearized SQL queries token-by-token using the input question and table schema. They report significant performance improvement over [76], a generic sequence-to-tree approach proposed for semantic parsing. More recently, [67] proposes a sketch-based sequence-to-set approach (SQLNet) eliminating sequence-to-sequence structure employed in [66], when the order does not matter. In our work, we provide a careful analysis of SQLNet results to better understand the limitations of this model on the WikiSQL task. Inspired by this analysis, we propose novel solutions to realize close to upper-bound condition accuracy in the scenario where SQL table is available as an optimal external knowledge. Another recent work [72] also focuses on using external knowledge (Freebase) along with the table content to generate SQL queries in a type aware fashion. A concurrent line of related work exploits graph-to-sequence neural models with the aim to better exploit syntactic information in the input question [81, 82]. On the other hand, [73] takes an orthogonal approach and introduces a dialogue-based query refinement mechanism where a candi-

date SQL query (generated by any black-box model) is refined by interactively validating and updating modular segments of the query with users. The authors show that by having successful interactions with users, not only the accuracy of the candidate queries can be improved but also new insights into limitations of current query generation systems can be gained.

There are also a number of recent studies on semantic parsing for semi-structured tables. For example, Pasupat and Liang [83] develop the WikiTableQuestions benchmark, where the task is to find table cells in HTML table to answer questions, while Jauhar et al. [84] focuses on multi-choice questions. On the other hand, Sun et al. [77] study how to answer user questions with table cells from millions of HTML tables. These studies directly find cells of semi-structured tables as answers, instead of generating SQL queries for relational databases.

4.7 Conclusion

In this chapter, we thoroughly analyzed the recently released WikiSQL dataset and the performance breakdown of SQLNet. Through the analysis, we identified an opportunity/need to further explore the potentials of incorporating external knowledge in the structured query generation process. In this direction, we developed alternative solutions to explore the potential performance limits for this task in the scenario where table content can be used. We showed that our proposed systems can reach up to 88.6% accuracy in condition generation and provided a discussion regarding what the remaining challenges were through an error analysis. We consider solving the WikiSQL task as a necessary preliminary step towards realizing natural language interfaces to databases in full fledge.

Chapter 5

Grounded Response Generation with Hierarchical Pointer Networks

5.1 Introduction

Recently, deep neural networks have achieved state-of-the-art results in various tasks including computer vision, natural language and speech processing. Specifically, neural sequence-to-sequence models [85, 86] have led to great progress in important downstream NLP tasks like text summarization [87, 88, 5, 89, 90], machine translation [91, 85, 92, 86, **?**], and reading comprehension [93]. However, achieving satisfactory performance on dialogue still remains an open problem. This is because dialogues can have multiple valid responses with varying semantic content. This is vastly different from the aforementioned tasks, where the generation is more conveniently and uniquely constrained by the input source.

Although neural models appear to generate meaningful responses when trained with sufficiently large datasets in the chit-chat setting, such generic chit-chat models reveal several weaknesses that were reported by previous research [94, 95]. Most common prob-

lems include inconsistency in personality, dull and generic responses, and unawareness of long-term dialogue context. To alleviate these limitations, we turn our focus on a different problem setting for dialogue response generation where the model is provided a set of relevant textual facts (speaker persona descriptions) and is allowed to harness this knowledge when generating responses in a multi-turn dialogue. To handle the personality inconsistency issue, we ground our dialogue generation model on external knowledge facts which are a list of persona descriptions in our application [96, 7]. We explicitly use the dialogue history as memory for the model to condition on which potentially encourages a more natural conversation flow. Towards encouraging generation of more specific and appropriate responses while avoiding generic and dull ones, we use a hierarchical pointer network in our model such that it can copy content from two sources: current dialogue history and persona descriptions.

In this work, we propose a novel and general architecture DeepCopy that extends the attentional sequence-to-sequence model with a hierarchical pointer network that enables the decoder to jointly attend and copy tokens from any of the facts available as external knowledge in addition to the dialogue context (encoder input). This is achieved entirely in an end-to-end fashion through factoring the whole copy mechanism into following three hierarchies/components: (i) a token-level attention mechanism over the dialogue context to determine the probability of copying a token from the dialogue context, (ii) A hierarchical pointer network to determine the probability of copying a token from each fact, and (iii) An inter-source meta attention over the input sources *dialogue context* and *external knowledge*, which combines the two copying probabilities. Using these components, a single copying probability distribution over the unique tokens appearing in the model input is computed exploiting the well-defined hierarchy among them. In addition, the model is equipped with a soft switch mechanism between *copying* and *generation* modes similar to [5], which allows us to softly combine the *copying probabilities* with the

decoder's *generation probabilities* over a fixed vocabulary into a final output probability distribution over an extended vocabulary. We empirically show the effectiveness of the proposed DeepCopy model compared to several baselines including [6, 7] on ConvAi2 challenge.

5.2 Related Work

Earlier work on data-driven, end-to-end approaches to conversational response generation treated the task as statistical machine translation, where the goal is to generate a response given the previous dialogue turn [97, 95]. While these studies resulted in a paradigm change compared to earlier work, they do not include mechanisms to represent conversation context. To tackle this problem and have a better representation of conversation context as input to generation, [94] proposed hierarchical recurrent encoder-decoder (HRED) networks. HRED combines two RNNs, one at the token level, modeling individual turns, and one at the dialogue level, inputting turn representations from the token-level RNNs. However, utterances generated by such neural response generation systems are often generic and contentless [95]. To improve the diversity and content of generated responses, HRED was later extended with a latent variable that aims to model the higher level aspects (such as topic) of the generated responses, resulting in the VHRED approach [98].

Another challenge for dialogue response generation is the integration of knowledge into the generated responses. [99] extracted facts relevant to a dialogue from knowledge using string matching, named entity recognition and linking, found additional entities from knowledge that are most relevant to the facts by a neural similarity scorer, and used these as input context features for the dialogue generation RNN. [6] used end-to-end memory networks to base the generated responses on knowledge, where an attention over

the knowledge relevant to the conversation context is estimated, and multiple knowledge representations are included as input during the decoding of responses. In this work, we use end-to-end memory networks as a baseline.

Although much research has focused on response generation in a chit-chat setting, models trained on large datasets of human-human interactions of diverse speaker characteristics often tend to generate responses which are too vague and generic (common for most speakers) or inconsistent in personality (switching between different speakers' characteristics). Recently, [7] presented the ConvAI2 challenge containing persona descriptions and over 10K real human chit-chats where speakers were required to converse based on their assigned persona. [96] learned speaker persona embeddings from a single-speaker setting (e.g. Twitter posts) or a speaker-address style (human-human conversations) to generate personalized responses given a single utterance input. Recently, several other dialogue challenges in both chit-chat [100] and task-oriented [101, 102, 103] settings have been released, further justifying the increasing research efforts in this domain. Another related work [104] applies hierarchical memory network for task oriented dialog problem. In this work, we compare our model with [6] and [7] which use a memory-augmented sequence-to-sequence response generator grounded on the dialogue history and persona.

5.3 Model

In this section, we first set up the problem, and then briefly revisit the baseline models using memory networks [105] and pointer-generator networks [5]. Subsequently, we introduce the proposed DeepCopy model with a hierarchical pointer network and our training process.

5.3.1 Problem Setup

Let $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ denote the tokens in the dialogue history. The dialogue is accompanied by a set of K relevant supporting facts, where $\mathbf{f}^{(i)} = (f_1^{(i)}, f_2^{(i)}, \ldots, f_{n_i}^{(i)})$ is the list of tokens in the i-th fact. Our goal is to generate the response as a sequence of tokens $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ using the dialogue history and supporting facts. Note here that we are not interested in retrieval/ranking based models [106] which rely on a set of candidate responses. Generative models are essential for this problem because we want to incorporate content from new facts during inference which may not be present in the training set. Hence, using a predefined set of candidates may not ensure high coverage.

5.3.2 Baseline Models

In this section, we describe several baseline response generation models including the ones from existing work [6, 7] and the in-house ones we propose as additional baselines.

Seq2Seq Models

In a sequence-to-sequence model with attention [86], a sequence of input tokens is encoded using an LSTM encoder. At decoder step t, the decoder state h_t, a context vector c_t and the previous decoder output y_{t-1} are used together to output a distribution over a fixed vocabulary of tokens obtained from the training set using a non-linear function. The context vector c_t is an attention-weighted combination of the encoder outputs. In the following baseline models, we use different features as inputs to the encoder. The underlying model remains the same.

Seq2Seq + NoFact. Only the dialogue context tokens $\mathbf{x}$ are used as input to the encoder.

Seq2Seq + BestFactContext. We select the fact $\mathbf{f}^{(c)}$ whose tokens have highest

unigram *tf-idf* similarity to the dialogue context tokens. $[\mathbf{x}||\mathbf{f}^{(c)}]$ is then used as input to the encoder, where || denotes concatenation.

Seq2Seq + BestFactResponse. We select the fact $\mathbf{f}^{(r)}$ whose tokens have highest unigram *tf-idf* similarity to the ground truth response. $[\mathbf{x}||\mathbf{f}^{(r)}]$ is used as input to the encoder. The aim of this experiment is to have a better understanding of the effect of fact selection on response generation, since using the ground truth for fact selection is not fair.

Memory Network Models

Our variations of Seq2Seq models described in Section 5.3.2 incorporate facts by concatenating them to the dialogue context. Memory networks [6, 7] are a more principled approach to incorporating external facts. Similar to [6], we use a context encoder to embed the context tokens $\mathbf{x}$ and obtain a list of outputs and final hidden state $u \in \mathbb{R}^d$. As outlined in [6], a fact $\mathbf{f}^{(i)}$ is embedded into key and value vectors k_i and m_i, respectively. A summary $o \in \mathbb{R}^d$ of facts is then computed as an attention weighted combination of $(m_1, m_2, \ldots, m_K)$ by conditioning on u and $(k_1, k_2, \ldots, k_K)$. We then combine the two summaries into $\hat{u} = u + o$, and use it to initialize the decoder state. We report results on the following variants:

MemNet. This is equivalent to the model used in [6], described above. This is essentially a sequence to sequence model without attention at every decoder step, except using the combined summary $\hat{u}$ to initialize the decoder.

MemNet+ContextAttention. At each decoder step, the decoder state attends over the encoder outputs and obtains a context vector $c_t^{(c)}$. This is equivalent to SEQ2SEQ + NOFACT model from Section 5.3.2, except using the fact summary $\hat{u}$ to initialize the decoder state.

MemNet+FactAttention. At each decoder step, we use the decoder state to attend

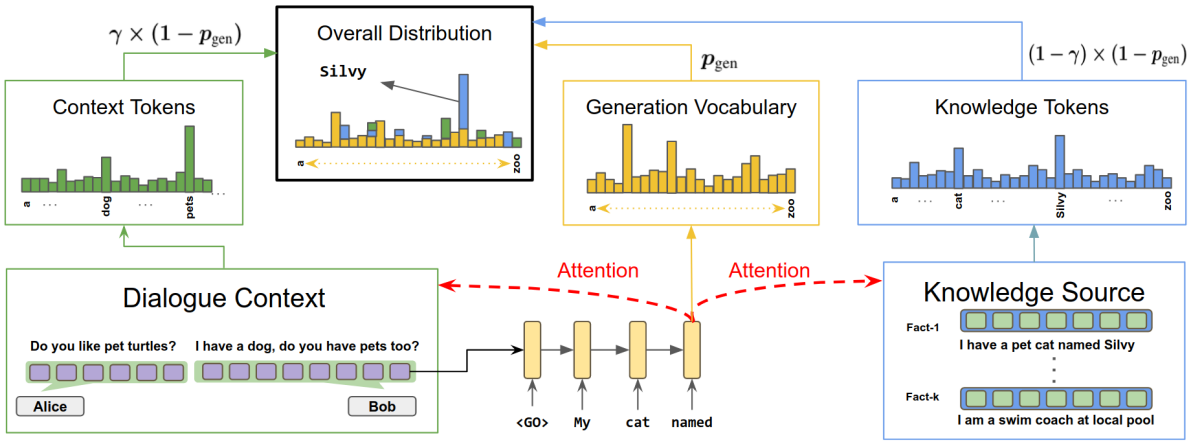

Figure 5.1: Overview of our proposed approach as described in Section 5.3.3. The decoder state d_t is used to attend over dialogue context and knowledge source to generate distributions for copying tokens from these sources. The decoder outputs a distribution over a fixed vocabulary. The three distributions are combined to yield the final distribution over tokens at each step t.

over the value embeddings $(m_1, m_2, \ldots, m_K)$ corresponding to facts, and obtain a context vector $c_t^{(f)}$. This model is similar to the *generative profile memory network* [7], where we apply attention only on facts, and we set the decoder's initial state to the combined summary $\hat{u}$.

MemNet+FullAttention. This model employs attention over both facts and dialogue context at each decoder step. The two attention modules are combined by concatenating $c_t^{(c)}$ and $c_t^{(f)}$ [107].

Seq2Seq Models with Copy Mechanism

Seq2seq models can only generate tokens present in a fixed vocabulary obtained from the training set. Pointer-generator network [5] extends the attentional sequence-to-sequence model [86] by employing a pointer network [70]. It has two decoding modes, copying and generating, which are combined via a soft switch mechanism, allowing it to copy tokens from source in addition to generating from vocabulary. We report the results for the following additional baselines obtained by equipping the corresponding Seq2Seq

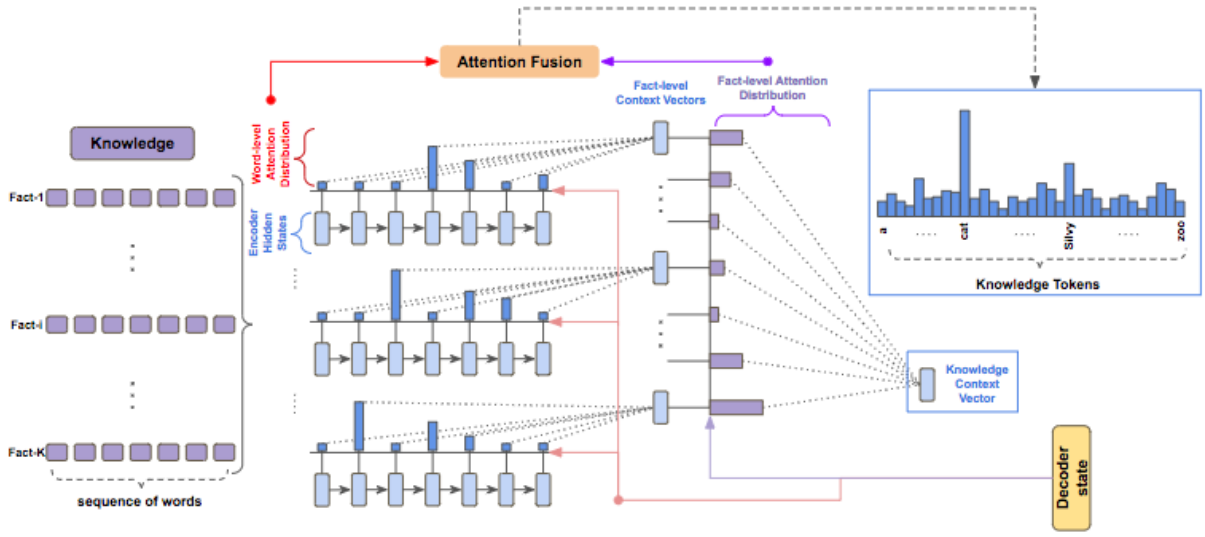

Figure 5.2: Illustration of hierarchical pointer network. The decoder state d_t is used to attend over tokens for each fact and also over the fact-level context vectors obtained by weighted average of token-level representations (w.r.t token-level attention weights) for each fact. The token-level attention weights are then combined with the attention distribution over facts (Equation 5.11) to generate the probability of copying each token in all the facts.

model in Section 5.3.2 with copy mechanism: Seq2Seq + NoFact + Copy, Seq2Seq + BestFactContext + Copy, Seq2Seq + BestFactResponse + Copy.

5.3.3 DeepCopy with Hierarchical Pointer Networks

In this section, we present our proposed DeepCopy model that extends pointer-generator network [5] using a novel hierarchical pointer network. Our model allows copying tokens from multiple input sources (facts $\mathbf{f}^{(i)}, 1 \leq i \leq K$), besides the encoder input (dialogue context x).

A high-level overview of the proposed approach is illustrated in Figure 5.1. At decoder step t, the decoder state h_t is used to attend over the dialogue context tokens and fact tokens to give a distribution over the tokens present in context and facts respectively. These distributions are then combined with the distribution output by the decoder over the fixed vocabulary to obtain the overall distribution.

Encoding a Sequence

Let $\mathbf{w} = (w_1, w_2, \ldots w_n)$ be a sequence of tokens. We first obtain a trainable embedded representation of each token in the sequence and then use a LSTM cell to encode the sequence of embedding vectors. We define $e, \mathbf{s} = \texttt{Encode}(\mathbf{w})$, where e denotes the final state of the LSTM and $\mathbf{s} = (s_1, s_2, \ldots s_n)$ denotes the outputs of the LSTM cell at all steps.

Attention

Let $\mathbf{u} = (u_1, u_2, \ldots u_n)$ be a sequence of vectors where $u_i \in \mathbb{R}^p, 1 \leq i \leq n$ and $v \in \mathbb{R}^q$ be a conditioning vector. The attention module generates a linear combination c of elements in $\mathbf{u}$ by conditioning them on v as defined by the equations below. We define $\alpha, c = \texttt{Attention}(\mathbf{u}, v)$, where $\alpha_i \in \mathbb{R}^n$ is the weight assigned to u_i, and $c \in \mathbb{R}^p$ is a vector representation of the sequence $\mathbf{u}$ conditioned on v. In the equations below, w_1 and W_2 are parameters of appropriate dimension. In our setup, we use $p = q, w_1 \in \mathbb{R}^p$, and $W_2 \in \mathbb{R}^{p \times 2p}$.

$$e_i = w_1^T \tanh(W_2[u_i; v]) \tag{5.1}$$

$$\alpha_i = \frac{\exp(e_i)}{\sum_{j=1}^{n} \exp(e_j)} \tag{5.2}$$

$$c = \sum_{i=1}^{n} \alpha_i u_i \tag{5.3}$$

Copying from Dialogue Context

Similar to our baseline models, we encode the dialogue context tokens $\mathbf{x}$ (Equation 5.4) and apply attention to the encoder outputs at a decoder step t (Equation 5.5). This outputs attention weights $\alpha_t^{(x)}$ and a representation of the entire context $c_t^{(x)}$. The attention weights are aggregated to obtain the distribution over context tokens $p_t^{(x)}(w)$ (Equation

5.6) as follows:

$$e^{(x)}, \mathbf{s}^{(x)} = \texttt{Encode}(\mathbf{x}) \tag{5.4}$$

$$\alpha_t^{(x)}, c_t^{(x)} = \texttt{Attention}(\mathbf{s}^{(x)}, h_t) \tag{5.5}$$

$$p_t^{(x)}(w) = \sum_{\{i:x_i=w\}} \alpha_{t,i}^{(x)} \tag{5.6}$$

Copying from Facts: Hierarchical Pointer Network

We introduce the hierarchical pointer network (Figure 5.2) as a general methodology for enabling token-level copy mechanism from multiple input sequences or facts. Each fact $\mathbf{f}^{(i)}$ is encoded (Equation 5.7) to obtain token level representations $\mathbf{s}^{(f)(i)}$ and overall representation $e^{(f)(i)}$. The decoder state h_t is used to attend over token level representations (Equation 5.8) and the overall fact-level representations of each fact (Equation 5.9) by

$$e^{(f)(i)}, \mathbf{s}^{(f)(i)} = \texttt{Encode}(\mathbf{f}^{(i)}) \tag{5.7}$$

$$\alpha_t^{(f)(i)}, c_t^{(f)(i)} = \texttt{Attention}(\mathbf{s}^{(f)(i)}, h_t) \tag{5.8}$$

$$\beta_t, c_t^{(f)} = \texttt{Attention}(\{c_t^{(f)(i)}\}_{i=1}^K, h_t) \tag{5.9}$$

to compute the probability of copying a word w from facts as

$$\begin{aligned} p_t^{(f)}(w) &= \sum_{j=1}^{K} p_t^{(f)}(\mathbf{f}^{(j)}) \cdot p_t^{(f)}(w|\mathbf{f}^{(j)}) \\ &= \sum_{j=1}^{K} \beta_{t,j} \sum_{\{l:f_l^{(j)}=w\}} \alpha_{t,l}^{(f)(j)} \end{aligned} \tag{5.10}$$

Inter-Source Attention Fusion

We now present the mechanism to fuse the two distributions $p_t^{(x)}(w)$ and $p_t^{(f)}(w)$ representing the probabilities of copying tokens from dialogue context and facts respectively. We use the decoder state h_t to attend over dialogue context representation $c_t^{(x)}$ and overall fact representation $c_t^{(f)}$ (Equation 5.11). The resulting attention weight $\gamma_t' = [\gamma_t, 1-\gamma_t]$

is used to combine the two copying distributions as shown in Equation 5.12.

$$\gamma_t, c_t = \texttt{Attention}([c_t^{(x)}, c_t^{(f)}], h_t) \tag{5.11}$$

$$p_t^{\text{copy}}(w) = \gamma_t \; p_t^{(x)}(w) + (1 - \gamma_t) \; p_t^{(f)}(w) \tag{5.12}$$

Similar to Seq2Seq models, the decoder also outputs a distribution p_t^{vocab} over the fixed training vocabulary at each decoder step using the overall context vector c_t and decoder state h_t. Having defined the copy probabilities p_t^{copy} for tokens that appear in the model input, either the dialogue context or the facts in external knowledge source, we combine p_t^{vocab} and p_t^{copy} using the mechanism outlined in [5], except we use c_t defined in Equation 5.11 as the context vector instead.

To better isolate the effect of copying, a key component of the proposed DeepCopy model, we also conduct experiments with MultiSeq2Seq model that incorporates the knowledge facts in the same way (by encoding each fact separately with LSTM, and attending on each by the decoder as in [107]), but relies completely on *generation probabilities* without a copy mechanism.

5.3.4 Training

We train all the models described in this section using the same loss function optimization. More precisely, given a model M that produces a probability $p_t(w|y_{<t})$ of generating token w at decoding step t, we train the whole network end-to-end with the negative log-likelihood loss function of

$$J_{\text{loss}}(\mathbf{\Theta}) = -\frac{1}{|\mathbf{y}|} \sum_{t=1}^{|\mathbf{y}|} \log(p_t(y_t|y_{<t}, \mathbf{x}, \{\mathbf{f}^{(i)}\}_{i=1}^K))$$

for a training sample $(\mathbf{x}, \mathbf{y}, \{\mathbf{f}^{(i)}\}_{i=1}^K)$ where $\mathbf{\Theta}$ denotes all the learnable model parameters.

5.4 Experiments

In this section, we describe the details of dataset, training process, evaluation metrics, and the performance results of DeepCopy model in comparison to proposed and existing baselines.

5.4.1 Dataset

We perform experiments for our problem setup on the recently released ConvAi2 *conversational AI challenge* dataset, which is an extended version of PersonaChat [7]. The conversations in ConvAi2 are obtained by asking a pair of crowdworkers to chat with each other naturally based on their randomly assigned personas (from a set of 1155 personas) towards getting to know each other. Personas are created by a different set of crowdworkers, and they consist of ~5 natural language sentences, each describing an aspect of a person that can range from common hobbies like *"I like to play basketball"* to very specific facts like *"I have a pet parrot named Tasha"*, reflecting a wide range of different personalities. The dataset contains ~11000 dialogues with ~160000 utterances, and 2000 dialogues with non-overlapping personas are used for validation and test. For our setting, we use personas as external knowledge sources that models can ground on while generating responses.

5.4.2 Training and Implementation Details

In all the models explored throughout this study, we set the dialogue context to concatenation of the last two dialogue turns separated by a special `CONCAT` token. The models are supplied with the persona facts of the side generating the response at the current turn, while the persona of the other side is concealed. We use a vocabulary of 18650 most frequent tokens and all the remaining tokens are replaced with a special `UNK`

token. Embeddings of size 100 are randomly initialized and updated during training. We set the size of LSTM hidden layer to 100 for both encoder and decoder. The encoder and decoder vocabularies and embeddings are shared. A shared LSTM encoder is used for encoding both dialogue context and facts of external knowledge source. The model parameters are optimized using Adam [50] with a batch size of 32, a fixed learning rate of 0.001. We apply gradient clipping to 5 when its norm exceeds this value. During inference, we generate responses by employing a beam search of width 4. Our models are implemented in *TensorFlow* [51].

5.4.3 Main Results

In this section, we present the experimental results in terms of both automatic measures and human evaluation.

Automatic Evaluation

In Table 5.1, we present our results in comparison with the existing and proposed baseline models. We report the performance of each model across several metrics commonly used for evaluation of text generation models including perplexity, corpus BLEU [108], ROUGE-L [109], CIDEr [110].

As expected, Seq2Seq + BestFactResponse model and its +COPY version outperform all the other models across all the evaluation metrics. This model pinpoints the importance of selecting the most suitable fact in the persona for the response to be generated at each turn, justifying our underlying motivation for conducting this experiment as highlighted in Section 5.3.2. However, the most suitable fact for the response is not available in the real application scenario, where the models are responsible for picking the useful pieces of information pertaining to the current dialogue turn to generate meaningful responses. Our proposed Seq2Seq + BestFactContext model and its

Model	PP	BLEU	R-L	CIDEr	App.
[M-1] MemNet	61.30	3.07	59.10	10.52	3.14 (0.51)
[M-2] MemNet + ContextAttention	57.37	3.24	59.20	11.79	3.41 (0.54)
[M-3] MemNet + FactAttention	61.50	2.43	59.34	9.65	1.45 (0.25)
[M-4] MemNet + FullAttention	59.64	3.26	59.18	12.25	3.20 (0.49)
[S2S-1] Seq2Seq + NoFact	60.48	3.38	59.46	11.41	3.12 (0.52)
[S2S-2] Seq2Seq + BestFactContext	58.68	3.35	59.13	10.77	3.08 (0.45)
[S2S-3] Seq2Seq + BestFactResponse*	49.74	4.02	60.04	16.15	2.97 (0.51)
[S2SC-1] S2S-1 + Copy	58.84	3.25	59.18	11.15	3.64 (0.54)
[S2SC-2] S2S-2 + Copy	60.25	3.17	59.46	11.17	3.60 (0.51)
[S2SC-3] S2S-3 + Copy*	38.60	4.54	60.96	21.47	3.83 (0.46)
[M-S2S] MultiSeq2Seq (no Copy)	57.94	2.88	59.10	10.92	3.32 (0.44)
DeepCopy†	**54.58**	**4.09**	**60.30**	**15.76**	**3.67** (0.59)
G.Truth	N/A	N/A	N/A	N/A	4.40 (0.45)

Table 5.1: Main results on Convai2 dataset. Evaluation metrics on last three columns are better the higher. Perplexity is lower the better. The results of the proposed approach are presented in bold. * indicates that the corresponding model should be considered as a kind of **ORACLE** because it has access to the fact that is most relevant to the ground-truth response during the inference/test time as defined in Section 5.3.2. † indicates that the improvement of DeepCopy in automatic evaluation metrics over each of the other models (except S2SC-3) is statistically significant with p-value of less than 0.001 on the paired t-test. **PP**, **R-L**, **App.** are abbreviations for **perplexity** and **ROUGE-L**, **appropriateness** metrics, respectively.

+COPY version, on the other hand, are valid baselines for this scenario where the best fact is selected completely based on the dialogue context without relying on the ground-truth response. This model outperforms the previously proposed memory network based model MemNet [6] for knowledge grounded response generation on all the evaluation metrics, demonstrating its effectiveness despite the fact that it does not have access to all the facts unlike [6]. However, this approach has the following potential weaknesses: (i) if the best persona fact selected w.r.t dialogue context is wrong (irrelevant) for the ground-truth response, the generated response might be drastically misinforming, and furthermore it is difficult for model to recover from this error because it has no access to other facts, (ii) selecting the best fact w.r.t dialogue context based on *tf-idf* similarity may result in poor fact selection when the lexical overlap between context and response

	Diversity	Fact-Inclusion			Agreement
Model	**Distinct**-2 / 3 / 4	**F.Inc**	**F.Per**	**F.Hal**	**F.Inc** / **F.Per**
M-1	.004 / .006 / .010	0.41	0.01	0.40	0.99 / 0.99
M-2	.010 / .019 / .031	0.43	0.01	0.42	0.97 / 0.99
M-3	.001 / .001 / .002	0.06	0.04	0.02	0.99 / 0.99
M-4	.054 / .010 / .156	0.51	0.09	0.42	0.98 / 0.98
S2S-1	.012 / .022 / .036	N/A	N/A	N/A	N/A / N/A
S2S-2	.012 / .022 / .035	0.54	0.04	0.50	0.97 / 0.99
S2S-3	.026 / .043 / .061	0.79	0.16	0.63	0.97 / 0.97
S2SC-1	.039 / .069 / .104	N/A	N/A	N/A	N/A / N/A
S2SC-2	.035 / .067 / .109	0.73	0.36	0.37	0.99 / 0.99
S2SC-3*	.058 / .111 / .178	0.73	0.55	0.18	0.98 / 0.96
M-S2S	.035 / .065 / .104	0.47	0.05	0.42	0.96 / 0.98
DeepCopy	.059 / .121 / .201	0.62	0.23	0.39	0.95 / 0.97
G.Truth	0.35 / 0.66 / 0.84	0.76	0.49	0.27	0.93 / 0.96

Table 5.2: Lexical diversity and fact inclusion analysis results. Model names are abbreviated according to Table 5.1. **F.Inc** denotes the ratio of responses that include factual information. **F.Per** and **F.Hal** denote the ratio of responses where the included fact is consistent with the persona or a hallucinated one, respectively. **Agreement** column corresponds to Cohen's κ statistic measuring inter-rater agreement on binary factual evaluation metrics for **F.Inc** and **F.Per**. * indicates the **ORACLE** model.

is small which might be a common case especially for the ConvAi2 dataset as the focus of conversation may often change swiftly across the dialogue turns. The latter might be the reason why copying does not help much for this model since it might end up copying irrelevant tokens in the scenario mentioned above.

Our proposed DeepCopy model is designed to effectively address the aforementioned issues, where it has access to the entire set of persona facts per dialogue from which it is expected to include the useful pieces of information in the response. DeepCopy model outperforms all the models reported in Table 5.1 except for Seq2Seq + BestContextResponse models, which we already deem as kind of an upper bound because it has access to the most relevant fact to the response. This justifies the effectiveness of

DeepCopy model compared to the existing works [6, 7] and the additional baselines we explored in this work. On the other hand, MultiSeq2Seq performs considerably worse than the DeepCopy model despite the fact they both have access to the entire set of facts and employ the same encoder-decoder architecture except for the copy mechanism. This further justifies the effectiveness of incorporating the proposed hierarchical pointer networks in DeepCopy because integrating the external knowledge simply by employing multi-source attention as in [107] does not yield to a good solution with competitive results, performing even worse than Seq2Seq + NOFACT on 3 of the metrics.

Human Evaluation

Although automatic metrics provide tangible information regarding the performance of the models, we augment them with human evaluations for a more comprehensive analysis of the resulting model generated responses. Towards this end, we randomly sample 100 examples from test data and ask human raters to evaluate the candidate model generated responses in terms of appropriateness. Each example is rated by 3 raters, who are shown a dialog history along with a set of persona facts (of the person in turn), and asked to rate each response based on its *appropriateness* in the dialogue context with a score from 1 (worst) to 5 (best).

In Table 5.1, we present the results of human evaluation under the *appropriateness* column. Since each response is rated by 3 different human raters, we report the average rating along with the standard deviation in parenthesis. We observe that DeepCopy outperforms both the existing memory-network baselines and the proposed sequence-to-sequence baselines on the appropriateness evaluation. It also achieves a performance that is close to the *oracle* model (S2SC-3), which has a leverage of having an access to the fact that is most relevant to the ground-truth response during the inference time. Overall, human evaluation of the responses in terms of appropriateness further justifies

the promise and effectiveness of our proposed DeepCopy model.

Appropriateness scores also demonstrate the advantage of incorporating the soft copy mechanism. Comparing S2S (and M-S2S) models to their copy-equipped counterparts (S2SC) (and DeepCopy) in Table 5.1 immediately reveals a significant gain in appropriateness score. Another significant observation to note here is that ground-truth responses obtain an average appropriateness score of 4.4/5, which reflects both the noise in ConvAI2 dataset and the difficulty of generating the perfect response even for humans.

5.4.4 Further Analysis and Discussion

Lexical Diversity Analysis

In Table 5.2, we report the lexical diversity results using the distinctness metric introduced in [111]. *distinct-n* score corresponds to the number of distinct n-grams divided by total number of generated n-grams. We can clearly observe that DeepCopy generates the most diverse responses among all the models including the copy-augmented oracle model (S2SC-3). Hence, diversity results further show that our proposed model is promising in addressing the most commonly observed *generic response problem* more effectively than existing models by generating more diverse responses.

Fact Inclusion Analysis

We also conduct an analysis on the kinds of factual information included in the model-generated responses. More precisely, our goal is to understand how often the generated response includes a factual information (F.Inc), and whether this information is consistent with the persona facts (F.Per) or a hallucinated one (F.Hal). A good model can naturally include available facts from the persona and hallucinate others when the conversation context requires them. Towards this end, we ask 3 human raters to label responses with

1 (or 0) based on whether a fact is included, and if so, whether this fact is a persona-fact or not.

In Table 5.2, we present an analysis for the kinds of factual information included in model generated responses. As can be seen from this analysis, models that have a copy mechanism include more facts from the persona than the ones that do not. Another important observation is that the ground-truth responses include facts from persona only in 49% of the times, which indicates that the provided persona facts remain insufficient to cover the complexity of the high entropy open-ended person-to-person conversations.

In Table 5.2, we present Cohen's κ score for each model and fact analysis metric pair using the scores from 3 raters for each example. We observe for each model and metric pair a κ statistic of greater than 0.9, which indicates a near perfect agreement among raters. Note that the ratio of hallucinated facts (**F.Hal**) is derived directly from human labels for fact inclusion (**F.Inc**) and persona-fact (**F.Per**). That is why, there is no separate labelling process for hallucinated facts (**F.Hal**). Hence, there is no κ statistic for **F.Hal** in Table 5.2.

Error Analysis

A deeper analysis of the examples where DeepCopy is assigned a worse appropriateness score than the best performing memory-network based baselines (M-2 and M-4) reveals the following further insights: (i) Some of these examples are corresponding to the cases where a generic response (e.g., "I've a dog named radar", one of the frequent generic responses, completely independent of persona facts) is rated much higher (5 to 1) than factual but slightly off (by a single word in this example) responses (e.g., "I have a dog for a living." coming from the persona fact "I walk dogs for a living."), (ii) In another subset of the analyzed examples, DeepCopy model generates a response (e.g., "yes, but I want to become a lawyer.") by incorporating a fact that has already been used in the previous turn of the dialog whereas M-2 produces a generic response (e.g., "that's great.

Persona Facts
1. i'm a clean eater.
2. i'm a cancer survivor.
3. my parents were both very athletic.
4. **i got a new job just yesterday to be a life coach.**
5. i love running and preparing for marathons.

Model	**Conversation/Response**
PERSON1	i really miss it but i have been eating healthy ever since i overcame cancer
PERSON2	omg i am glad you did , do you work now ?
MEMNET*	yes, i do not work, but i do not work.
SEQ2SEQ*	no i ' ve a job at a restaurant
SEQ2SEQ**	i'm a life coach
DEEPCOPY	yes, i just got a new job

Figure 5.3: Example dialogue where the previous two turns from PERSON1 and PERSON2 along with the responses generated by the models acting as PERSON1 are shown on the right. Persona facts for PERSON1 are provided on the left, among which the one in bold is the best fact w.r.t response. MEMNET*, SEQ2SEQ*, SEQ2SEQ** are abbreviations for MEMNET + FULLATTENTION, SEQ2SEQ + BESTFACTRESPONSE, SEQ2SEQ + BESTFACTRESPONSE +COPY models, respectively.

do you have any hobbies?", again irrelevant to facts) which is rated higher. (iii) And most of the remaining cases fall into the class of examples where incorporating knowledge facts breaks the conversation flow, which is a crucial observation specific to this dataset that can also be supported by the low persona-fact inclusion ratio (49%) of ground-truth responses.

5.4.5 Qualitative Observations

In Figure 5.3, we present an example dialogue where DEEPCOPY model generates a meaningful and fluent response by effectively mixing *copy* and *generate* modes. We can observe that it is able to attend on the right persona fact by taking the dialogue context (especially the question at the end of PERSON2's turn) into consideration. Furthermore, attending to the tokens of this fact, it produces a fluent and valid answer to yes/no question by generating "yes" and copying the rest (and most) of the tokens from the fact.

Although it copies most of the tokens from the fact, it is good to observe that it copies exactly the relevant pieces instead of just copying the entire fact. SEQ2SEQ + BESTFACTRESPONSE + COPY model's response is also meaningful and fluent although it may not be as engaging for the continuation of dialog. However, the quality of the response by SEQ2SEQ + BESTFACTRESPONSE quickly degrades compared to its +COPY version. Although the response is still fluent and relevant to the dialogue context, it becomes rather irrelevant to the persona as the model seems to have difficulty of picking the useful information from even the best persona fact it is provided with when the copy mechanism is disabled. Lastly, the response generated by MEMNET+FULLATTENTION model seems to still suffer from repetition, semantic consistency, and relevancy problems that were observed and reported by previous work.

5.5 Conclusion

We propose a hierarchical pointer network for knowledge grounded dialogue response generation. Our approach extends the pointer-generator network to enable the decoder to simultaneously copy tokens from the available set of relevant external knowledge in addition to dialogue context. We demonstrate the effectiveness of our approach through

various automatic and human evaluations in comparison with several baselines on the ConvAi2 dataset. Furthermore, we conduct diversity, fact inclusion, and error analysis providing further insights into model behaviors. In the future, we plan to apply our model to datasets of the same fashion where the dialogue is accompanied by a much larger set of knowledge facts (e.g., Wikipedia articles) [112]. This could be done by adding a retrieval component which identifies a few contextually relevant facts [6] to be used as input to DeepCopy.

Chapter 6

Improved Sequence-Level Optimization with CaLcs

6.1 Introduction

Recently, deep neural networks have achieved state-of-the-art results in various tasks including computer vision, natural language processing, and speech processing. Specifically, neural text generation models, central focus of this work, have led to great progress in central downstream NLP tasks like text summarization [87, 88, 5], machine translation [91, 85, 86], reading comprehension [93], and dialogue response generation [94, 98, 6, 113]. For example, the abstractive summarization task, which has previously not been the popular choice for text summarization due to lack of appropriate text generation methods, has gained revived attention with the success of neural sequence-to-sequence models. There has been several recent work with an impressive progress on this task including [87, 114, 88, 115, 5, 89, 116]. Machine translation is another central field in NLP where the emergence of neural sequence-to-sequence models has enabled viable alternative approaches [52, 117, 118, 119, 120] to challenge traditional phrase-based methods [121].

Most of the recent existing works on neural text generation are based on variants of sequence-to-sequence models with attention [86] trained with Maximum-likelihood estimation (MLE) with teacher forcing. As [122] points out in a previous work, these models have two major drawbacks. First, they are trained to maximize the probability of correct next word given the entire sequence of previous ground truth words. While, at test time, the models need to generate the entire sequence by feeding its own predictions at previous time steps. This discrepancy is called *exposure bias* and hurts the performance as the model is never exposed to its own predictions during training. The second drawback, called *wrong objective*, is due yet another discrepancy between training and testing. It refers to the critique [122] that MLE-trained models tend to have suboptimal performance as they are trained to maximize a convenient objective (i.e., maximum likelihood of word-level correct next step prediction) rather than a desirable sequence-level objective that correlates better with the common discrete evaluation metrics such as ROUGE [109] for summarization, BLEU [108] for translation, and word error rate for speech recognition, not log-likelihood. On the other hand, training models that directly optimize for such discrete metrics as objective is hard due to non-differentiable nature of the corresponding loss functions [123]. To address these issues, [122] introduces an incremental learning recipe that uses a hybrid loss function combining REINFORCE [124] and cross-entropy. Recently, [125] also explored combining maximum-likelihood and policy gradient training for text summarization.

Towards sequence level optimization, previous works [122, 126, 125] employ reinforcement learning (RL) with a policy-gradient approach which works around the difficulty of differentiating the reward function by using it as a weight. However, REINFORCE is known to suffer from high sample variance and credit assignment problems which makes the training process difficult and unstable besides resulting in models that are hard to reproduce [127].

In this chapter, we propose an alternative approach for sequence-level training with longest common subsequence (LCS) metric that measures the sequence-level structure similarity between two sequences. We essentially introduce a continuous approximation to the discrete LCS metric which can be directly optimized against using standard gradient-based methods. Our proposed approach has the advantage of being able to directly optimize for a surrogate reward as opposed to using the exact reward only as a weight as in RL-inspired works. Hence, it provides a viable alternative perspective to policy-gradient methods for side stepping the non-differentiability with respect to the exact reward. In addition, it simultenuously combats the *exposure bias* problem through exposing the model to its own predictions while computing our approximation to LCS metric.

To this end, we introduce a new learning recipe that incorporates the aformentioned continuous approximation to LCS metric (CaLcs) as an additional objective on top of maximum-likelihood loss in existing neural text generation models. We evaluate the proposed approach on abstractive text summarization and machine translation tasks. To this end, we use recently introduced *pointer-generator* network [5] and *transformer* [117] as underlying baselines for summarization and machine translation, respectively. More precisely, we start from a pre-trained baseline model with cross-entropy loss, and continue training the model to optimize for the proposed differentiable objective based on CaLcs. Using this recipe, we conduct various experiments on *CNN/Daily Mail* [128, 88] summarization and *WMT 2014 English-to-German* machine translation tasks. Experimental results validate the effectiveness of the proposed approach on both tasks.

6.2 Continuously Approximating Longest Common Subsequence Metric

In this work, we explore the potential use of longest common subsequence (LCS) metric from an algorithmic point of view to address the aforementioned *wrong objective* and *exposure bias* problems. LCS metric measures a sequence-level structure similarity between discrete sequences by identifying longest co-occurring in sequence n-grams and it has been shown to correlate well with human judgments for downstream text generation tasks [109]. To this end, we propose a way to continuously approximate LCS metric and use this differentiable approximation as the objective to train text generation models rather than the exact LCS measure, which is hard to optimize for due to non-differentiability of the corresponding loss function. Although such differentiable approximation provides a unique advantage from modeling and optimization perspective, the difficulty of controlling its tightness might be a potential drawback in terms of its applicability. In this section, we will first introduce our proposed approximation to LCS metric, and then provide a natural way to control its tightness.

Consider a sequence generation problem conditioned on an input sequence $\mathbf{x} = (x_1, x_2, \ldots, x_n)$ and let $\mathbf{y} = (y_1, y_2, \ldots, y_m)$ denote its corresponding ground-truth output sequence. Let

$$f(\mathbf{x}, \boldsymbol{\Theta}) = \mathbf{z} = (z_1, z_2, \ldots, z_k)$$

denote hypothesis sequence obtained by greedy decoding from a generic encoder-decoder architecture for input sequence $\mathbf{x}$, where $\boldsymbol{\Theta}$ represents model parameters. Also, let $p_1, p_2, \ldots, p_k$ be the probability distributions over vocabulary V at decoding time steps from which $z_1, z_2, \ldots, z_k$ are generated via *argmax* operator, respectively.

6.2.1 CaLcs

In this section, we define our approach to continuously approximate the longest common subsequence measure (LCS), which is an unnormalized version of ROUGE-L metric [109] that is commonly used for performance evaluation of text summarization models. The main intuition behind our approach is to relax the common necessity for hard inferences while computing discrete metrics by instead comparing discrete tokens in a soft way. Towards this end, we start by defining LCS metric.

Definition 1 *Given two sequences* $\mathbf{y}$ *and* $\mathbf{z}$ *of tokens, longest common subsequence* $LCS(\mathbf{y}, \mathbf{z})$ *is defined as the longest sequence of tokens that appear left-to-right (but not necessarily in a contiguous block) in both sequences.*

The most common and intuitive solution for computing longest common subsequence is via dynamic programming. We will briefly revisit this here as it will be useful in terms of both recall and notational convenience while describing our surrogate LCS measure. Let $r_{i,j}$ denote the longest common subsequence between prefix sequences $\mathbf{y}_{[:i]} = (y_1, y_2, \ldots, y_i)$ and $\mathbf{z}_{[:j]} = (z_1, z_2, \ldots, z_j)$ of $\mathbf{y}$ and $\mathbf{z}$, respectively. A dynamic programming solution is given by

$$r_{i,j} = \begin{cases} 0 & \text{if } i = 0 \text{ or } j = 0 \\ r_{i-1,j-1} + 1 & \text{if } y_i = z_j \\ max(r_{i-1,j}, r_{i,j-1}) & \text{o/w.} \end{cases} \tag{6.1}$$

$r_{i,j}$ for all $i = 1, 2, \ldots, m$ and $j = 1, 2, \ldots, k$. It can be computed in mk iterations using the formula in Eqn 6.1. After computing 2D dynamic programming matrix r, we obtain $\text{LCS}(\mathbf{y}, \mathbf{z}) = r_{m,k}$.

Towards removing the dependence on hard inference for computing LCS, we now define our approximation to longest common subsequence, which we call CaLcs. At

high-level, the idea is to continuously relax the original LCS measure. To this end, we leverage output probability distributions $p_1, p_2, \ldots, p_k$ as soft predictions to refine the dynamic programming formulation for original LCS. More precisely, we recursively define soft longest common subsequence $s_{i,j}$ between prefixes $\mathbf{y}_{[:i]}$ and $\mathbf{z}_{[:j]}$ in analogous to $r_{i,j}$ as follows:

$$s_{i,j} = p_j^{(y_i)}(s_{i-1,j-1} + 1) + (1 - p_j^{(y_i)}) \max(s_{i-1,j}, s_{i,j-1}) \tag{6.2}$$

for $i, j > 0$ and $s_{i,0} = s_{0,j} = 0$, where $p_j^{(y_i)}$ denote the probability of generating y_i at j-th decoding step. Intuitively, CALCS replaces the hard token comparison $1\,[y_i = z_j]$ in Eq. 6.1 with the probability $p_j^{(y_i)}$ that appear in Eq. 6.2. Interpreting the probability $p_j^{(y_i)}$ as a continuous relaxation of discrete comparison operator $1\,[y_i = z_j]$, $s_{i,j}$ establishes a natural continuous approximation to $r_{i,j}$. Similar to LCS, after iteratively filling up $s_{i,j}$ matrix, we define

$$\text{CALCS}(\mathbf{y}, \mathbf{z}) = s_{m,k} \tag{6.3}$$

Although the proposed approximation is a natural way of relaxing/extending the hard binary comparison of discrete tokens, it is not clear how tight the approximation is, which is established in the next section.

6.2.2 On the Tightness of Approximation

In this section, we first discuss the tightness of the proposed approximation, and then provide a natural way of controlling it.

Bounding the Approximation Error

We now present a bound on the approximation error of the proposed CALCS compared to the original LCS measure. Characterization of this bound will enable us to

theoretically argue about the feasibilty of using the proposed surrogate reward function for our objective as well as controlling its tightness.

LCS measure is intrinsically monotonic by definition. We start by a lemma that establishes a similar monotonicity property for CaLcs.

Lemma 1 *[Monotonicity] The following two inequalities*

$$s_{i,j} \leq s_{i,j+1} \leq s_{i,j} + 1$$

$$s_{i,j} \leq s_{i+1,j} \leq s_{i,j} + 1$$

hold for all $0 \leq i < m$ *and* $0 \leq j < k$.

Proof: We will prove this lemma by induction on $i + j$.

Base Case: $i + j = 0$. In this case, we have $i = j = 0$. Since $s_{0,0} = s_{1,0} = s_{0,1} = 0$ by definition, both $s_{0,0} \leq s_{0,1} \leq s_{0,0} + 1$ and $s_{0,0} \leq s_{1,0} \leq s_{0,0} + 1$ hold.

Inductive Step: Assume for $i + j = l$ that the following two inequalities hold:

$$s_{i,j} \leq s_{i,j+1} \leq s_{i,j} + 1 \tag{6.4}$$

$$s_{i,j} \leq s_{i+1,j} \leq s_{i,j} + 1 \tag{6.5}$$

We will now prove that the inequalities of inductive hypothesis hold for $i + j = l + 1$.

We will start by showing that $s_{i,j+1} \geq s_{i,j}$ holds. By definition, we have

$$s_{i,j+1} = p_{j+1}^{(y_i)}(s_{i-1,j} + 1) + (1 - p_{j+1}^{(y_i)}) \max(s_{i-1,j+1}, s_{i,j}) \tag{6.6}$$

$$\geq p_{j+1}^{(y_i)}(s_{i-1,j} + 1) + (1 - p_{j+1}^{(y_i)}) s_{i,j} \tag{6.7}$$

$$\geq p_{j+1}^{(y_i)} s_{i,j} + (1 - p_{j+1,y_i}) s_{i,j} \tag{6.8}$$

$$\geq s_{i,j} \tag{6.9}$$

where inequality 6.7 follows from the definition of max operator, and inequality 6.8 follows from induction assumption 6.4 because $(i-1) + j = l$. Hence, final inequality 6.9

establishes the proof of $s_{i,j+1} \geq s_{i,j}$.

Now, we will show that $s_{i,j+1} \leq s_{i,j} + 1$ holds. Again by definition, we have

$$s_{i,j+1} = p_{j+1}^{(y_i)}(s_{i-1,j} + 1) + (1 - p_{j+1}^{(y_i)}) \max(s_{i-1,j+1}, s_{i,j}) \tag{6.10}$$

$$\leq p_{j+1}^{(y_i)}(s_{i-1,j} + 1) + (1 - p_{j+1}^{(y_i)})(s_{i-1,j} + 1) \tag{6.11}$$

$$\leq s_{i-1,j} + 1 \tag{6.12}$$

$$\leq s_{i,j} + 1 \tag{6.13}$$

where inequalities 6.11 and 6.13 follow from inequalities 6.4 and 6.5 of inductive step as $(i-1) + j = l$.

Note that 6.9 and 6.13 completes the proof of $s_{i,j} \leq s_{i,j+1} \leq s_{i,j} + 1$ for $i + j = l + 1$. Following similar arguments, one can easily establish the correctness of $s_{i,j} \leq s_{i+1,j} \leq s_{i,j} + 1$ for $i + j = l + 1$, which completes the proof of Lemma by induction. ■

Having established a certain monotonicity property for CaLcs, we will discuss its approximation error to the original LCS measure. Let

$$\delta_{i,j} = s_{i,j} - r_{i,j} \tag{6.14}$$

denote the *approximation error* of CaLcs to LCS measure between generated prefix sequence $\mathbf{y}_{[:i]}$ and the ground-truth prefix $\mathbf{z}_{[:j]}$.

Lemma 2 *Let $P_{i,j} = \{(0,0), (i_1, j_1), \ldots, (i_{q-1}, j_{q-1}), (i_q, j_q)\}$ denote the path of dynamic programming algorithm for LCS ending at $(i,j) = (i_q, j_q)$ cell of $m \times k$ grid. Then,*

$$|\delta_{i,j}| < \left|\delta_{i_{q-1}, j_{q-1}}\right| + (1 - max(p_j)) \tag{6.15}$$

where $max(p_j) = \max\{p_j^{(1)}, p_j^{(2)}, \ldots, p_j^{(|V|)}\}$.

Proof: We will establish the proof by investigating two cases and combining them.

CASE 1: $z_j = y_i$.

In this case, we have

$$r_{i,j} = 1 + r_{i-1,j-1} \tag{6.16}$$

$$(i_{q-1}, j_{q-1}) = (i-1, j-1) \tag{6.17}$$

by 6.1. Using Eq. 6.16, we get

$$\begin{aligned}
\delta_{i,j} &= s_{i,j} - r_{i,j} \\
&= \left(1 - p_j^{(y_i)}\right) \max\left(s_{i-1,j}, s_{i,j-1}\right) + \ p_j^{(y_i)} \left(s_{i-1,j-1} + 1\right) - \left(1 + r_{i-1,j-1}\right) \\
&= \left(s_{i-1,j-1} - r_{i-1,j-1}\right) + \ \left(1 - p_j^{(y_i)}\right) \Big[\max(s_{i-1,j}, s_{i,j-1}) - \ (1 + s_{i-1,j-1}) \Big]
\end{aligned}$$

Using the definition of δ and triangle inequality, we get

$$\begin{aligned}
|\delta_{i,j}| &\leq |\delta_{i-1,j-1}| + \left(1 - p_j^{(y_i)}\right) \Big| \left(1 + s_{i-1,j-1}\right) - \ \max\left(s_{i-1,j}, s_{i,j-1}\right) \Big| \\
&\leq |\delta_{i-1,j-1}| + \left(1 - p_j^{(y_i)}\right)
\end{aligned} \tag{6.18}$$

where inequality 6.18 follows from the monotonicity established by Lemma 1.

Moreover, $z_j = y_i$ implies $p_j^{(y_i)} = \max(p_j)$ because $\mathbf{z}$ is generated by greedy decoding. Plugging this in Eq. 6.18 and using Eq. 6.17, we can immediately conclude that

$$|\delta_{i,j}| < \left|\delta_{i_{q-1}, j_{q-1}}\right| + (1 - \max(p_j)) \tag{6.19}$$

CASE 2: $z_j \neq y_i$.

By definition 6.1, we have $r_{i,j} = \max\left(r_{i-1,j}, r_{i,j-1}\right)$. Using this identity, we obtain

$$\begin{aligned}
\delta_{i,j} &= s_{i,j} - r_{i,j} \\
&= \Big[\left(1 - p_j^{(y_i)}\right) \max\left(s_{i-1,j}, s_{i,j-1}\right) \ + p_j^{(y_i)} \left(s_{i-1,j-1} + 1\right) \Big] - \max\left(r_{i-1,j}, r_{i,j-1}\right) \\
&= p_j^{(y_i)} \left[\left(1 + s_{i-1,j-1}\right) - \max\left(s_{i-1,j}, s_{i,j-1}\right)\right] + \left[\max\left(s_{i-1,j}, s_{i,j-1}\right) - \max\left(r_{i-1,j}, r_{i,j-1}\right)\right]
\end{aligned}$$

Applying triangle inequality on the last equation above, we get

$$|\delta_{i,j}| \leq p_j^{(y_i)} \left|(1 + s_{i-1,j-1}) - \max\left(s_{i-1,j}, s_{i,j-1}\right)\right| + \left|\max\left(s_{i-1,j}, s_{i,j-1}\right) - \max\left(r_{i-1,j}, r_{i,j-1}\right)\right|$$
$$\leq p_j^{(y_i)} \left|(1 + s_{i-1,j-1}) - \max\left(s_{i-1,j}, s_{i,j-1}\right)\right| + \max\left(\left|s_{i-1,j} - r_{i-1,j}\right|, \left|s_{i,j-1} - r_{i,j-1}\right|\right) \tag{6.20}$$

$$= p_j^{(y_i)} \left|(1 + s_{i-1,j-1}) - \max\left(s_{i-1,j}, s_{i,j-1}\right)\right| + \max\left(\left|\delta_{i-1,j}\right|, \left|\delta_{i,j-1}\right|\right)$$
$$\leq p_j^{(y_i)} + \max\left(\left|\delta_{i-1,j}\right|, \left|\delta_{i,j-1}\right|\right) \tag{6.21}$$

where inequality 6.21 follows from again the monotonicity of $s[\cdot, \cdot]$, and inequality 6.20 follows from the following identity that holds true for all real numbers $a, b, c, d \geq 0$

$$|\max(a, b) - \max(c, d)| \leq \max(|a - c|, |b - d|)$$

Moreover, since $z_j \neq y_i$, we know that $p_j^{(y_i)} \neq \max(p_j)$, which implies

$$p_j^{(y_i)} \leq 1 - \max(p_i). \tag{6.22}$$

Combining 6.20 and 6.22 completes the proof for this case. Finally, two cases investigated above together establish the proof of Lemma 2. ■

Lemma 2 leads to the following important corollary.

Corollary 1 *Let $P_{i,j} = \{(0,0), (i_1, j_1), \ldots, (i_q, j_q)\}$ be the path of dynamic programming algorithm for LCS ending at $(i, j) = (i_q, j_q)$ cell of $m \times k$ grid. Then,*

$$|\delta_{i,j}| \leq \sum_{w=1}^{q} (1 - \max(p_{j_w})) \tag{6.23}$$

Proof: Applying Lemma 2 iteratively and using $\delta_{0,0} = 0$, we get

$$\begin{aligned}
|\delta_{i,j}| &\leq \left|\delta_{i_{q-1},j_{q-1}}\right| + (1 - \max(p_{j_q})) \\
\left|\delta_{i_{q-1},j_{q-1}}\right| &\leq \left|\delta_{i_{q-2},j_{q-2}}\right| + (1 - \max(p_{j_{q-1}})) \\
\left|\delta_{i_{q-2},j_{q-2}}\right| &\leq \left|\delta_{i_{q-3},j_{q-3}}\right| + (1 - \max(p_{j_{q-2}})) \\
&\vdots \\
|\delta_{i_1,j_1}| &\leq |\delta_{0,0}| + (1 - \max(p_{j_1})) \\
|\delta_{0,0}| &\leq 0
\end{aligned}$$

Summing $(q+1)$-many inequalities above side by side and cancelling out the same terms appearing on both sides of the resulting inequality establishes the proof of corollary. ■

Controlling the Tightness of Approximation

Corollary 1 hints for a natural way of controlling the tightness of approximation CALCS by exploiting the peakedness of model's softmax output probability distributions. More precisely, upper bound on the approximation error is represented as a sum of $1 - \max(p_j)$'s, hence the more peaked the model's output probability distributions on average, the smaller the approximation error we are guaranteed by the established bounds.

We exploit this property to control the tightness of approximation by making a modification to computation of the proposed CALCS measure. Formally, let $l_1, l_2, \ldots, l_k$ denote the unnormalized logits of the model output before applying softmax to obtain probabilities $p_1, p_2, \ldots, p_k$ at decoding time steps, respectively. Hence,

$$p_j^{(i)} = \frac{\exp(l_j^{(i)})}{\sum_i \exp(l_j^{(i)})} \tag{6.24}$$

Recall that CALCS is computed using p_j's. Using *peaked softmax*, we can obtain more peaked probability distributions without causing any change in the actual generated

sequence $\mathbf{z}$ via greedy decoding. This is simply because the order of probabilities for corresponding vocabulary words will not change, only the probability disribution p_j will get more peaked. So, we define *peaked softmax* operator with hyperparameter α as

$$p_j^{(i)}(\alpha) = \frac{\exp(l_j^{(i)}/\alpha)}{\sum_i \exp(l_j^{(i)}/\alpha)} \tag{6.25}$$

By Corollary 1, $|\delta_{i,j}| \to 0$ as $\alpha \to 0$ for CaLcs measure computed with $p_j(\alpha)$. One can further attempt to use Corollary 1 as a guide to pinpoint a range of α values to force the approximation error within certain desired limits. We will use α as a hyperparameter in this work.

Corollary 1 is also useful for alternative ways of controlling the tightness of approximation such as incurring penalty for high-entropy output probability distributions or simply penalizing the maximum output probability values less than a desired threshold (that explicitly controls the tightness of the approximation). We leave such options of controlling the approximation error for future work.

With the guidance of Corollary 1 and *peaked softmax* in Eq. 6.25, we conclude that CaLcs establishes a promising approximation for LCS measure. In the next section, we introduce a new objective function using CaLcs as a continuously differentiable reward to be directly maximized.

6.2.3 Sequence Level Optimization via CaLcs

In this section, we describe how to leverage CaLcs to define a loss function for sequence level optimization. For notational consistency, we will use $f(\mathbf{x}, \boldsymbol{\Theta})$ to denote an encoder-decoder architecture that takes an input sequence $\mathbf{x}$ and outputs a sequence of tokens $\mathbf{z} = (z_1, z_2, \ldots, z_m)$ via greedy decoding from corresponding probability distributions $p_1, p_2, \ldots, p_m$ at each step.

For a pair of input sequence $\mathbf{x}$ and its corresponding ground-truth output sequence $\mathbf{y}$, we define

$$J_{\text{CaLcs}}(\mathbf{x}, \mathbf{y}; \mathbf{\Theta}) = -\log\left(\frac{\text{CaLcs}(\mathbf{y}, f(\mathbf{x}, \mathbf{\Theta}))}{|\mathbf{y}|}\right) \tag{6.26}$$

as the loss function for a sample $(\mathbf{x}, \mathbf{y})$ based on the CaLcs, where $|\mathbf{y}|$ denote the length of sequence $\mathbf{y}$. It is important to note here that while computing probability distribution p_t at decoding step t, we feed model's own prediction z_{t-1} at the previous time step to fight *exposure bias*.

It is important to observe here that $J_{\text{CaLcs}}(\mathbf{x}, \mathbf{y}; \mathbf{\Theta})$ is differentiable in $p_1, p_2, \ldots, p_k$ by definition and each p_i is differentiable in model parameters $\mathbf{\Theta}$. Hence, $J_{\text{CaLcs}}(\mathbf{x}, \mathbf{y}; \mathbf{\Theta})$ is differentiable in model parameters $\mathbf{\Theta}$, which allows us to directly optimize the network parameters with respect to LCS metric. The bound we established on the approximation error and our proposed strategy to control it theoretically ensures the feasibility of using the introduced loss function J_{CaLcs} to optimize for LCS metric.

6.3 Model

In this section, we first briefly revisit the pointer-generator [5] and transformer [117] networks that are used as the underlying baselines in our experiments. Subsequently, we describe how the proposed objective function and its variants are used to train new summarization and machine translation models.

6.3.1 Baseline Models

Pointer-Generator Network. We use pointer-generator network [5] as our baseline sequence-to-sequence model for text summarization. It is essentially a hybrid between sequence-to-sequence model with attention [86] and a pointer network [70] that supports

two decoding modes, copying and generating, via a soft switch mechanism. This enables the model to copy a word from the input sequence based on the attention distribution. On each decoding time step t, the decoder LSTM is fed the word embedding of the previous word, and computes a decoder state s_t, an attention distribution a^t over the words of input article, and a probability $P_{\text{vocab}}(w)$ of generating word w for summary from output vocabulary V, which is then softly combined with the *copy* mode's probability distribution $P_{\text{copy}}(w)$ via soft switch probability $p_{\text{gen}} \in [0, 1]$ by

$$p_t^{(w)} = p_{\text{gen}} P_{\text{vocab}}(w) + (1 - p_{\text{gen}}) P_{\text{copy}}(w)$$

and

$$P_{\text{copy}}(w) = \sum_{\{i:w_i=w\}} a_i^t$$

where a_i^t indicates the attention probability on i-th word of the input article. The whole network is then trained end-to-end with the negative log-likelihood loss function of

$$J_{\text{PG}}(\mathbf{x}, \mathbf{y}; \mathbf{\Theta}) = -\frac{1}{|y|} \sum_{t=1}^{|y|} \log(p_t^{(y_t)})$$

for a sample article-summary pair $(\mathbf{x}, \mathbf{y})$ where $\mathbf{\Theta}$ denote the learnable model parameters. It is important to note here that we do not use the coverage mechanism introduced by the original work [5] to prevent the potential repetition problem in the summaries generated by the model.

Transformer Network. For machine translation, we use the transformer network [117], which is a recent model that achieves state-of-the-art results on *WMT 2014 English-to-German* MT task with less computational time owing to its highly parallelizable architecture. The core idea behind this model is to use stacked self-attention mechanisms along with point-wise, fully connected layers for both encoder and decoder to represent its input and output. For the sake of brevity, we refer the reader to [117] for further details regard-

ing the architecture. Similar to previously defined loss functions, let $J_{\text{TF}}(\mathbf{x}, \mathbf{y}; \mathbf{\Theta})$ denote the per-example loss function of transformer networks for an input-output translation pair $(\mathbf{x}, \mathbf{y})$ where $\mathbf{\Theta}$ is again indicating the learnable model parameters.

6.3.2 Model Variants and Training

Let $\{(\mathbf{x}^{(l)}, \mathbf{y}^{(l)})\}_{l=1}^{N}$ denote the set of training examples, where $\mathbf{x}^{(l)}$'s are input sequences, and $\mathbf{y}^{(l)}$'s are their corresponding ground-truth output sequences. Before optimizing for the introduced objective J_{CALCS}, we first train the corresponding baseline network by minimizing

$$J_{\{\text{PG,TF}\}}(\mathbf{\Theta}) = \frac{1}{N} \sum_{l=1}^{N} J_{\{\text{PG,TF}\}}(\mathbf{x}, \mathbf{y}; \mathbf{\Theta}).$$

Unlike J_{CALCS}, loss functions $J_{\{\text{PG,TF}\}}$ for baseline models are computed by teacher forcing, feeding the previous ground-truth word at each decoding step. We will denote the baseline models by POINTGEN for pointer-generator and TRANSFORMER for transformer network.

To optimize for the proposed objective J_{CALCS}, we initialize the model parameters $\mathbf{\Theta}$ from the pre-trained baseline network and continue training the model by minimizing the joint loss

$$J(\mathbf{\Theta}) = \lambda J_{\text{CALCS}}(\mathbf{\Theta}) + (1 - \lambda) J_{\{\text{PG,TF}\}}(\mathbf{\Theta}) \tag{6.27}$$

$$J_{\text{CALCS}}(\mathbf{\Theta}) = \frac{1}{N} \sum_{l=1}^{N} J_{\text{CALCS}}(\mathbf{x}, \mathbf{y}; \mathbf{\Theta}) \tag{6.28}$$

where λ is a hyperparameter controlling the balance between the two losses. During the training with the joint loss, we compute $J_{\text{CALCS}}(\mathbf{x}, \mathbf{y}; \mathbf{\Theta})$, by performing $|y|$-many decoding steps as a simple strategy to prevent the model from gaming the training objective by generating longer and longer hypotheses instead of incurring an additional length penalty. We will refer to the resulting model trained with the loss function in Eq. 6.27 as {POINTGEN, TRANSFORMER}+CALCS depending on the baseline model.

Model	ROUGE-1	ROUGE-L
[88]	35.46	32.65
w/o coverage [5]	36.44	33.42
w/ coverage [5]	39.53	36.38
LEAD-3 baseline [5]	**40.34**	36.57
RL [125]	**41.16**	39.08
ML + RL [125]	39.87	36.90
Our Models		
PointGen*	39.11	26.97**
PointGen*+Ss	39.33	26.94**
PointGen*+Ss+CaLcs	**40.37**	**29.18****

Table 6.1: ROUGE F_1 results on *CNN/Daily Mail* summarization dataset. Our reimplementation of PointGen* corresponds to w/o coverage [5]. ** sign near ROUGE-L results reported for our models indicates a difference in our ROUGE-L evaluation as explained below.

6.4 Experiments

We numerically evaluate the proposed method on two sequence generation benchmarks: abstractive document-summarization and machine translation. We compare the results of the proposed method against the recently proposed strong baseline models [5] for summarization and and [117] for machine translation tasks. All models are implemented in Tensorflow-Lingvo [129].

6.4.1 Abstractive Summarization

We use a modified version of the CNN/Daily Mail dataset [128] that is first used for summarization by [88]. However, we follow the processing script provided by [5] to obtain *non-anonymized* version of the data that contains 287,226 training pairs, 13,368 validation pairs, and 11,490 test pairs of news articles (781 tokens on average) and their corresponding ground-truth summaries (56 tokens on average). We refer the reader to [5] for further details of the difference of their version from [88].

For training our baseline model, we use single layer LSTM encoder (bi-directional) and decoder with hidden dimensions of 512 and 1024, respectively. We use a vocabulary of 50k words for both source and target. Following the original paper, we also do not pre-train word embeddings, which are learned with the rest of model parameters during training. We use the Adam [50] optimizer with a learning rate of 0.00001 for training. We pre-train the baseline model for 20k steps by applying greedy scheduled sampling [130] with fixed ground-truth feeding probability of 75%. Once the baseline model training is complete, we start optimizing for CaLcs objective as described in the previous section. Also, we set $\lambda = 1.0$ and $\alpha = 1.0$, which are tuned on the development set.

In Table 6.1, we report our main results on the summarization task. PointGen+Ss refers to the baseline model trained with scheduled sampling. PointGen+Ss+CaLcs corresponds our model trained with CaLcs starting from PointGen+Ss model. Experimental results demonstrate that training with our proposed objective provides an improvement of 2.2 points in ROUGE-L score. This also provides empirical evidence to justify that our approximate CaLcs effectively captures what the original LCS metric is supposed to measure, recalling ROUGE-L is a normalized LCS. The reason why ROUGE-L scores of our models are lower than previously reported is that we evaluate ROUGE-L score by taking the entire summary as a single sequence instead of splitting it into sentences, which is also the way we compute CaLcs objective during the model training process. The main motivation behind this approach is to encourage the model to preserve the sentence order within a summary, and evaluate its performance in the same way. We consider the capability of preserving the order across produced sentences as an important attribute a multi-sentence summarization model should have in terms of readability and fluency of its generated summaries as a whole. When PointGen*+Ss and PointGen*+Ss+CaLcs are evaluated by splitting the generated summaries into sentences, their corresponding ROUGE-L scores become 35.38 and 35.12, respectively.

Model	BLEU
Gnmt [126]	24.61
Gnmt+RL [126]	24.60
Transformer [117]	27.3
Weighted Transformer [131]	28.4
Transformer*	27.6
Transformer*+CaLcs	**27.8**

Table 6.2: Machine translation results on *WMT 2014 English-to-German* task. Transformer* corresponds to our training of the original model in [117].

We also observe a nice side-improvement of 1.0 point in ROUGE-1 score over the baseline, which achieves a comparable performance with the long-overdue LEAD-3 baseline score. It might also be comparable to the recently reported state-of-the-art ROUGE-1 result on CNN/DailyMail dataset by [125] as they used a different dataset processing pipeline, which makes it difficult to directly compare with ours.

6.4.2 Machine Translation

We also evaluate our sequence-level training approach on the *WMT 2014 English-to-German* machine translation task, which contains 4.5M pairs of sentences.

To train our baseline transformer model, we closely follow the small model in the original transformer paper [117]. We use a vocabulary of size 32k. Our encoder and decoder consist of $N = 6$ identical layers each. Following the notation in the original paper, we set the other parameters as $d_{\text{model}} = 512$, $d_{\text{ff}} = 2048$, $h = 8$, $P_{\text{drop}} = 0.1$. We set $\lambda = 0.3$ and $\alpha = 1.0$, which are tuned on the development set.

In Table 6.2, we show our empirical results on machine translation task. Our first observation is that our trained baseline transformer network achieves a better performance than the one reported in the original paper [117] by 0.3 BLEU score, which might be solely due to hyperparameter tuning. More importantly, we observe that training

with our proposed CaLcs objective leads to noticeable 0.2 BLEU point improvements over the baseline, which further reinforces our confidence in effectiveness of our proposed sequence-level training approach and its applicability to other sequence prediction tasks. It is also interesting to note that optimizing for LCS metric via its continuous approximation leads to improvements in evaluation with another discrete metric BLEU. On the other had, optimizing for the exact discrete metric BLEU via reinforcement learning strategy may not improve the evaluation performance in BLEU as reported by [126]. As a final remark, we would like to note that our proposed approach is orthogonal to advancements in more expressive and powerful architecture designs. Hence it has the potential to provide further improvements over the recently proposed models such as Weighted Transformer [131].

6.5 Related Work

Text Summarization. Before the successful application of neural generative models, most of the existing works on text summarization [132, 133] have focused on extractive methods. While some of the early approaches have used a rich set of heuristic rules or sparse features to select textual units to include in the summary, more recent works [134, 135] leverage neural models to select words and sentences from the original text. With the emergence of sequence-to-sequence models [85] and large-scale datasets like *CNN/Daily Mail* [128, 88] and *NYT* [125], abstractive summarization of longer text have become a more feasible and popular task. Several recent approaches have been proposed to tackle abstractive summarization problem, where [88] exploits hierarchical encoders, [5] proposes *pointer-generator* network and coverage mechanism to overcome OOV and repetition problems, [89] introduces a graph-based attention mechanism and hierarchical beam search strategy, and [125] proposes to optimize for ROUGE metric via

reinforcement learning. Although impressive progress has been achieved for sentence-level summarization, efforts in document-level abstractive summarization are still in early stages where the simple LEAD-3 approach is still a very strong baseline for the task.

Neural Machine Translation. With the recent success of encoder-decoder architectures [85, 86], neural machine translation systems has gained a a lot of attention both from academia [91, 52, 136] and industry [126, 117, 131] over statistical machine translation, which has been the dominating translation paradigm for years. Most of these works has focused more on enhancing the architecture design aspect to tackle with various challenges such as different attention mechanisms [86, 52], a character-level decoder [137], a translation coverage mechanism [138], and so on. However, only very recently, a few works [126, 122, 139, 140, 141, 142, 143] have investigated sequence-level optimization by training to maximize BLEU score.

Neural Sequence Generation with RL. Most neural sequence generation models are trained with the objective of maximizing the probability of the next correct word. However, this results in a major discrepancy between training and test settings of these models because they are trained with cross-entropy loss at word-level, but evaluated based on sequence-level discrete metrics such as ROUGE [109] or BLEU [108]. On the other hand, directly optimizing for such evaluation metrics is hard due to non-differentiable nature of the exact objective [123]. Recent works [122, 126, 141, **?**] address the difficulty of differentiating with respect to rewards based on such discrete metrics using variants of reinforcement learning. These methods essentially propose to mitigate the problem by optimizing the reward weighted log-likelihood of the hypothesis sequences generated by the model distribution. In this work, we propose an alternative solution to tackle this problem by introducing a differentiable approximation to exact LCS metric that can be directly optimized by standard gradient-based methods without RL, while still addressing the *exposure bias* problem.

6.6 Conclusion

In this chapter, we explored an alternative approach for training text generation models with sequence-level optimization to combat *wrong objective* and *exposure bias* problems. We introduced a new objective function based on a continuous approximation of LCS metric that measures sequence-level structure similarity between sentences. We applied our proposed approach to *CNN/Daily Mail* dataset for long document summarization and *WMT 2014 English-to-German* machine translation task. By extending the objectives of strong neural baseline models with our proposed objective, we empirically demonstrated its effectiveness on these two tasks. Our proposed approach suggests a promising alternative to policy-gradient methods to side step the difficulty of differentiating with respect to reward function while directly optimizing for surrogate functions as approximations to sequence-level discrete metrics.

Chapter 7

Conclusion and Future Directions

7.1 Conclusion

In this dissertation, we first discussed one of the main obstacles for non-expert users to leverage the massive amount of digital data: the gap between natural language and the formal machine executable languages (e.g., SQL) required to even effectively access the data. We then investigated into natural language interfaces, an important and promising research direction for enabling human users to use natural language to interact with computers and data. Towards better understanding and tackling the central challenges that the modern NLIs face with, in this dissertation, we studied deep knowledge grounding for factual and conversational NLIs, and made key technical contributions to this field that were categorized into three main parts spanning five chapters: Facutal NLIs, Conversational NLIs, and Novel Objective for Improved Language Generation.

In Chapter 2, we started by a significant observation showing the potential performance improvement of predicting the answer type for factual questions over knowledge bases. Inspired by this observation, we proposed a neural answer type inference for factual NLIs over knowledge bases and demonstrated that it can significantly improve the

performance of the state-of-the-art systems. As a follow up study, in Chapter 3, we proposed a novel post-inspection component to cross-check the corresponding KB relations behind the predicted answers to identify potential inconsistencies, and showed that it could help models recover from their own mistakes. In Chapter 4, we presented an in-depth analysis towards better understanding the depth and kinds of language understanding capabilities required to solve current benchmark NLI tasks. The answers to the analyzed questions helped us gain insights about the directions we should explore in order to further improve the natural language to SQL translation accuracy. Motivated by this analysis, we then investigated alternative solutions to realize the potential ceiling performance on WikiSQL benchmark. We showed that our proposed solution could reach up to 88.6% condition accuracy, further justifying the value of the initial analysis.

In Chapter 5, we studied conversational NLIs that can generate responses to user queries in a multi-turn fashion. More specifically, we focused on addressing some commonly observed issues of model generated responses being short, dull, and too generic. We proposed and experimented with a series of response generation models that aim to serve in the general scenario where in addition to the dialogue context, relevant unstructured external knowledge in the form of text is also assumed to be available. Our main approach extends pointer-generator networks [5] by allowing the decoder to hierarchically attend and copy from external knowledge in addition to the dialogue context to make the response more informative and engaging. We demonstrated the effectiveness of the proposed approach by both automatic and human evaluation metrics.

In Chapter 6, we studied a more foundational line of research where we proposed a novel training objective for conditional language generation. We then demonstrated the effectiveness of the proposed objective on text summarization and machine translation, two of the most popular benchmark tasks for conditional language generation. This is an important contribution towards building the next generation NLIs and putting them in

real use because natural language generation is among the most crucial attributes that a modern NLI system should be equipped with to be able to interact with humans.

7.2 Future Directions

There are several exciting and promising directions to pursue for further bridging the gap between humans and intelligent agents. As a natural extension of the studies covered in this dissertation, here we discuss the following future directions that we plan to explore towards building more adaptive, knowledgeable, and flexible intelligent agents.

One of the most interesting and promising future directions is to adapt our proposed DeepCopy model to task-oriented dialog setting. More precisely, the goal is to develop a single model that takes conversation history and an external knowledge source as input and jointly produces both text response and action to be taken by the system via latent knowledge reasoning without intermediate symbolic states. Thus, the model must learn to reason on the provided structured knowledge source with weak supervision signal coming from the text generation and the action prediction tasks, hence removing the need for belief state annotations. Despite being ambitious, this direction has a great potential for replacing traditional task-oriented dialog systems designed as pipelines of several independently trained modules (e.g., user intent classifier, belief state tracker, dialog act predictor), which not only require collection of very costly human annotations but also suffer from error propagation across the modules. The proposed direction, on the other hand, has the promise to address both of these shortcomings of traditional approaches. Furthermore, reducing the cost of data collection and annotation naturally facilitates quicker adaptation to new emerging domains. Overall, this is an interesting and exciting direction that is worth investigating towards building more adaptive intelligent agents.

Another interesting future direction is to explore novel approaches to combine models developed for chit-chat and task-oriented dialog systems instead of treating them as separate paradigms. More precisely, rather than treating precise knowledge retrieval from structured data for dialogue response generation and holding a chit-chat conversation as separate tasks, it might be worth relaxing these definitions and designing more general end-to-end conversational agents that are capable of simultaneously consuming and harnessing large scales of heterogeneous knowledge sources including structured knowledge bases (e.g., Frebase and DBpedia), semi-structured web tables, and unstructured free-form text (e.g., news articles and product reviews). It is plausible to argue that this is one of the most essential capabilities that the next generation conversational agents must be equipped with. Furthermore, this goal can be seen as a natural extension of the previous direction that needs further research and discovery of novel approaches to accommodate reasoning over heterogeneous knowledge sources. A very simple approach to start the exploration with could be adding another attention hierarchy in DeepCopy model for knowledge sources. However, even this approach would require novel revisions to scale up to several knowledge sources of various types. Despite the challenges, it is a promising direction to explore for pushing the frontiers of dialogue research and developing more knowledgeable, flexible, next generation conversational agents.

Bibliography

[1] J. Berant and P. Liang, *Imitation learning of agenda-based semantic parsers*, *Transactions of the Association for Computational Linguistics (TACL)* (2015).

[2] S. Yavuz, I. Gur, Y. Su, and X. Yan, *Recovering question answering errors via query revision*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2017.

[3] W. Yih, M.-W. Chang, X. He, and J. Gao, *Semantic parsing via staged query graph generation: Question answering with knowledge base*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2015.

[4] S. Yavuz, I. Gur, Y. Su, and X. Yan, *What it takes to achieve 100% condition accuracy on WikiSQL*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2018.

[5] A. See, P. J. Liu, and C. D. Manning, *Get to the point: Summarization with pointer-generator networks*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

[6] M. Ghazvininejad, C. Brockett, M. Chang, B. Dolan, J. Gao, W. Yih, and M. Galley, *A knowledge-grounded neural conversation model*, in *AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[7] S. Zhang, E. Dinan, J. Urbanek, A. Szlam, D. Kiela, and J. Weston, *Personalizing dialogue agents: I have a dog, do you have pets too?*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018.

[8] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor, *Freebase: A collaboratively created graph database for structuring human knowledge*, in *ACM SIGMOD International Conference on Management of Data*, 2008.

[9] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives, *DBpedia: A nucleus for a web of open data*, in *International Semantic Web Conference (ISWC)*, 2007.

[10] F. M. Suchanek, G. Kasneci, and G. Weikum, *Yago: A core of semantic knowledge*, in *World Wide Web (WWW)*, 2007.

[11] M. Recasens, M. catherine De Marneffe, and C. Potts, *The life and death of discourse entities: Identifying singleton mentions*, *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)* (2013).

[12] A. Carlson, J. Betteridge, R. C. Wang, E. R. Hruschka, Jr., and T. M. Mitchell, *Coupled semi-supervised learning for information extraction*, in *ACM International Conference on Web Search and Data mining (WSDM)*, 2010.

[13] L. Yao, S. Riedel, and A. McCallum, *Unsupervised relation discovery with sense disambiguation*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2012.

[14] Y. Su, H. Liu, S. Yavuz, I. Gür, H. Sun, and X. Yan, *Global relation embedding for relation extraction*, in *The North American Chapter of the Association for Computational Linguistics (NAACL)*, 2018.

[15] T. Lin, Mausam, and O. Etzioni, *No noun phrase left behind: Detecting and typing unlinkable entities*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2012.

[16] X. Li and D. Roth, *Learning question classifiers*, in *International Conference on Computational Linguistics (COLING)*, 2002.

[17] J. Berant, A. Chou, R. Frostig, and P. Liang, *Semantic parsing on freebase from question-answer pairs.*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2013.

[18] T. Kwiatkowski, E. Choi, Y. Artzi, and L. Zettlemoyer, *Scaling semantic parsers with on-the-fly ontology matching*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2013.

[19] A. Bordes, S. Chopra, and J. Weston, *Question answering with subgraph embeddings*, *ArXiv* (2014).

[20] X. Yao and B. Van Durme, *Lean question answering over freebase from scratch*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2014.

[21] H. Bast and E. Haussmann, *More accurate question answering on freebase*, in *ACM International Conference on Information and Knowledge Management (CIKM)*, 2015.

[22] A. Lally, J. M. Prager, M. C. McCord, B. Boguraev, S. Patwardhan, J. Fan, P. Fodor, and J. Chu-Carroll, *Question analysis: How watson reads a clue*, *IBM Journal of Research and Development* (2012).

[23] K. Balog and R. Neumayer, *Hierarchical target type identification for entity-oriented queries*, in *ACM International Conference on Information and Knowledge Management (CIKM)*, 2012.

[24] H. Sun, H. Ma, W. Yih, C. Tsai, J. Liu, and M. Chang, *Open domain question answering via semantic enrichment*, in *World Wide Web (WWW)*, 2015.

[25] Y. Su, S. Yang, H. Sun, M. Srivatsa, S. Kase, M. Vanni, and X. Yan, *Exploiting relevance feedback in knowledge graph search*, in *ACM International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, 2015.

[26] L. Dong, F. Wei, H. Sun, M. Zhou, and K. Xu, *A hybrid neural model for type classification of entity mentions*, in *International Joint Conference on Artificial Intelligence (IJCAI)*, 2015.

[27] A. Graves, *Supervised Sequence Labelling with Recurrent Neural Networks*, pp. 5–13. Springer Berlin Heidelberg, 2012.

[28] T. Tieleman and G. E. Hinton, *Lecture 6.5 - RMSProp, COURSERA: Neural networks for machine learning*, *Technical Report* (2012).

[29] C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky, *The stanford corenlp natural language processing toolkit*, in *Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 2014.

[30] A. Bordes, N. Usunier, S. Chopra, and J. Weston, *Large-scale simple question answering with memory networks*, *ArXiv* (2015).

[31] J. Pennington, R. Socher, and C. D. Manning, *Glove: Global vectors for word representation*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2014.

[32] Theano Development Team, *Theano: A Python framework for fast computation of mathematical expressions*, *ArXiv* (2016).

[33] X. Yao and B. V. Durme, *Information extraction over structured data: Question answering with freebase*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2014.

[34] J. Berant and P. Liang, *Semantic parsing via paraphrasing*, *Annual Meeting of the Association for Computational Linguistics (ACL)* (2014).

[35] J. Bao, N. Duan, M. Zhou, and T. Zhao, *Knowledge-based question answering as machine translation*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2014.

[36] M. Yang, N. Duan, M. Zhou, and H. Rim, *Joint relational embeddings for knowledged-based question answering*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2014.

[37] L. Dong, F. Wei, M. Zhou, and K. Xu, *Question answering over freebase with multi-column convolutional neural networks*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2015.

[38] X. Yao, *Lean question answering over freebase from scratch*, in *The North American Chapter of the Association for Computational Linguistics (NAACL)*, 2015.

[39] S. Reddy, O. Täckström, M. Collins, T. Kwiatkowski, D. Das, M. Steedman, and M. Lapata, *Transforming Dependency Structures to Logical Forms for Semantic Parsing*, *Transactions of the Association for Computational Linguistics (TACL)* (2016).

[40] K. Xu, S. Reddy, Y. Feng, S. Huang, and D. Zhao, *Question answering on freebase via relation extraction and textual evidence*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[41] Y. Yang and M. Chang, *S-mart: Novel tree-based structure learning algorithms applied to entity linking*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2015.

[42] S. Reddy, M. Lapate, and M. Steedman, *Large scale semantic parsing without question-answer pairs*, *Transactions of the Association for Computational Linguistics (TACL)* (2014).

[43] E. Choi, T. Kwiatkowski, and L. Zettlemoyer, *Scalable semantic parsing with partial ontologies*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2015.

[44] K. Xu, Y. Feng, S. Huang, and D. Zhao, *Hybrid question answering over knowledge base and free text*, in *International Conference on Computational Linguistics (COLING)*, 2016.

[45] M. Tan, C. dos Santos, B. Xiang, and B. Zhou, *Improved representation learning for question answer matching*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[46] P. Neculoiu, M. Versteegh, and M. Rotaru, *Learning text similarity with siamese recurrent networks*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[47] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, *Teaching machines to read and comprehend*, in *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[48] D. Chen, J. Bolton, and C. D. Manning, *A thorough examination of the cnn/daily mail reading comprehension task*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[49] J. Mueller and A. Thyagarajan, *Siamese recurrent architectures for learning sentence similarity*, in *AAAI Conference on Artificial Intelligence (AAAI)*, 2016.

[50] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2015.

[51] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, *Tensorflow: A system for large-scale machine learning*, in *12th USENIX Symposium on Operating Systems Design and Implementation (OSDI 16)*, 2016.

[52] M.-T. Luong, H. Pham, and C. D. Manning, *Effective approaches to attention-based neural machine translation*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2015.

[53] S. Yavuz, I. Gur, Y. Su, M. Srivatsa, and X. Yan, *Improving semantic parsing via answer type inference*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2016.

[54] Y. Qiu and H. Frei, *Concept based query expansion*, 1993.

[55] M. Mitra, A. Singhal, and C. Buckley, *Improving automatic query expansion*, 1998.

[56] R. Navigli and P. Velardi, *An analysis of ontology-based query expansion strategies*, 2003.

[57] S. Riezler, A. Vasserman, I. Tsochantaridis, V. Mittal, and Y. Liu, *Statistical machine translation for query expansion in answer retrieval*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2007.

[58] H. Fang, *A re-examination of query expansion using lexical resources*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2008.

[59] A. Sordoni, Y. Bengio, and J.-Y. Nie, *Learning concept embeddings for query expansion by quantum entropy minimization*, in *AAAI Conference on Artificial Intelligence (AAAI)*, 2014.

[60] F. Diaz, B. Mitra, and N. Craswell, *Query expansion with locally-trained word embeddings*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[61] A. Téllez-Valero, M. Montes-y Gómez, L. Villaseñor-Pineda, and A. Peñas, *Improving question answering by combining multiple systems via answer validation*, 2008.

[62] A. Trischler, Z. Ye, X. Yuan, and K. Suleman, *Natural language comprehension with epireader*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2016.

[63] W. A. Woods, *Progress in natural language understanding: an application to lunar geology*, in *Proceedings of the American Federation of Information Processing Societies Conference*, 1973.

[64] I. Androutsopoulos, G. D. Ritchie, and P. Thanisch, *Natural language interfaces to databases–an introduction*, *Natural language engineering* **1** (1995), no. 1 29–81.

[65] A.-M. Popescu, O. Etzioni, and H. Kautz, *Towards a theory of natural language interfaces to databases*, in *Proceedings of the 8th international conference on Intelligent user interfaces*, pp. 149–157, ACM, 2003.

[66] V. Zhong, C. Xiong, and R. Socher, *Seq2sql: Generating structured queries from natural language using reinforcement learning*, *arXiv preprint arXiv:1709.00103* (2017).

[67] X. Xu, C. Liu, and D. Song, *Sqlnet: Generating structured queries from natural language without reinforcement learning*, *arXiv preprint arXiv:1711.04436*.

[68] L. Mou, Z. Lu, H. Li, and Z. Jin, *Coupling distributed and symbolic execution for natural language queries*, in *International Conference on Machine Learning (ICML)*, 2017.

[69] S. Iyer, I. Konstas, A. Cheung, J. Krishnamurthy, and L. Zettlemoyer, *Learning a neural semantic parser from user feedback*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

[70] O. Vinyals, M. Fortunato, and N. Jaitly, *Pointer networks*, in *Advances in Neural Information Processing Systems (NIPS)*.

[71] Y. Sun, D. Tang, N. Duan, J. Ji, G. Cao, X. Feng, B. Qin, T. Liu, and M. Zhou, *Semantic parsing with syntax- and table-aware sql generation*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018.

[72] T. Yu, Z. Li, Z. Zhang, R. Zhang, and D. Radev, *Typesql: Knowledge-based type-aware neural text-to-sql generation*, in *Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT)*, 2018.

[73] I. Gur, S. Yavuz, Y. Su, and X. Yan, *Dialsql: Dialogue based structured query generation*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2018.

[74] L. S. Zettlemoyer and M. Collins, *Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars*, 2005.

[75] R. J. Kate, Y. W. Wong, and R. J. Mooney, *Learning to transform natural to formal languages*, in *AAAI Conference on Artificial Intelligence (AAAI)*, 2005.

[76] L. Dong and M. Lapata, *Language to logical form with neural attention*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[77] H. Sun, H. Ma, X. He, W.-t. Yih, Y. Su, and X. Yan, *Table cell search for question answering*, in *World Wide Web (WWW)*, 2016.

[78] J. M. Zelle and M. Ray, *Learning to parse database queries using inductive logic programming*, in *Proceedings of the AAAI Conference on Artificial Intelligence*, 1996.

[79] L. Zettlemoyer and M. Collins, *Online learning of relaxed ccg grammars for parsing to logical form*, in *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, 2007.

[80] Y. Su, H. Sun, B. Sadler, M. Srivatsa, I. Gur, Z. Yan, and X. Yan, *On generating characteristic-rich question sets for qa evaluation*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2016.

[81] K. Xu, L. Wu, Z. Wang, and V. Sheinin, *Graph2seq: Graph to sequence learning with attention-based neural networks*, *arXiv preprint arXiv:1804.00823* (2018).

[82] K. Xu, L. Wu, Z. Wang, M. Yu, L. Chen, and V. Sheinin, *Exploiting rich syntactic information for semantic parsing with graph-to-sequence model*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2018.

[83] P. Pasupat and P. Liang, *Compositional semantic parsing on semi-structured tables*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2015.

[84] S. K. Jauhar, P. Turney, and E. Hovy, *Tables as semi-structured knowledge for question answering*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[85] I. Sutskever, O. Vinyals, and Q. V. Le, *Sequence to sequence learning with neural networks*, in *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[86] D. Bahdanau, K. Cho, and Y. Bengio, *Neural machine translation by jointly learning to align and translate*, 2015.

[87] M. A. Rush, S. Chopra, and J. Weston, *A neural attention model for abstractive sentence summarization*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2015.

[88] R. Nallapati, B. Zhou, C. d. Santos, C. Gulcehre, and B. Xiang, *Abstractive text summarization using sequence-to-sequence rnns and beyond*, 2016.

[89] J. Tan, X. Wan, and J. Xiao, *Abstractive document summarization with a graph-based attentional neural model*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

[90] S. Yavuz, C.-C. Chiu, P. Nguyen, and Y. Wu, *CaLcs: Continuously approximating longest common subsequence for sequence level optimization*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2018.

[91] K. Cho, B. V. Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H.-. Schwenk, and Y. Bengi, *Learning phrase representations using rnn encoder-decoder for statistical machine translation*, *arXiv preprint arXiv:1406.1078* (2014).

[92] M. Luong, H. Pham, and C. D. Manning, *Effective approaches to attention-based neural machine translation*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2015.

[93] C. Xiong, V. Zhong, and R. Socher, *Dynamic coattention networks for question answering*, 2017.

[94] I. V. Serban, A. Sordoni, Y. Bengio, A. C. Courville, and J. Pineau, *Building end-to-end dialogue systems using generative hierarchical neural network models.*, in *AAAI Conference on Artificial Intelligence (AAAI)*, 2016.

[95] O. Vinyals and Q. Le, *A neural conversational model*, *arXiv preprint arXiv:1506.05869* (2015).

[96] J. Li, M. Galley, C. Brockett, G. Spithourakis, J. Gao, and B. Dolan, *A persona-based neural conversation model*, in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 994–1003, Association for Computational Linguistics, 2016.

[97] A. Ritter, C. Cherry, and W. B. Dolan, *Data-driven response generation in social media*, in *Proceedings of the conference on empirical methods in natural language processing*, pp. 583–593, Association for Computational Linguistics, 2011.

[98] I. V. Serban, A. Sordoni, R. Lowe, L. Charlin, J. Pineau, A. C. Courville, and Y. Bengio, *A hierarchical latent variable encoder-decoder model for generating dialogues.*, in *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[99] S. Liu, H. Chen, Z. Ren, Y. Feng, Q. Liu, and D. Yin, *Knowledge diffusion for neural dialogue generation*, in *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 1489–1498, Association for Computational Linguistics, 2018.

[100] E. Dinan, S. Roller, K. Shuster, A. Fan, M. Auli, and J. Weston, *Wizard of wikipedia: Knowledge-powered conversational agents*, 2019.

[101] P. Budzianowski, T.-H. Wen, B.-H. Tseng, I. Casanueva, S. Ultes, O. Ramadan, and M. Gašić, *MultiWOZ - a large-scale multi-domain wizard-of-Oz dataset for task-oriented dialogue modelling*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2018.

[102] M. Eric, R. Goel, S. Paul, A. Sethi, S. Agarwal, S. Gao, and D. Hakkani-Tür, *Multiwoz 2.1: Multi-domain dialogue state corrections and state tracking baselines*, *CoRR* (2019) [arXiv:1907.0166].

[103] B. Byrne, K. Krishnamoorthi, C. Sankar, A. Neelakantan, D. Duckworth, S. Yavuz, B. Goodrich, A. Dubey, A. Cedilnik, and K.-Y. Kim, *Taskmaster-1: Toward a realistic and diverse dialog dataset*, 2019.

[104] D. Raghu, N. Gupta, and Mausam, *Hierarchical pointer-generator network for task oriented dialog*, *arXiv preprint arXiv:1805.01216* (2018).

[105] S. Sukhbaatar, A. Szlam, J. Weston, and F. Rob, *End-to-end memory networks*, in *Advances in Neural Information Processing Systems (NIPS)*, 2014.

[106] J. Weston, E. Dinan, and A. H. Miller, *Retrieve and refine: Improved sequence generation models for dialogue*, *arXiv preprint arXiv:1808.04776v2* (2018).

[107] B. Zoph and K. Knight, *Multi-source neural translation*, in *The North American Chapter of the Association for Computational Linguistics (NAACL)*, 2016.

[108] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, *Bleu: A method for automatic evaluation of machine translation*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2002.

[109] C. Lin and F. J. Och, *Automatic evaluation of machine translation quality using longest common subsequence and skip-bigram statistics*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2004.

[110] R. Vedantam, C. L. Zitnick, and D. Parikh, *Cider: Consensus-based image description evaluation*, *arXiv preprint arXiv:1411.5726* (2014).

[111] J. Li, M. Galley, J. Brockett, Chris ad Gao, and B. Dolan, *A diversity-promoting objective function for neural conversation models*, in *The North American Chapter of the Association for Computational Linguistics (NAACL)*, 2016.

[112] M. Galley, C. Brockett, X. Gao, B. Dolan, and J. Gao, "End-to-end conversation modeling: Moving beyond chitchat." `http://workshop.colips.org/dstc7/proposals/DSTC7-MSR_end2end.pdf`, 2018. Online; accessed 23 October 2018.

[113] S. Yavuz, A. Rastogi, G.-L. Chao, and D. Hakkani-Tur, *Deepcopy: Grounded response generation with hierarchical pointer networks*, in *Annual Meeting of the Special Interest Group on Discourse and Dialogue (SIGDIAL)*, 2019.

[114] S. Chopra, M. Auli, and M. A. Rush, *Abstractive sentence summarization with attentive recurrent neural networks.*, in *The North American Chapter of the Association for Computational Linguistics (NAACL)*, 2016.

[115] Y. Miao and P. Blunsom, *Discrete generative models for sentence compression*, in *Empirical Methods on Natural Language Processing (EMNLP)*, 2016.

[116] Q. Zhou, N. Yang, F. Wei, and M. Zhou, *Selective encoding for abstractive sentence summarization*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2017.

[117] A. Vaswani, S. Noam, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, *Attention is all you need*, in *Advances in Neural Information Processing Systems (NIPS)*, 2017.

[118] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, *BERT: Pre-training of deep bidirectional transformers for language understanding*, in *The North American Chapter of the Association for Computational Linguistics (NAACL)*, 2019.

[119] K. Song, X. Tan, T. Qin, J. Lu, and T.-Y. Liu, *Mass: Masked sequence to sequence pre-training for language generation*, *arXiv preprint arXiv:1905.02450* (2019).

[120] N. Arivazhagan, C. Cherry, W. Macherey, C.-C. Chiu, S. Yavuz, R. Pang, W. Li, and C. Raffel, *Monotonic infinite lookback attention for simultaneous machine translation*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2019.

[121] P. Koehn, F. J. Och, and D. Marcu, *Statistical phrase-based translation*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2003.

[122] M. Ranzato, S. Chopra, M. Auli, and W. Zaremba, *Sequence level training with recurrent neural networks*, 2016.

[123] A.-V. I. Rosti, B. Zhang, S. Matsoukas, and R. Schwartz, *Expected bleu training for graphs: Bbn system description for wmt11 system combination task*, in *Proceedings of the Sixth Workshop on Statistical Machine Translation*, 2011.

[124] R. J. Williams, *Simple statistical gradient-following algorithms for connectionist reinforcement learning*, *Machine Learning* (1992).

[125] R. Paulus, C. Xiong, and R. Socher, *A deep reinforced model for abstractive summarization*, 2018.

[126] Y. Wu, M. Schuster, Z. Chen, Q. V. Le, N. Mohammad, W. Macherey, M. Krikun, C. Yuan, Q. Gao, and e. a. Macherey, Klaus, *Googles neural machine translation system: Bridging the gap between human and machine translation*, *arXiv preprint arXiv:1609.08144* (2016).

[127] P. Henderson, R. Islam, P. Bachman, J. Pineau, D. Precup, and D. Meger, *Deep reinforcement learning that matters*, in *AAAI Conference on Artificial Intelligence (AAAI)*, 2018.

[128] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, *Teaching machines to read and comprehend*, in *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[129] J. Shen, P. Nguyen, Y. Wu, Z. Chen, *et. al.*, *Lingvo: a modular and scalable framework for sequence-to-sequence modeling*, 2019.

[130] S. Bengio, O. Vinyals, N. Jaitly, and N. Shazeer, *Scheduled sampling for sequence prediction with recurrent neural networks*, in *Advances in Neural Information Processing Systems (NIPS)*, 2015.

[131] K. Ahmed, N. S. Keskar, and R. Socher, *Weighted transformer network for machine translation*, 2018.

[132] B. Dorr, D. Zajic, and R. Schwartz, *Hedge trimmer: A parse-and-trim approach to headline generation*, in *Proceedings of the HLT-NAACL 03 on Text Summarization Workshop - Volume 5*, 2003.

[133] G. Durrett, T. Berg-Kirkpatrick, and D. Klein, *Learning-based single-document summarization with compression and anaphoricity constraints*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[134] J. Cheng and M. Lapata, *Neural summarization by extracting sentences and words*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[135] R. Nallapati, F. Zhai, and B. Zhou, *Summarunner: A recurrent neural network based sequence model for extractive summarization of documents*, in *AAAI Conference on Artificial Intelligence (AAAI)*, 2017.

[136] M. Luong and C. D. Manning, *A hybrid Word-Character approach to open vocabulary neural machine translation*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[137] J. Chung, K. Cho, , and Y. Bengio, *A character-level decoder without explicit segmentation for neural machine translation*, *arXiv preprint arXiv:1603.06147* (2016).

[138] Z. Tu, Z. Lu, Y. Liu, X. Liu, and H. Li, *Coverage-based neural machine translation*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[139] M. Norouzi, S. Bengio, Z. Chen, N. Jaitly, M. Schuster, Y. Wu, and D. Schuurmans, *Reward augmented maximum likelihood for neural structured prediction*, in *Advances in Neural Information Processing Systems (NIPS)*, 2016.

[140] S. Shen, Y. Cheng, Z. He, W. He, H. Wu, M. Sun, and Y. Liu, *Minimum risk training for neural machine translation*, in *Annual Meeting of the Association for Computational Linguistics (ACL)*, 2016.

[141] D. Bahdanau, P. Brakel, K. Xu, A. Goyal, R. Lowe, J. Pineau, A. C. Courville, and Y. Bengi, *An Actor-Critic algorithm for sequence prediction*, 2017.

[142] V. Zhukov and M. Kretov, *Differentiable lower bound for expected BLEU score*, in *NIPS Workshop on Conversational AI*, 2017.

[143] N. Casas, M. R. Costa-juss, and J. R. Fonollosa, *A differentiable bleu loss. analysis and first results*, *Workshop of the 6th International Conference on Learning Representations* (2018).