Example-Driven Question Answering

Di Wang

CMU-LTI-19-011
August 2019

School of Computer Science
Carnegie Mellon University
Pittsburgh, PA 15213

Thesis Committee:

Eric Nyberg (chair)	(Carnegie Mellon University)
Jaime Carbonell	(Carnegie Mellon University)
Graham Neubig	(Carnegie Mellon University)
Nebojsa Jojic	(Microsoft Research)

Submitted in partial fulfillment of the requirements
for the degree of Doctor of Philosophy.

Copyright © 2019 Di Wang

Keywords: question answering, recurrent neural networks

To my parents and grandparents.

Abstract

Open-domain question answering (QA) is an emerging information-seeking paradigm, which automatically generates accurate and concise answers to natural language questions from humans. It becomes one of the most natural and efficient ways to interact with the web and especially desirable in hands-free speech-enabled environments. Building QA systems, however, either has to rely on off-the-shelf natural language processing tools that are not optimized for the QA task or train domain-specific modules (e.g., question type classification) with annotated data. Additionally, optimizing QA systems with hand-crafted procedures or feature engineering is costly, time-consuming and laborious to transfer to new domains and languages.

This dissertation studies the idea of *example-driven* question answering, which focuses on learning to search, select, and generate answers to unseen questions solely by observing existing noisy question-answer examples along with text corpus or knowledge base. To achieve this goal, we developed novel neural network architectures throughout the QA pipeline, that can be trained directly from question-answer examples. First, we propose candidate retrieval models that can utilize noisy signals to produce dense indexing for text corpus and generate structured queries for knowledge graphs. Second, we developed generative relevance models that do not require annotated negative QA pairs and discriminative relevance models that can utilize pseudo negative examples. Third, we improved encoder-decoder models for response text generation which can accept external guidance for specific language style and topic. The integrated QA pipeline aims to generate answer-like embedding vectors to search, select the most relevant passages, and compose a natural-sounding response based on the selected passages.

This dissertation demonstrates the feasibility of creating open-domain example-driven QA pipelines based on neural networks without any feature engineering or dedicated manual annotations for each QA module. Experiments show our models achieve state-of-the-art or competitive performance on several real-world ranking and generation tasks on domains of QA and conversation generation. When applying in the TREC LiveQA competitions, our approach received the highest average scores among automatic systems in main tasks of 2015, 2016 and 2017, and the highest average score in the medical subtask of 2017.

Acknowledgments

First and foremost, I would like to sincerely thank my advisor Professor Eric Nyberg for his guidance, encouragement, and support. He generously shared with me his deep insights on the field of question answering, instilled in me the great importance of solving real-world challenges, and gave me helpful advice whenever needed. Without his tremendous support, this thesis would not have been possible. I am also very grateful to Dr. Nebojsa Jojic for welcoming me as an intern at Microsoft Research, and for countless enlightening discussions about machine learning and conversational agents. Furthermore, I am thankful to Professor Jaime Carbonell and Professor Graham Neubig for agreeing to be on my thesis committee, and for providing valuable feedback and suggestions.

I was fortunate to collaborate with many exceptional researchers during my PhD journey, including Eugene Agichten, Jun Araki, Leonid Boytsov, Chris Brockett, David Carmel, Christopher Cieri, Bill Dolan, Robert Frederking, Michel Galley, Jeff Gee, Alexander Hauptmann, Zhiting Hu, Nancy Ide, Bonnie E. John, Noriko Kando, Zhengzhong Liu, Xuezhe Ma, Teruko Mitamura, Alkesh Patel, Dan Pelleg, James Pustejovsky, Chunqi Shi, Shashank Srivastava, Keith Suderman, Marc Verhagen, William Wang Yang, Jonathan Wright, Chenyan Xiong, and Chunting Zhou. It was great to have your help on our projects and so many wonderful adventures together. Furthermore, I would like to gratefully acknowledge the support of the Tencent Fellowship, the Yahoo! Fellowship, and the Carnegie Mellon Presidential Fellowship.

During my years in graduate school, I am truly blessed for being surrounded by so many amazing people to whom I wish to express my gratitude: Abdelwahab Bourai, Kai-min Kevin Chang, Wei-Cheng Chang, Kay Chen, Yun-Nung Chen, Xinlei Chen, Xiang Chen, Linghao Cui, Zhuyun Dai, Zihang Dai, Desai Fan, Anhong Guo, Kun He, Ting-Yao Hu, Junjie Hu, Ting-Hao Huang, Xuan Hui, Ruby Hui, Lu Jiang, Guokun Lai, Bohan Li, Jiwei Li, Shuangshuang Li, Hanxiao Liu, Jingzhou Liu, Rui Liu, Wenhao Liu, Kaixin Ma, Collin McCormack, Salvador Medina, Cun Mu, Ni Song, Chenyu Wang, Haoming Wang, Haohan Wang, Ling Wang, Rui Wang, Xu Wang, Yun Wang, Yi-Chia Wang, Miaomiao Wen, Yuexin Wu, Qizhe Xie, Jiateng Xie, Keyang Xu, Ruochen Xu, Xiaohua Yan, Diyi Yang, Zi Yang, Yang Yi, Pengcheng Yin, Donghan Yu, Shoou-I Yu, Zhou Yu, Shikun Zhang, Wei Zhang, Huimin Zhao, Ran Zhao, Tiancheng Zhao, Guoqing Zheng, and Haiyi Zhu. I would like to especially thank Stacey Young, Jennifer Lucas, and all the wonderful administrative staff at LTI for making my academic life at CMU easier.

Finally, I am eternally grateful to my parents and my grandparents. Thank you for your unconditional and endless love, encouragement and inspiration.

Contents

1 Introduction **1**
1.1 Motivation 1
1.2 Thesis Statement and Contributions 3
1.3 Outline 4

2 Background **7**
2.1 Open-domain Question Answering 7
2.2 Recurrent Neural Networks 13

I Example-driven Candidate Retrieval **19**

3 Answer Candidate Retrieval from Text Corpus **21**
3.1 Introduction and Background 21
3.2 Approach 23
3.3 Experiments 26
3.4 Results and Analysis 27
3.5 Related Work 29
3.6 Conclusion 29

4 Answer Candidate Retrieval from Knowledge Graph **31**
4.1 Introduction 31
4.2 Related Work 32
4.3 Approach 34
4.4 Experiments 41
4.5 Conclusion 43

II Example-driven Relevance Modeling **45**

5 Answer Context Ranking **47**
5.1 Introduction 47
5.2 Related Work 48
5.3 Question Paraphrase Identification with Neural Dual Entailment 48

5.4 Experiments 51
5.5 Summary 52

6 Answer Passage Selection 53
6.1 Introduction 53
6.2 Related Work 54
6.3 Answer Sentence Selection with Stacked BLSTM 55
6.4 Answer Passage Selection with Attentional Encoder-Decoders 60
6.5 Summary 62

7 Application to TREC LiveQA (2015, 2016, 2017) 63
7.1 Introduction 63
7.2 System Architecture 65
7.3 Evaluation Results 69
7.4 Summary and Discussion 72

III Example-driven Response Generation 75

8 Controllable Response Generation 77
8.1 Introduction 77
8.2 Related Work 79
8.3 Decoding with Selective Sampling 79
8.4 Output style restriction using a small ‘scenting’ dataset 81
8.5 Restricting the Output Topic 82
8.6 Experiments 85
8.7 Summary 90

9 Conclusion and Future Work 91

A Example outputs from CMU-OAQA LiveQA system 93
A.1 Example: Bad Answer 93
A.2 Example: Fair Answer 94
A.3 Example: Good Answer 94
A.4 Example: Excellent Answer 95

B Chatbots Interview: bJFK vs bNixon 97

C Chatbots Interview: bStarWars vs bTrump vs bHillary 109

Bibliography 115

List of Figures

1.1 Workflow Diagram of the Example-driven Question Answering 6

2.1 An illustration of a stacked bidirectional LSTM network 15

3.1 An illustration of the unified dense indexing for QA. Diamond and round dots represent the mapped locations of questions and answers in the semantic space. During training, the blue arrow is the expected gradient direction for Q2, which move towards its corresponding answer and move away from those dots that are outside of the self-learning margin. 22
3.2 An illustration of triplet training issues on community QA pairs. Regarding the question q_i, the gradients from the positive (blue) and negative (red) pairs are combined (purple line) and lead to a negative match a_k. . . 24

4.1 An example knowledge subgraph describing "Bob Dylan" 34
4.2 The workflow of the proposed training process for our KG-QA system . . 35
4.3 Example N-Triples in DBpedia knowledge graph 36

5.1 Model Structure of our Neural Dual Entailment Model 49

6.1 An illustration of our QA sentence relevance model based on stacked BLSTM 55
6.2 An illustration of Answer Passage Selection with Attentional Encoder-Decoders 61

7.1 Architecture of the CMU-OAQA LiveQA system 65

8.1 A part of a Counting Grid trained on Twitter data and its use in providing topical hints in decoding. For the source sentence at the top, the decoder may produce the two target samples on the right, if the circled locations are used as a hint, or the two sentences at the bottom if the locations in the lower right are picked. 84

List of Tables

3.1 Overview of results on the community QA retrieval task. The last three rows are results from our method with different size of negative samples $\|\mathcal{T}\|$ used during training. 28
3.2 Overview of Recall@1000 results on retrieval from MS MARCO full corpus. Dense refer to our best performing model in Table 3.1. The combined search are generated by the method described in Section 3.2.3 28

4.1 Summary of the Discovered Template-Predicate Mappings 42
4.2 Summary of our and past results of the relation detection task on the SimpleQuestions benchmark. Note that the previous models were trained with the annotated training dataset while our models were bootstrapped from the unannotated community QA corpus. 43

5.1 Evaluation results on the Quora paraphrase identification dataset in terms of accuracy. 51

6.1 Example answer candidate sentences for the question “What sport does Jennifer Capriati play?” from TREC QA 2004 54
6.2 Overview of prior results on the answer sentence selection task 58
6.3 Overview of our results on the answer sentence selection task. 58
6.4 Overview of our results on the answer retrieval and ranking on Yahoo! Answer Manner Questions dataset. 60
6.5 Results on re-ranking submitted answers from TREC LiveQA 2015 62

7.1 Results on development dataset 69
7.2 Official TREC 2015 LiveQA track evaluation results. 70
7.3 Official TREC 2016 LiveQA track evaluation results. 70
7.4 Official TREC 2017 LiveQA track evaluation results. 71
7.5 First reply from Yahoo! Answers vs CMU-OAQA answers 72
7.6 Asker selected best answer from Yahoo! Answers vs CMU-OAQA answers 72

8.1 Example responses from our Star Wars, Hillary, Trump, and Kennedy bots with scented conversation models. 78
8.2 Ranking the true target answer among random answers on Yahoo! Answers test set. 87

8.3 Results of quality assessments with 5-scale mean opinion scores (MOS) and JFK style assessments with binary ratings. Style results are statistically significant compared to the *selective-sampling* by paired t-tests ($p < 0.5\%$). 89
8.4 The style and topic assessments (both binary) of three models with different personas and with restriction of specific target topic for each chat context. All style and topic results are statistically significant compared to the Base (*selective-sampling*) by paired t-tests with $p < 0.5\%$. 90

Chapter 1

Introduction

In Section 1.1 we motivate the need for example-driven question answering. The research hypothesis and an overview of our approach is summarized in Section 1.2. Finally, we outline the structure of this dissertation in Section 1.3.

1.1 Motivation

Open-domain Question Answering (QA) aims at automatically providing accurate and concise answers to natural language questions from users. When compared with traditional search engines, QA systems automate the process of constructing proper search queries and generating the answer from retrieved web pages. In hands-free environments, speech-enabled QA has become one of the most natural and efficient interaction paradigms to seek information.

However, building comprehensive open-domain QA systems is still not cost-efficient. For example, IBM Watson showed remarkable performance on the Jeopardy! quiz show, but it took many person-years to develop and optimize the DeepQA pipelines for that task (Ferrucci et al., 2010). Watson team trained and integrated hundreds of modules to extract the potential answer, recognize indicative signals, and aggregate evidence for the final prediction. Each module only focused on one or a few specific types of Jeopardy questions. Additionally, developing each module in different domains or languages requires hand-crafted procedures or feature engineering to extract different properties from question and candidate texts, which is costly, time-consuming and laborious.

QA components either have to rely on off-the-shelf Natural Language Processing (NLP) resources and tools that are not optimized for the QA task or train the component with specific annotation data. Although several datasets have been annotated and can be used for factoid QA, like question type classifiers (Li and Roth, 2002), answer sentence selection (Wang et al., 2007b) and answer span prediction (reading comprehension) (Rajpurkar et al., 2016), each dataset is still restricted regarding taxonomy, size, or language.

Meanwhile, a vast amount of previously-answered questions are already available on question-and-answer Internet forums and essentially cover all the popular knowledge topics. For example, Yahoo! Answers has received more than 300 million questions as of July 2012, and on average two new questions and six new answers were posted per second (Grace and Soo, 2013). Most question-and-answer Internet forums can provide data in the Q-A pairs format. Some forums also keep records of community-authored duplicated questions (Fader et al., 2013), i.e., the question groups in Figure 1.1. On the one hand, these data sources reflect real user intents and cover a wide variety of knowledge and language patterns; on the other hand, the data can no longer be well represented by hand-craft taxonomies, rules or patterns as in domain-specific QA systems.

Recently neural models have demonstrated great promise in many sequence modeling and generation tasks, including image captioning, machine translation, summarization, and conversation response generation. Derived from the encoder-decoder framework, neural conversation models treat respond generation as a source-target translation problem. However, although the well-trained models can generate natural responses in many cases, these models are still far from real use. The generated responses are mainly based on source-to-target language patterns and thus can not provide much helpful information. QA systems, on another hand, are mainly information-seeking oriented retrieval and extraction pipelines, which mostly didn't consider conversation context, user's intention, readability, or entertainment. These encouraging early successes and limitations of neural models and QA systems have motivated the research interest in the data-driven training of natural-sounding conversational question answering systems.

1.1.1 Towards Efficient Training for Question Answering

At its core, **example-driven** question answering aims to automatically learn to answer questions solely by observing existing noisy question-answer examples along with text corpus or knowledge base. For example, to train a system to answer new questions submitted to Internet-based community question-and-answer websites, such as Yahoo! Answers, an example-driven question answering system would take a large set of existing question-answer pairs, and learn models that can then search, select, and generate answers to unseen questions. The significant advantage of this paradigm is that building QA systems no longer depends on dedicated manual annotations for each QA pipeline module and domain. It also frees us from having to heuristically design and implement complex taxonomies and functions to cover questions in a variety of domains, types, and languages. Ultimately this will allows us to build QA systems in a cost and time efficient manner.

This thesis follows the line of research initiated by Lita (2006) called instance-based QA which clusters training questions according to multiple similarity metrics, and learns cluster-specific strategies for answer typing, query expansion and answer candidate extraction. We extended it by introducing generative neural models of query generation, relevance scoring and response generation without explicit question clustering. Our models also can be learned directly from the noisy large-scale community QA data instead of TREC-style factoid question-answer pairs.

With the help of recent advancements in neural network models for NLP, feature representations now can be extracted and learned by back-propagation directly and thus no longer rely on humans to handcraft features. In this dissertation, we study the potentials of using neural networks as a central building block for all QA modules. To that end, we introduce and evaluate several neural architectures to approach full QA pipelines, including candidate retrieval, answer ranking and response generation. We also believe that prototyping neural based QA components is critical for the next stage development of conversational AI, which potentially enables knowledge retrieval, machine reading, and response generation jointly optimized for real conversation goals in social media environments.

1.2 Thesis Statement and Contributions

> In this thesis, we demonstrate the feasibility of the example-driven question answering in which the fundamental components of a conversational answering agent can be trained directly from question-answer examples and achieve strong performance.

We argue that an optimal conversational answering agent should be able to:

- understand the query with background knowledge, generate answer-alike queries, and retrieve relevant passages or facts from the open-domain text corpus and knowledge base;
- read, rank and select helpful and informative content; and
- generate personalized conversational response conditioned on selected passages and current dialog session.

Towards this ultimate goal, this thesis centers around the development and experiment of training neural networks based models to search, rank, and answer without relying on dedicated human annotations or feature engineering. As depicted in the workflow diagram Figure 1.1, we propose three classes of new QA modules that are capable to extract training signals from large volumes of open-domain human-to-human online interactions, particularly from Internet forums. Specifically, we made the following contributions to advance the example-driven QA:

- Example-driven **candidate retrieval** for question answering.

  The lexical chasm between an input question and relevant answer make it challenging to discover relevant answer passages merely by searching with the question keywords. In this part of the thesis, we propose an example-driven dense indexing model that, by encoding the questions and answer passages into the same dense semantic space, can discover candidate answers that near to the question in the trained vector space. We found that designing a new robust objective function is needed to effectively learn such indexer from the noisy community Q-A data. The experimental results show that the indexer trained using our proposed objective function achieved 1.5-times higher recall on the community QA dataset and 6.49 points higher recall on MS MARCO dataset. As far as we are aware, this work presents the first effective neural dense indexer for the large-scale QA retrieval task.

We also propose a model that learns to map natural language question to structured query for knowledge graphs. Our approach automatically discover potential question-to-query mappings from community QA data, and use them for training. We introduce a training strategy that denoises the extracted training instance. We demonstrate significant cumulative improvements correlated with each denoising step and achieves strong performance for the relation prediction task on the SimpleQuestions dataset without using any annotated training data.

- Example-driven **relevance modeling** for question answering. Answer context matching and answer passage selections modules used to build on top of extensive feature engineering or external knowledge. In this part of the thesis, we proposed neural architectures to modeling relevance without feature engineering. We proposed discriminative models that can be trained with automatically sampled negative examples and generative ranking models trained from noisy question-answer examples. Furthermore, when applying in TREC LiveQA competitions, our approach received the highest average scores among automatic systems in main tasks of 2015, 2016 and 2017, and the highest average score in the medical subtask of 2017.
- Example-driven **response generation** for question answering. Merely using retrieved text or entity as the answer has limited abilities to provide summarized comprehensive information in a natural-sounding way. In this part of the thesis, we proposed neural methods to generate the response that bias the generation process with a specific language style restriction, or a topic restriction. We decompose the neural generation process into empirically easier sub-problems: a faithfulness model and a decoding method based on selective-sampling. Human evaluation results show that our proposed methods are able to restrict style and topic without degrading output quality in conversational tasks.

1.3 Outline

The rest of this dissertation is organized as below.

- Chapter 2: Background

  This chapter gives an overview of canonical QA pipelines and fundamentals about recurrent neural networks.

- Chapter 3 and Chapter 4: Query Generation and Candidate Retrieval

  These chapters present candidate retrieval methods that could be directly trained out of question-answer examples targeting structured and unstructured knowledge sources. We present experiments of the proposed methods for dense indexing for text corpus and structured query generation for knowledge graphs.

- Chapter 5 and Chapter 6: Relevance Modeling

  These chapters introduce example-driven models to addresses two main kinds of relevance ranking in question answering, explicitly scoring answer candidates base on its context text (e.g. title of existing online discussion) and base on the answer passage itself.

- Chapter 7: Application to LiveQA

  We applied our relevance modeling models in the TREC LiveQA track 2015, 2016, and 2017. This chapter presents our LiveQA system architecture along with its intrinsic and extrinsic evaluation results.

- Chapter 8: Controllable Response Generation

  This chapter describes our neural encoder-decoder based approaches to generate plausible natural language response text, whose topic and language style are dynamically controlled.

- Chapter 9: Conclusion

  This chapter concludes this dissertation and lays out interesting directions for future work.

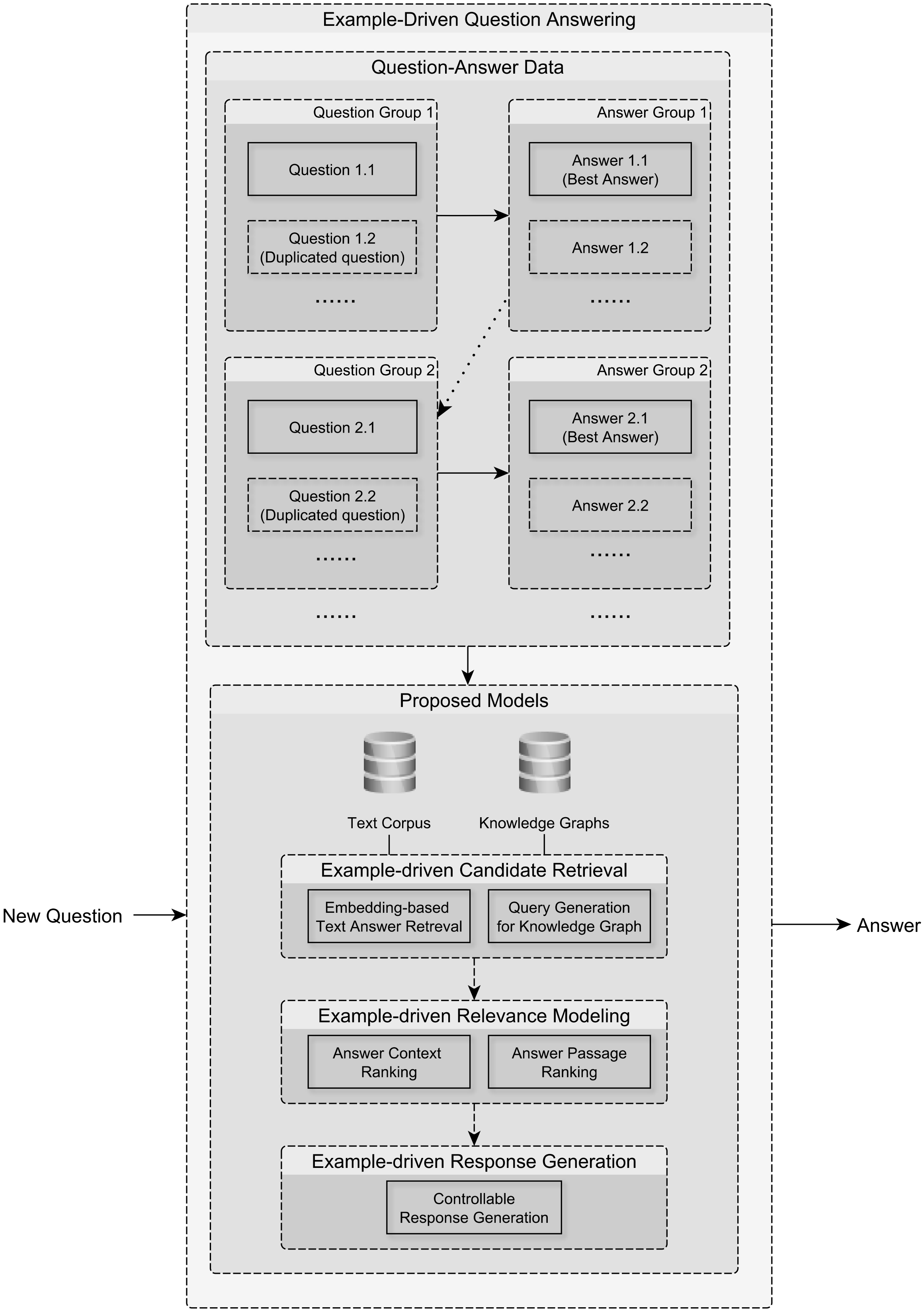


Figure 1.1: Workflow Diagram of the Example-driven Question Answering

Chapter 2

Background

In this chapter, I will first review the background knowledge on question answering systems in Section 2.1, and then introduce the model fundamentals of recurrent neural networks in Section 2.2 which will be used in the later chapters.

2.1 Open-domain Question Answering

The task of open-domain QA is to find accurate and concise answers (usually in the form of phrases or sentences) to natural language questions. Factoid QA systems address questions that ask for concise facts. For example, a factoid QA system can accept natural language questions like "Who is the first president of United States?", and attempt to output the exact phrases "George Washington" or answer sentences such as "George Washington was the first President of the United States from 1789 to 1797", based on open-domain knowledge sources such as Wikipedia or Wikidata. In contrast, a non-factoid QA system handles all type of questions, which could be as varied as "Pregnant cat? What to do?!" and "How do I convince my mom to pay for my gym?".

Early History The study of question answering systems has started since the 1960s. Several expert systems were built for restricted domains based on manually constructed knowledge graphs, such as BASEBALL (Green et al., 1961), LUNAR (Woods et al., 1972), and UNIX Consultant (Wilensky et al., 1988). The MURAX system (Kupiec, 1993, 1999) was a forerunner of the QA approach that processes open-domain questions, searches Grolier's online encyclopedia, and extracts noun-phrase answers with regular expressions. Katz (1997) created the first Web-based QA system START (Katz, 1997) shortly after the emergence of the World Wide Web.

QA research has gained more popularity after the inclusion of the QA tracks at the Text Retrieval Conference (TREC) (Voorhees, 2000; Agichtein et al., 2015)[1], NTCIR Confer-

---

[1] The TREC QA tracks were supported by the ARDA Advanced Question and Answering for Intelligence (AQUAINT) initiative. `https://www-nlpir.nist.gov/projects/aquaint/`

ence (Fukumoto and Kato, 2001; Shibuki et al., 2014) and Cross-Language Evaluation Forum (CLEF) (Magnini et al., 2004), which enables QA researchers around the world to evaluate and compare algorithms with common benchmark suites.

Typical Software Architecture A typical simplified architecture of open-domain QA systems is composed of three high-level major steps (Prager, 2006): (a) question analysis and retrieval of candidate clues and passages; (b) ranking and selecting of clues and passages which contain the answer; and optionally (c) extracting, verifying, and synthesizing the final answer. Full-fledged QA systems often make use of multiple strategies (Nyberg et al., 2003) to find answers and utilize as many types of resources as possible (Ferrucci, 2012). To that end, the algorithms and models in QA systems also need to be modular (Nyberg et al., 2004) and reuseable (Ferrucci and Lally, 2004; Ide et al., 2016).

High-performance open-domain QA systems require access to large-scale information sources that cover all the general knowledge topics. The resource can be a large corpus of textual information (sometimes the Web itself) and or heterogeneous structured knowledge graphs. QA based on text corpus extracts answers as text snippets from one or more semi-structured text documents (e.g. Wikipedia pages and or discussion forum threads). On another hand, QA based on knowledge-graph fetches answer entities from the knowledge-graph index with structured queries.

2.1.1 Candidate Retrieval

Question Analysis

The question analysis component is commonly the first step of all QA systems. The ultimate goal of this step is to interpret the semantic meaning of the user's inquiry from plain-text questions. Intuitively, errors in the question analysis step can cascade in the later stages of the QA pipeline. Therefore the end-to-end performance of a QA system highly depends on the quality of its question analysis module.

To produce richer features for analyzing questions, researchers in the QA community already introduced a wide range of NLP technologies to the question analysis task, including Named Entity Recognition (NER) (McNamee et al., 2008), syntactic and semantic parsing (Hermjakob, 2001), coreference resolution (Morton, 1999), semantic role labeling (Shen and Lapata, 2007). Additionally, new features also were explicitly created for better interpreting question intention, such as question classification (Li and Roth, 2002).

Text-based QA systems can parse the input question and generate query keywords for the passage retrieval module. Given the question "Who is the wife of Bill Clinton", if the IR module wishes to retrieve passages like "Hillary Rodham married Bill Clinton at their home in Fayetteville, Arkansas.", the question analysis component is responsible for feeding IR module with query keyword extension *(wife OR married OR spouse) AND*

Bill Clinton.

To help increase the recall rate for relevant documents, many kinds of query expansion or reformulation methods have been introduced to the QA field, which is vital for questions with few available extracted keywords to feed to the IR module. The most straightforward would be expansion using morphology derivations or synonymy from WordNet (Miller, 1995a). Hovy et al. (2000) and Greenwood (2004) have shown improvements in IR quality with the restricted use of WordNet resources. Pasca and Harabagiu (2001) manually maintained a rule-based method for inserting morphology and lexical derivations of the original keyword. Lin and Pantel (2002) present an approach in which questions are syntactically paraphrased to more flexible answer matching. Similarly, Hermjakob and Echihabi (2002) reformulate the questions with semantic paraphrases, called phrasal synonyms, to enhance retrieval performance. Negri and Kouylekov (2009) studied how to account for syntactic and semantic correspondences between questions and passages, by computing similarities between their respective syntactic trees. Therefore, those query expansion methods attempt to extend the original query by different expressions with similar meaning in the given context.

Answer Candidates Retrieval from Text

Passage retrieval is an information retrieval method that matches highly relevant paragraphs to the input query. Passage retrieval is widely used in text-based QA systems to discover answer bearing text passages. Salton et al. (1993) first proposed the concept of passage retrieval which suggested applying a specific strategy to identify that two text fragments are similar to each other. Passage retrieval extents document retrieval and further locates the relevant text snippets. One group of approaches extracts passages by predefined segments (e.g. sliding windows of a fixed number of sentences) and index them as pseudo-documents. Another group of approaches retrieves documents first with the help of standard document search tools such as Lucene or web search engines, then identify most relevant passages within top-ranked documents.

The passages themselves also vary in sizes and degrees of overlapping context. One critical challenge in passage retrieval is to identify the boundaries of coherent passages relevant to the query. Kaszkiel and Zobel (2001) reviewed and evaluated three segmentation strategies for passage extraction: discourse-based (using document structure such as `PARAGRAPHS`), semantic-based (using topic structure), and window-based passages. They found passage-based rankings are more effective than whole documents, but no single passage type showed superior retrieval performance than others. Jiang and Zhai (2006) argued that pre-segmenting documents into fixed-length passages were unlikely optimal because the optimal span of a relevant passage is presumably highly query-dependent.

Community Question Answering (CQA) makes use of collective human intelligence to resolve questions. Based on the observation that same question might repeatedly be asked by different users, preparing a list of frequently asked questions (FAQ) with answers has been a convenient way to share knowledge since the early days of the Internet. Community QA services such as Yahoo! Answers, Quora, and Stack Overflow provided large-scale knowledge sharing and collaborative learning platforms and accumulated large volume of user-generated knowledge over the last decades. Since the resources in CQA sites is in natural language question-answer format, when users ask full-sentence queries, text search engines can return previously answered questions from CQA site directly (Liu et al., 2011). Si and Chang (2010) reported that 25% of Google's (China) first-page search results contain at least one link to Q&A sites.

Research carried on CQA field range from question intention analysis, question retrieval, quality estimation, and user analysis. Liu et al. (2005) and Jurczyk and Agichtein (2007) investigated ways of finding experts from users behavior in the community, which enable forwarding unanswered question to certain experts in the community. Li et al. (2008) and Bian et al. (2008) tried automatically identifying whether a question is personal or subjective. Xue et al. (2008) and Jeon et al. (2005) investigated combinations of translation and retrieval models to retrieve similar questions and potential reusable answers to the new questions. Wang and Ming (2009) reported that the syntactic tree based question comparison achieves higher performance than bag-of-word or tree kernel based. Cao and Cong (2010) found the category information can improve CQA retrieval performance.

Related Evaluation Text REtrieval Conferences (TRECs), organized by the U.S. National Institute of Standards and Technology (NIST), has organized large-scale evaluations for QA systems annually between 1999 and 2007. Each year, the QA task involves a new set of questions and answer nuggets that can be found in corpus likes AQUAINT which contains approximately 1 million articles in 3 gigabytes of text. Documents in those corpus covering archives such as Financial Times Limited, Congressional Record of the 103rd Congress, the Federal Register, Associated Press World stream News, New York Times News, and the English portion of the Xinhua News. All those answers submitted to the evaluation and judged correct by evaluators can be also used to train and evaluate other QA systems. Similar to TRECs, the Cross-Language Evaluation Forum (CLEF) in Europe organized evaluations in monolingual and cross-lingual QA since 2003.

Answer Candidates Retrieval from Knowledge Graphs

Semantic Web (Berners-Lee et al., 2001) has provided a distributed platform for users to generate and share knowledge. In recent years, large-scale structured knowledge graphs, such as those in Linked Open Data (LOD) (Bizer et al., 2009a), already contain billions of entries in Resource Description Framework (RDF) (Manola and Miller, 2004) formats and act as a rich source of information for various user needs. To bring the value of these data to casual users and further answer their questions, natural language interfaces to access those knowledge databases are highly demanded.

In knowledge-graph QA systems, information is stored in structured databases with certain schema and is only accessible by structured query corresponding to that schema. Therefore, one of the core challenges is to parse and understand the natural language input and map it to structured query in a format such as SPARQL (Prud'hommeaux and Seaborne, 2008). Early knowledge-graph QA systems required controlled natural languages (CNLs) as input to enable accurate mapping from question to predefined terminology and schema of data source restricted to closed domains (e.g. a geography database). However, most large-scale open-domain knowledge graphs are community-driven such as DBpedia (Bizer et al., 2009b) and Wikidata (Vrandečić and Krötzsch, 2014). The problems of knowledge sparseness, duplicity and incompleteness extensively exist in those knowledge-graphs. Therefore, open-domain knowledge-graph QA needs more robust question analysis to generate structured queries and answer selection to validate and choose the results returned from multiple queries.

The research on accessing structured database with natural language was started more than forty years. Since the 1970s, Woods et al. built one of the world's first natural language question answering systems (LUNAR) (Woods et al., 1972) for two databases about chemical analyzes and the literature references of moon rocks. The translation from the question in English to its database query language was done by a rule-driven ATN parser (Woods, 1970). Soon after, several other systems with larger databases or larger vocabulary were developed such as LADDER (Hendrix et al., 1978) and TEAM (Martin et al., 1986). Zelle and Mooney (1993, 1996) investigated learning approaches, specifically learning to parse into Prolog queries. These work of Mooney focused on applying Inductive Logic Programming to generate semantic grammar from examples in certain domains. Popescu et al. invented PRECISE (Popescu et al., 2003, 2004) which guaranteed the correctness of its output by identifying natural language query that it can not fully understand.

Recently, the field of knowledge-graph QA focused on the large number of newly available ontology data (Guarino and Giaretta, 1995). The task becomes translating a natural language question into graph matching queries for retrieval knowledge statements. ORAKEL (Cimiano et al., 2007, 2008) system introduced by Cimiano et al. rigorously parses input using lambda calculus. AquaLog (Lopez et al., 2007) and its successor PowerAqua (Lopez and Fernndez, 2011) map question into sets of triple text segments, match to most similar ontology entities and then generate triple queries to find the answer. Querix (Kaufmann et al., 2006) and PANTO (Wang et al., 2007a) are implemented with similar approaches as AquaLog, and cover geography, restaurants and jobs domains. Damljanovic et al. (2011) designed FREyA system which emphasizes user interaction and feedback. However, these ontology-based QA systems are both close-domain and strongly reply on vocabularies derived from ontology nodes and WordNet extension.

2.1.2 Relevance Modeling

Passage retrieval algorithms can rank passage according to keywords frequencies as in MITRE (Light et al., 2002) and keywords densities as in MultiText (Cormack et al., 2000), IBM (Ittycheriah et al., 2001) and Site Q (Lee et al., 2002). Language models also can be used to retrieval and rank passage (Liu and Croft, 2002). Mittendorf and Schauble (1994) considered passage retrieval as a stochastic process, which generates text fragments independent of any particular query. Hidden Markov Models (HMMs) was used to model these stochastic processes. Melucci (1998) presented a Bayesian framework to estimate the concentrations of query terms in the text and extract passages based on term concentration weights. Because passages are considerably shorter than the whole document, term frequencies are close in relevant and many irrelevant passages. Tellex et al. (2003) found that scoring passage requires not only lexical matching but also word relation modeling to reduce false positive. Cui et al. (2005a) proposed a statistical word relation matching method that examines dependency parsing paths within both question and candidate sentences. It measures the matchings of dependency parsing relations in candidate sentences with their corresponding relations in the question. Bilotti et al. (2007) archived higher retrieval precision with a structured search that matching semantic role labeling structures between the question and candidate sentences.

Beyond keyword similarity, several researchers have tried pre-processing the corpus and matching question focus (e.g. “where”) to structurally annotated entitie (Lin and Katz, 2003; Bilotti et al., 2007), and the predicate annotation (Prager et al., 2000). Riezler et al. (2007) attempt to create query expansions by n-best phrase-level translation-based methods trained from crawled FAQ pairs. Derczynski et al. (2008) tried identifying the difficulty of input questions and then inserting extra search keywords for the difficult question, which discovered extensions from previous QA evaluations’ answer text. Komiya et al. (2013) find expansions by calculating mutual information from community question answering data. Monz (2003) finds a negative result when applying blind pseudo relevance feedback for QA in TREC 9, 10 and 11. Lita and Carbonell (2008) first cluster questions and provide cluster-specific query expansion. Bernhard and Gurevych (2009) use lexical semantic resources such as Wiki Answer, glosses and Wikipedia for training translation models. Zhang et al. (2015a) derived the paraphrases of question’s key concepts by pivot translation between another language (e.g. Chinese). Park and Croft (2015) proposed features and training approaches to identify key concepts in questions (e.g. named entities) and using a translation model for the rest of the question. Khashabi et al. (2017) introduced the concept of essential question terms and an annotated dataset with these terms. Their experiments have shown that those terms have larger impacts on human understanding and can improve the QA system by incorporating this notion.

2.1.3 Response Generation

The last stage of text-based questions answering is to generate a concise answer from the candidate answer-bearing passages. For factoid questions, a QA system can return a phrase, a paragraph, or a document. Lin et al. (2003) analyzed different lengths of answer context for factoid QA; they found that users prefer paragraphs as answers rather than phrases or documents. For non-factoid questions, a comprehensive answer usually means long paragraphs due to the complexity of the question.

Several families of algorithms have been developed to exacting exact phrases as the answer. For example, returning the user with an answer "1969" to the question "What year did the first man land on the moon?". One class of approaches extract answer by answer-type specific template patterns. Extraction templates can be composed with texts, POS tags, syntactic parsing, named entity classes, and placeholders. For example, Ravichandran and Hovy (2002) introduced an algorithm that automatically discover patterns (e.g. `<NAME> was born on <ANSWER> ,`) from online documents with seed question and answer anchors. There is also a class of approaches that explored web redundancy for answer extraction and validation such as the AskMSR system (Brill et al., 2002) and MultiText QA system Clarke et al. (2001). Those algorithms mine n-grams from retrieved passages and rank them based on their popularities and the predicted answer type. Supervised models were also developed using previous TREC question-answer as the training dataset. Yao et al. (2013) treats answer extraction as sequence labeling problem and employed linear chain CRFs with features including POS-tags, dependency parsing paths., named entities, and tree edit distances. Sultan et al. (2016) proposed jointly optimized probabilistic models for both answer sentence ranking and answer phrase extraction tasks by exploiting the correlation of the two tasks.

For improving machine comprehension and reasoning, answer extraction is also studied in reading comprehension based question answering setup. The questions and answers are formulated based on a given text passage, and the system is expected to provide an answer span based on its understanding of the given texts. Hermann et al. (2015) created a cloze-style dataset using CNN and Daily Mail news articles. Rajpurkar et al. (2016) released the SQuAD dataset where the questions are generated by the crowd using a given Wikipedia passage.

2.2 Recurrent Neural Networks

RNN is an extension of the conventional feed-forward neural network, used to deal with variable-length sequence input. It uses a recurrent hidden state whose activation is dependent on that of the one immediately before. More formally, given an input sequence $x = (x_1, x_2, \dots, x_T)$, a conventional RNN updates the hidden vector sequence $h = (h_1, h_2, \dots, h_T)$ and output vector sequence $y = (y_1, y_2, \dots, y_T)$ from $t = 1$ to T as follows:

$$h_t = \mathcal{H}(W_{xh}x_t + W_{hh}h_{t-1} + b_h) \tag{2.1}$$

$$y_t = W_{hy}h_t + b_y \tag{2.2}$$

where the W denotes weight matrices, the b denotes bias vectors and $\mathcal{H}(\cdot)$ is the recurrent hidden layer function.

Long Short-Term Memory (LSTM) Due to the vanishing gradient problem, conventional RNNs is found difficult to be trained to exploit long-range dependencies. In order to mitigate this weak point in conventional RNNs, specially designed activation functions have been introduced. LSTM is one of the earliest attempts and still a popular option to tackle this problem. LSTM cell was originally proposed by Hochreiter and Schmidhuber (1997). Several minor modifications have been made to the original LSTM cell since then. In our approach, we adopted a slightly modified implementation of LSTM in (Graves, 2013).

In the LSTM architecture, there are three gates (input i, forget f and output o), and a cell memory activation vector c. The vector formulas for recurrent hidden layer function $\mathcal{H}$ in this version of the LSTM network are implemented as the following:

$$i_t = \sigma(W_{xi}x_t + W_{hi}h_{t-1} + b_i) \tag{2.3}$$

$$f_t = \sigma(W_{xf}x_t + W_{hf}h_{t-1} + b_f) \tag{2.4}$$

$$c_t = f_t c_{t-1} + i_t \tau(W_{xc}x_t + W_{hc}h_{t-1} + b_c) \tag{2.5}$$

$$o_t = \sigma(W_{xo}x_t + W_{ho}h_{t-1} + b_o) \tag{2.6}$$

$$h_t = o_t \theta(c_t) \tag{2.7}$$

where, τ and θ are the cell input and cell output non-linear activation functions which are stated as tanh in our implementation.

LSTM uses input and output gates to control the flow of information through the cell. The input gate should be kept sufficiently active to allow the signals in. The same rule applies to the output gate. The forget gate is used to reset the cell's own state. In (Graves, 2013), peephole connections are usually used to connect gates to the cell in tasks requiring precise timing and counting of the internal states. In our approach, we don't use peephole connections because the precise timing does not seem to be required.

Bidirectional RNNs Another weak point of conventional RNNs is their utilization of only previous context with no exploitation of future context. Unlike conventional RNNs, bidirectional RNNs utilize both the previous and future context, by processing the data from two directions with two separate hidden layers. One layer processes the input sequence in the forward direction, while the other processes the input in the reverse direction. The output of current time step is then generated by combining both layers' hidden vector $\overrightarrow{h_t}$ and $\overleftarrow{h_t}$ by concatenation or weighted sum: $y_t = W_{\overrightarrow{h}y}\overrightarrow{h_t} + W_{\overleftarrow{h}y}\overleftarrow{h_t} + b_y$.

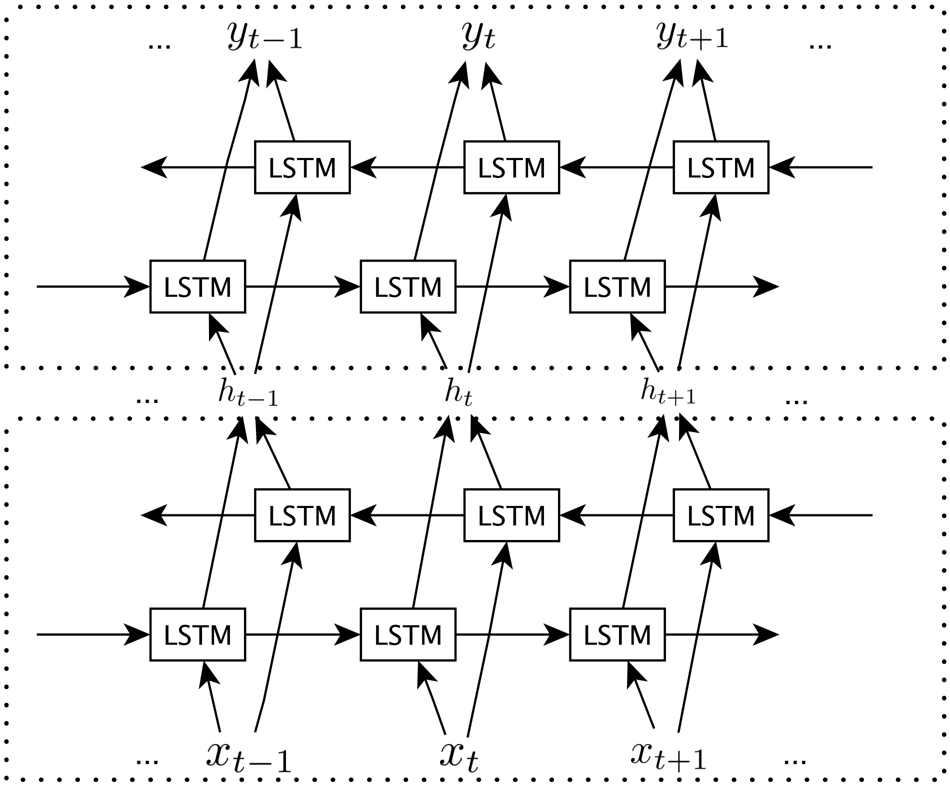


Figure 2.1: An illustration of a stacked bidirectional LSTM network

Multi-layer Stacked RNNs In a stacked RNN, the output h_t from the lower layer becomes the input of the upper layer. Through the multi-layer stacked network, it is possible to achieve different levels of abstraction from multiple network layers. There are theoretical supports indicating that a deep, hierarchical models can be more efficient in representing some functions than a shallow ones (Bengio, 2009). Empirical performance improvement is also observed in LSTM network compared with the shallow network (Graves et al., 2013).

2.2.1 Attentional Neural Encoder-Decoder

A basic encoder-decoder neural translation model from (Sutskever et al., 2014) suffers from translating a long source sequence efficiently. This is largely due to the fact that the encoder of this basic approach needs to compress the whole source sequence's information into a vector of fixed dimensionality. Motivated by this, the attention mechanism enables the decoder to revisits the input sequence's hidden states and dynamically collects information needed for each decoding step. Specifically, our relevance models are based on a combination of the models of (Bahdanau et al., 2014) and (Luong et al., 2015) that we found to be effective. Here we describe the attention-based neural encoder-decoder model we used for ranking in greater detail.

2.2.2 Model Structure

In general, our neural encoder-decoder model aims at generating a target sequence $Y = \left(y_1, \ldots, y_{T_y}\right)$ given a source sequence $X = (x_1, \ldots, x_{T_x})$. Each word in both source and target sentences, x_t or y_t, belongs to the source vocabulary V_x, and the target vocabulary V_y respectively.

First, an encoder converts the source sequence X into a set of context vectors $C = \{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_{T_x}\}$, whose size varies with regard to the length of the source passage. This context representation is generated using a multi-layered recurrent neural network (RNN). The encoder RNN reads the source passage from the first token until the last one, where

$$\mathbf{h}_i = \Psi\left(\mathbf{h}_{i-1}, \mathbf{E}_x\left[x_t\right]\right). \tag{2.8}$$

Here $\mathbf{E}_x \in \mathbb{R}^{|V_x| \times d}$ is an embedding matrix containing vector representations of words, and Ψ is a recurrent activation unit that we used the Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) in our submission.

The decoder, which is implemented as an RNN as well, generates one word at a time, based on the context vectors set returned by the encoder. The decoder's hidden state $\bar{\mathbf{h}}_t$ is a fixed-length continuous vector that updated in the same way as Eq. (2.8). At each time step t in the decoder, a time-dependent attentional context vector $\mathbf{c}_t$ is computed based on the current hidden state of the decoder $\bar{\mathbf{h}}_t$ and the whole context set C.

This starts by computing the content-based score of each context vector as:

$$e_{t,i} = \bar{\mathbf{h}}_t^\top W_a \mathbf{h}_i. \tag{2.9}$$

This relevance score measures how helpful the i-th context vector of the source sequence is in predicting next word based on decoder's current hidden state $\bar{\mathbf{h}}_t^\top$. These relevance scores are further normalized by the softmax function:

$$\alpha_{t,i} = \text{softmax}(e_{t,i}) = \frac{\exp(e_{t,i})}{\sum_{j=1}^{T_x} \exp(e_{t,j})}, \tag{2.10}$$

and we call $\alpha_{t,i}$ the attention weight.

The time-dependent context vector $\mathbf{c}_t$ is then the weighted sum of the context vectors with their attention weights from above:

$$\mathbf{c}_t = \sum_{i=1}^{T_x} \alpha_{t,i} \mathbf{h}_i. \tag{2.11}$$

With the context vector $\mathbf{c}_t$ and the hidden state $\mathbf{h}_t$, we then combine the information from both vectors to produce an attentional hidden state as follow:

$$\mathbf{z}_t = \tanh(\mathbf{W}_c[\mathbf{c}_t; \mathbf{h}_t]). \tag{2.12}$$

The probability distribution for the next target symbol is computed by

$$p(y_t = k | y_{<t}, X) = \mathrm{softmax}(\mathbf{W}_s \mathbf{z}_t + \mathbf{b}_t). \tag{2.13}$$

2.2.3 Parameter Optimization and Network Setup

If we define all the parameters in this model as θ, training this attention-based model can be done by minimize the negative conditional log-likelihood of the training data

$$\hat{\theta} = \arg\min_{\theta} \sum_{n=1}^{N} \sum_{t=1}^{T_y} -\log p(y_t = y_t^{(n)} | y_{<t}^{(n)}, \mathrm{X}^{(n)}; \theta),$$

where the log probability inside the inner summation is from Eq. (2.13). It is important to note that the ground-truth target words $y_{<t}^{(n)}$ is used as input during training. Because the entire model is end-to-end differentiable, so the gradient of the log-likelihood function with respect to all the parameters θ can be computed efficiently by back-propagation.

Part I

Example-driven Candidate Retrieval

Chapter 3

Answer Candidate Retrieval from Text Corpus

Current neural answer ranking models focus on re-ranking the candidates returned by the keyword-based search due to the efficiency constraint. Therefore, despite the recent success of neural models, the end-to-end answer retrieval performance is still bounded by the recall of the keyword-based search. The lexical gaps between questions and answers also make it hard to effectively retrieve some answer candidates with only keywords.

In this chapter, we explore the possibility of indexing and retrieving passages with neural embeddings. Specifically, we propose a neural encoder model that maps questions and answers into the same compact vector space where distances correspond to a measure of question-answer similarity. We also present simple, yet effective, training methods that can directly learn such unified encoder from the noisy community question-answer pairs without any dedicated human annotation. The complete module uses hybrid dense embeddings and keyword indexes to retrieve candidate passages for question answering. Our experiments indicate the retrieval effectiveness of the proposed method on the community answer retrieval task and show improved retrieval recall on the MS MARCO 8.8 million passages dataset without using its training data.

3.1 Introduction and Background

Large-scale open-domain question answering (QA) systems (Prager, 2006) typically consists of three stages: 1) corpus indexing, question analysis, and passages retrieval; 2) re-ranking and selecting passages that contain the answer; and 3) extracting (e.g. reading comprehension) and verifying the answer. In this chapter, we focus on the early-stage indexing and candidate retrieval task. Given a new question, the candidate retrieval module needs to search from a large corpus of passages and output the top-K candidate answer passages for the later stage passage re-rankers and readers.

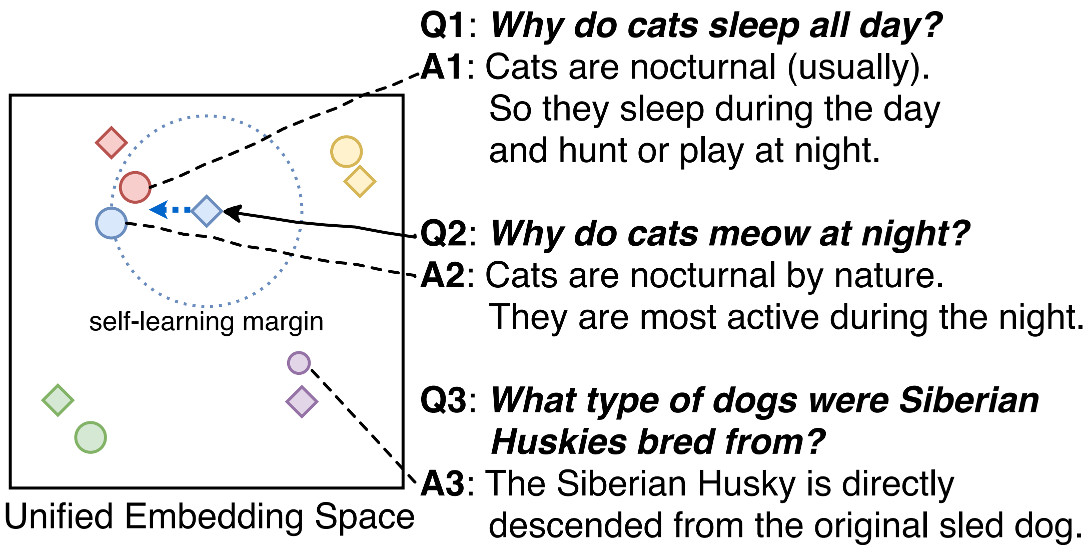

Q1: Why do cats sleep all day?
A1: Cats are nocturnal (usually).
So they sleep during the day
and hunt or play at night.
Q2: Why do cats meow at night?
A2: Cats are nocturnal by nature.
They are most active during the night.
Q3: What type of dogs were Siberian
Huskies bred from?
A3: The Siberian Husky is directly
descended from the original sled dog.
self-learning margin
Unified Embedding Space

Figure 3.1: An illustration of the unified dense indexing for QA. Diamond and round dots represent the mapped locations of questions and answers in the semantic space. During training, the blue arrow is the expected gradient direction for Q2, which move towards its corresponding answer and move away from those dots that are outside of the self-learning margin.

Recently, neural models have demonstrated great promise in many tasks including information retrieval (Mitra and Craswell, 2018) and QA (Chen et al., 2017a; Wang et al., 2018). However, despite the recent success of neural models on re-ranking benchmarks such as TrecQA (Wang et al., 2007b), WikiQA (Yang et al., 2015) and MS MARCO (Bajaj et al., 2016), the end-to-end QA performance is still limited by the keyword search recall. State-of-the-art re-ranking methods rely on evaluating the interactions (e.g. attention) between every question term and every candidate passage term, which is intractable to directly re-ranking the full corpus for every new query. At the same time, because of the "Lexical Chasm" (Berger et al., 2000) between questions and answers, lots of candidate answers don't have enough keyword overlap with the question, which makes it difficult to retrieve with just keyword combinations.

In this chapter, we propose a neural encoder model that maps questions and answers into the same compact embedding space where distances correspond to a measure of similarity between, as shown in Figure 3.1. Once this space has been produced, tasks such as retrieval and clustering can be easily implemented using standard nearest neighbor search techniques (Boytsov, 2018). The vector space nearest-neighbor search has already been widely used in real-world applications such as content-based image search or face identification.

However, there have been few positive results on training dense neural representations for large-scale QA retrieval. For instance, Guo et al. (2016) already shown that the state-of-the-art representation-based re-ranking models (e.g. DSSM (Huang et al., 2013), CDSSM (Shen et al., 2014), and ARC-I (Hu et al., 2014)) perform significantly worse than the traditional retrieval methods on ad-hoc retrieval. Note that the existing representation-based re-ranking models were all trained with softmax or triplet loss functions on click-through re-ranking datasets. Recently, the large-scale pre-trained language models such as ELMo (Peters et al., 2018) and BERT (Devlin et al., 2018a) have led to impressive gains in many NLP tasks. Nevertheless, these unsupervised representations are trained on plain text corpora and provide general-purpose sentence vectors, which still require supervised fine-tuning for each downstream NLP task. Qiao et al. (2019) shown that the pre-trained BERT (Devlin et al., 2018b) representations (without supervised fine-tuning) are ineffective on QA ranking tasks.

One core obstacle to achieving effective dense representations for QA dense indexing is the lack of large-scale clean supervision data. The existing training datasets for the re-ranking benchmarks are relatively small to learn the rich space that retrieval needs, and datasets created by annotating keyword retrieval results inevitably contains bias. Meanwhile, a vast amount of human answered questions are already available on Internet forums and essentially cover all the popular knowledge topics. However, the community QA corpus is noisy and doesn't contain gold-standard negative matches. The re-ranking analysis from (Surdeanu et al., 2011) shown that 18% of the re-ranker 'errors' are also correct answers that were automatically labeled negative, same as the A1 for Q2 in the Figure 3.1.

In this chapter, we address the challenge of training dense indexing encoder directly from noisy community QA pairs. We developed a neural encoder architecture and proposed a novel training objective to allow the robust training of distance metrics from noisy QA pairs. Our experiments, conducted on an open-domain corpus of community QA, show that the proposed approach significantly outperforms traditional retrieval models in all metrics. We also evaluated the pre-trained model on the MS MARCO (Bajaj et al., 2016) dataset for early-stage retrieval without using its training data, where the combined system of dense and keyword index shows large improvement on retrieval recall. As far as we are aware, this work presents the first effective neural dense indexer for the large-scale QA retrieval task.

3.2 Approach

This section first describes the problem formulation, and then introduces the training algorithm along with the proposed encoder network that maps QA pairs into the embedding space.

Let $\{(\mathbf{q}_i, \mathbf{a}_i)\}_{i=1}^{N}$ be the training data, where $(\mathbf{q}_i, \mathbf{a}_i)$ indicates i-th question and its corresponding answer. The indexing encoders are two learnable mapping functions $f_q(\mathbf{q}) \in \mathbb{R}^d$ and $f_a(\mathbf{a}) \in \mathbb{R}^d$ that each *independently* takes question or answer text as input and returns a d-dimensional embedding vector. Additionally, the output of encoders is L_2 normalized, i.e. $\|f_q(\mathbf{q})\|_2 = \|f_a(\mathbf{a})\|_2 = 1$. We then use the cosine similarity as the relevance scoring function for two encoded vectors: $r(\mathbf{q}, \mathbf{a}) = \langle f_q(\mathbf{q}), f_a(\mathbf{a}) \rangle \in [-1, 1]$.

We uses a recurrent neural network to implement the mapping functions f_q and f_a. The networks details are described in section 3.2.2. Given that the encoder networks are end-to-end learnable, the most crucial part of our approach lies in the training method that can effectively extract accurate supervision signals from the noisy community QA pairs. To this end, we design a new softmax based loss (in Section 3.2.1) which reflects the QA retrieval goal and accommodate the reality that training labels itself are not perfectly clean.

3.2.1 Self-Learning Additive Margin Softmax

Here, we aim to learn an embedding space such that the relevance score between the matched question and answer is high, whereas the relevance score between all the incorrectly matched QA pair is low. Namely, considering the ranking task, we want to ensure:

$$r(\mathbf{q_i}, \mathbf{a_i}) > r(\mathbf{q_i}, \mathbf{a_j}), \forall j \neq i. \tag{3.1}$$

The fixed-margin triplet loss (Chechik et al., 2010) ought to be directly applied here for metric learning. However, in our training data, each question is only associated with one positive training pair but has millions of noisy negative matches. The triplet loss was found to be ineffective in our experiments (in Section 3.4). We hypothesize that gradient directions learned from triplets are too localized and therefore not globally accurate for the complex semantic space, especially when some negative labels are wrong. For example, in Figure 3.2, the triplet objective will learn to push q_i toward to another negative match a_k since it only considered a_i and a_j in the current batch. Instead, we propose to use a very large set of negative samples $\mathcal{T}_i$ (e.g. $\|\mathcal{T}_i\| = 2k$) for each positive pairs to aggregate more accurate gradient direction for every gradient updates.

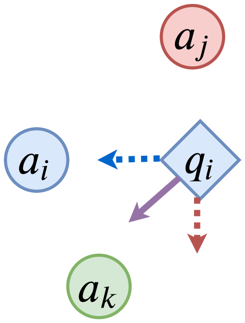


Figure 3.2: An illustration of triplet training issues on community QA pairs. Regarding the question q_i, the gradients from the positive (blue) and negative (red) pairs are combined (purple line) and lead to a negative match a_k.

Selecting informative training samples plays a key role in the metric learning. Meaningful improvements in performance were achieved by mining the hard training samples

(Bucher et al., 2016; Shrivastava et al., 2016). The hard samples are referring to the training instances that are being mislabeled by the current model. In our metric learning scenario, the hard negatives will be all a_j that satisfy the relation: $r(\mathbf{q_i}, \mathbf{a_i}) < r(\mathbf{q_i}, \mathbf{a_j})$. Despite the ground-truth hard samples help learn more discriminative models, the noisy hard samples also lead to large gradient noises. Unlike domains such as face identification, where negative labels are less noisy, simply treating all other answers in the community QA corpus as negative matches is problematic. As shown in Figure 3.1, the question Q2 could be pushed away from the answer A1 even they are also a good match. Therefore, we propose a simple margin-based method to recognize close neighbors and adjust gradients for those examples that might also be a good match, inspired by self-training (Rosenberg et al., 2005) which automatically augment the training labels with model predictions. Specifically, we filter the negative sample set and exclude samples that the model predicts to be good matches:

$$\hat{\mathcal{T}}_i = \{\mathbf{a_j} \in \mathcal{T}_i | r(\mathbf{q_i}, \mathbf{a_i}) + m_s < r(\mathbf{q_i}, \mathbf{a_j})\}, \tag{3.2}$$

where m_s is called the self-learning margin.

Sohn (2016) showed that softmax cross-entropy loss is a generalized triplet loss by enabling comparisons among multiple negative examples. However, from the analysis of (Liu et al., 2017), we can learn that even softmax loss is typically good at classification, it is not good at making the embeddings of the same class compact. To address this, we follow the additive margin softmax loss (Wang et al., 2018) that adds an angular margin via

$$\psi(\mathbf{q_i}, \mathbf{a_i}) = \cos\theta - m_a = r(\mathbf{q_i}, \mathbf{a_i}) - m_a. \tag{3.3}$$

Combining both margins above, we propose the Self-Learning Additive Margin Softmax (SLAM-Softmax) loss, for the task of selecting the correction answer given a question, we have:

$$\underset{q\to a}{\mathcal{L}} = -\frac{1}{N}\sum_{i=1}^{N} \log \frac{e^{s\cdot\psi(\mathbf{q_i},\mathbf{a_i})}}{e^{s\cdot\psi(\mathbf{q_i},\mathbf{a_i})} + \sum_{j\in\hat{\mathcal{T}}_i} e^{s\cdot r(\mathbf{q_i},\mathbf{a_j})}}, \tag{3.4}$$

where s is a fixed scale factor, ψ is the additive margin function that push positive question-answer pair closer, and $\hat{\mathcal{T}}_i$ is the large set of self-learned negative examples that aims to aggregate a globally accurate gradient direction in the embedding space.

Similarly, we define a reversed task of selecting the correctly matched question given an answer. And the final loss is the sum of both selection tasks:

$$\mathcal{L}_{q\leftrightarrow a} = \mathcal{L}_{q\to a} + \mathcal{L}_{a\to q}. \tag{3.5}$$

In order to reduce the computational burden of encoding $\mathcal{O}(N \cdot \|\mathcal{T}\|)$ vectors during training, we set the batch size equals to $\|\mathcal{T}\| + 1$ and treat answers from other positive QA pairs in the same batch as the negative sets. In this way, the negative samples are shared in the same batch and only need to be encoded once. In total, our training method just needs to encode $\mathcal{O}(N)$ vectors for each epoch.

3.2.2 Encoder Architecture

The goal of this encoders is to obtain a dense representation based on input words that capture the most discriminative passage meaning for ranking QA. Instead of training two separate encoders f_q and f_a, we train a single encoder that can accept two different types of inputs: $f_{qa}(\mathbf{w}, type)$, where $type \in \{question, answer\}$. This allows the model parameters to be effectively shared and can be efficiently trained.

The encoder first maps each word w_i into a d_e-dimensional vector using a mixed embedding matrix $\mathbf{E} \in \mathbb{R}^{|V| \times d_e}$, where $\mathbf{E} = \mathbf{E}_{pre} + \mathbf{E}_{tune}$. Here, $\mathbf{E}_{pre}$ is a pre-trained word embedding matrix that will be fixed during the model training, and $\mathbf{E}_{tune}$ is a trainable embedding matrix that is randomly initialized. To explicitly signal encoder the input *type*, the encoder accepts a binary feature to differentiate question and answer, which was concatenated to the input word embeddings at every time step.

The contextual representations $\mathbf{h}_i$ is then generated using a bidirectional LSTM (Hochreiter and Schmidhuber, 1997) from word embedding inputs (more background details regarding the recurrent neural networks can be found at Section 2.2). The encoder applies a max-pooling among the BiLSTM outputs: $\{\mathbf{h}_1, \ldots, \mathbf{h}_{\ell^w}\}$, where ℓ^w is the passage length. Finally, the encoder post-processes the max-pooling output with a two-layer fully-connected network with ReLU activations and dropout in between.

3.2.3 Integrating with Keyword Search

Ideally, the proposed dense indexer can encode and find candidate answers by 'meaning', which means it can discover those answers that have no overlapping word with the query. However, at the same time, the pre-trained neural model has limited ability to understand the unseen concepts or proper nouns that are out of vocabulary (OOV) during indexing and retrieving. For example, the neural model will fail at short query with unknown words (UNK), such as "could you define UNK?", while keyword search can fetch such rare concepts easily. To access the benefits of both type of indexes, we use a naïve approach to retrieval top K passages from both sources. Specifically, when all the words in query are In-Vocabulary (IV), the combined retrieval first adds top $K/2$ passages from the dense index, and fill the rest half with passages from keyword search that has not been added by the dense index search. If there is OOV words appear in the query, the combined retrieval only use the keyword searcher.

3.3 Experiments

Datasets The data used to train the index encoder consists of 10 million question-answer pairs crawled from answers.com website. A development and a testing dataset that both have 100K question-answer pairs are holdout from the crawled corpus. Namely, the testing corpus contains 100K candidate answer passages, and the query set contains

100K questions. The corpus is pre-processed using the BlingFire[1] tokenizer. We limit our model's vocabulary to 60K most-frequent terms in the CQA training set, excluding the boundary symbol and the unknown word tag.

We also evaluate our pre-trained model for indexing and retrieval on the MS MARCO corpus (Bajaj et al., 2016). The corpus contains 8.8 million unique passages. The development set contains 6,980 queries. On average, each query has one relevant passage. The data was originally used for re-ranking benchmark where each query was paired with 1K passages retrieved with BM25.

The Community QA and MS MARCO datasets are large-scale QA datasets. Both consist of open-domain real-user queries. MS-MARCO contains mixed factoid and non-factoid questions, while CQA mainly contains non-factoid questions.

Evaluation Metric We use Success@1 (S@1), Mean Reciprocal Rank (MRR@K), and Recall@K as evaluation metrics, which are calculated using the *trec_eval* evaluation scripts.

Implementation Details In this chapter, we limit our encoder to use single-layer BiLSTM with a hidden size of 500. The index embeddings (after concatenating from both directions and post-processing) are 1000-dimensional. The model uses the 300-dimensional FastText embeddings pre-trained on Wikipedia (Bojanowski et al., 2017). The network weights were randomly initialized using a uniform distribution $(-0.08, 0.08)$, and are trained with the ADAM optimizer (Kingma and Ba, 2014). The hyper-parameters are all chosen by sweeping on the CQA's validation dataset. In the following experiments, the fixed scale factor s, additive margin m_a, and self-learning margin m_s are set as 40, 0.1, and 0.05 accordingly.

Answer passage embeddings are stored with faiss[2] for efficient nearest neighbor search. In our experiment, even without search approximations, the indexing and retrieval speed of dense search using Faiss (GPU) is comparable to the Lucene-based Anserini toolkit (CPU), and becomes faster than Anserini when processed in batches.

3.4 Results and Analysis

Table 3.1 summarizes the results from our model and baselines on the community QA retrieval task. We compared our indexer with the classical BM25 model and also BM25 with pseudo relevance feedback based query expansion RM3[3]. Following (Otsuka et al., 2018), we also implemented and compared a query expansion model using neural machine translation, referring as NMT. The results indicate that the "lexical-chasm" issue, which the question and answer often use different terms, fundamentally limits the

---

[1] https://github.com/microsoft/BlingFire

[2] https://github.com/facebookresearch/faiss

[3] Anserini toolkit (Yang et al., 2018a) was used with default parameters to produce results of traditional retrieval models.

	S@1	MRR	Recall		
		@1K	@10	@100	@1K
BM25	0.215	0.258	0.338	0.452	0.560
RM3	0.178	0.224	0.316	0.435	0.557
NMT	0.186	0.230	0.313	0.444	0.570
Triplet	0.008	0.018	0.033	0.128	0.401
Softmax	0.020	0.040	0.075	0.236	0.553
$\|\mathcal{T}\|$ with SLAM-Softmax					
20	0.131	0.191	0.307	0.555	0.792
200	0.288	0.367	0.523	0.739	0.873
2000	**0.371**	**0.442**	**0.590**	**0.758**	**0.878**

Table 3.1: Overview of results on the community QA retrieval task. The last three rows are results from our method with different size of negative samples $\|\mathcal{T}\|$ used during training.

	IV	OOV	All
(size)	(5512)	(1468)	(6980)
Dense	0.8685	0.3621	0.7619
BM25	0.8346	**0.9201**	0.8578
Combined	**0.9234**		**0.9227**

Table 3.2: Overview of Recall@1000 results on retrieval from MS MARCO full corpus. Dense refer to our best performing model in Table 3.1. The combined search are generated by the method described in Section 3.2.3

keyword-search performance. Therefore, query expansions (RM3, NMT) has shown little improvement as they also easily introduce noise term. Plain softmax (20 negative samples and same scaling factor) and triplet losses are also included for comparisons. The results show that our model significantly outperforms plain keyword, query expansion, plain softmax, and triplet based methods on all metrics. We also found that the size of the set $\|\mathcal{T}\|$ correlates positively with the ranking performance.

Table 3.2 summarizes our retrieval experiment results on the full MS MARCO corpus. We are evaluating recall@1000 since this early-stage retrieval output will serve as input to later stage re-rankers (e.g., based on BERT). Note that we follow a practical usage and let BM25 has a full unrestricted vocabulary. Therefore BM25 doesn't have any OOV query understanding issues in our experiments. When applying our pre-trained model to index and retrieval in this new domain, we found the dense model is more effective than BM25 when query concepts are in-vocabulary, but less so when there are OOVs. Furthermore, two indexes discover complementary kinds of relevant answer passages, and the recall improved significantly when combining both searchers.

3.5 Related Work

In the context of embedding question and answer candidates with dense semantic representation, several approaches have been proposed. Zhou et al. (2016) learned the dense representations of questions and answers by first pre-training two separate autoencoders and then transforming the encoded representations to optimize the cosine similarities. The similarity score produced by dense representations has been used as an additional learning-to-rank feature along with BM25 for the re-ranking task. The triplet loss was used for correlatedness training, which is also used as a baseline loss function in our experiments

Padmanabhan et al. (2017) proposed a latent space model to embed each question-answer pair (q_i, a_i) into one d-dimensional vector u_i, which can later be used to retrieve similar question-answer pairs given a new question. The model attempts to reconstruct ti-idf of (q_i, a_i) while preserving the local tf-idf cosine similarities. The dimensionality reduction is achieved by extracting d eigenvectors of the reconstructed ti-idf representations. However, the model was designed for duplicated question-answer retrieval, which cannot be directly applied and compared on answer retrieval tasks such as MS MARCO. Specifically, the CQADupStack dataset (Hoogeveen et al., 2015) was used for training and evaluation, which was created from StackExchange forums for the task of duplicate question retrieval. On the other hand, the CQA and MS MARCO datasets we evaluated on are designed for the task of retrieving candidate answers directly.

3.6 Conclusion

Improving the QA retrieval's recall with the dense index is desired in many end-to-end QA and chatbot applications. However, training this open-domain semantic space requires massive data. And, unfortunately, there are no large-scale ground-truth negative labels for any QA dataset. In this study, we showed that constructing the dense semantic space for large-scale QA retrieval is achievable. The proposed training algorithm can directly learn such space from the noisy community question-answer pairs without any dedicated human annotations. Experimental results on real-world datasets show that our model can capture useful information from the noisy training and achieve significant improvements in QA retrieval tasks as compared to all baselines. At the dawn of embedding-based QA retrieval research, we intended to keep the encoder simple and efficient for practical use-cases and focus on the training objective. The more complex encoders are therefore left as future work.

Chapter 4

Answer Candidate Retrieval from Knowledge Graph

This chapter introduces a bootstrapping approach for training open-domain question answering over knowledge graph. Specifically, we propose a relation detection model utilizing the potential question-to-query mappings extracted from community question-answer pairs.

4.1 Introduction

Open-domain Knowledge Graphs (KG) such as Freebase (Bollacker et al., 2008), NELL (Carlson et al., 2010), Wikidata (Vrandečić and Krötzsch, 2014) and DBpedia (Auer et al., 2007) contain a vast wealth of knowledge. There has been growing research interest to build automatic QA systems that can answer factoid questions via the lookup of a simple KG fact. SimpleQuestions (Bordes et al., 2015) is a commonly used benchmark for factoid QA over KG (KG-QA), whose questions can be answered using a single knowledge entry. The research community has developed various models targeting its key sub-tasks such as entity linking and relation detection. However, existing approaches to training KG-QA systems have relied on dedicated human annotations that are not only costly to create but are also restricted to a particular set of KG schemas. This is undesirable as open-domain KGs are varied and their schemas keep changing, while static training dataset cannot adapt to the changes or new domain.

In this chapter, we introduce an approach to train KG-QA systems without relying on any human annotated mappings from question to the structured query. Instead, we propose to automatically extract question-to-query mappings from community QA sites and use them for training. However, the automatically extracted mappings are extremely noisy. For instance, there are only around 2% of the raw mappings are ground-truth labels in our SimpleQuestions experiments. Therefore, we introduced a training strategy that denoises the extracted training samples through popularity weighting, keyword-based instance filtering, and instance re-weighting with self-training. We demonstrate significant cumulative improvements correlated with each training weighting strategy and validate our approach on SimpleQuestions benchmark dataset. Our approach achieves an accuracy of 78.11% for the relation detection task on the SimpleQuestions dataset without using any annotated training data.

4.2 Related Work

In open-domain KG-QA systems, the input is a natural language question. The goal is to map the text question to a structured query. Researchers have proposed various approaches to this task. The early thread of work derives from semantic parsing (Cai and Yates, 2013; Berant et al., 2013; Berant and Liang, 2014; Reddy et al., 2014), where the question is mapped to a logical form representation (e.g. CCG or λ-DCS) and then converted into a structured query. Another research thread attempts to select the answer by measuring the semantic similarity between question and KG facts in latent embedding spaces (Bordes et al., 2014b,a). Yih et al. (2015) proposed a multi-stage strategy to match question utterance to graph queries, where multiple dedicated models were trained to produce partial similarity measurements for each segment of the query. Recently, the SimpleQuestions (Bordes et al., 2015) dataset has become the de facto benchmark for evaluating KG-QA systems, where questions can be answered by a single fact. In this setup, building KG-QA systems need to solve the following three principal challenges:

- Recognizing named entities in the question and linking them with the most likely entries in the knowledge graph. The word or phrase in the unstructured question often ambiguously implies various database entries. For example, the word "Obama" can refer to both "Barack Obama (44th President of the United States) " and "Obama, Fukui (a city located in Fukui Prefecture, Japan)".
- Mapping question's semantic intention to graph relations (the predicate in the triple). For instance, the question "who is the wife of …" could link to the "spouse"[1] relation.
- Identifying constraints and dependencies between the linked entities and predicates, and deciding the final query.

Bordes et al. (2015) originally introduced a memory-networks solution for the SimpleQuestions dataset. In addition to the SimpleQuestions training set, Bordes et al. (2015) also utilized the 3K training questions from (Berant et al., 2013) and 15M paraphrases from WikiAnswers (Fader et al., 2013) for the training purpose. Later, various neural architectures have been proposed for this task from different perspectives to further advance the performance. Dai et al. (2016) formulated the task with a unified conditional probabilistic framework and trained components using BiGRUs. Golub and He (2016) treated the task as sequence-to-sequence problem and proposed a character-

---

[1] `http://dbpedia.org/ontology/spouse`

based encoder-decoder model. Yin et al. (2016) experimented separated entity linking and fact selection modules with CNN, while (Yu et al., 2017) employed a hierarchical residual BiLSTM encoder. Lukovnikov et al. (2017) introduced hierarchical encoders using both character and word representations and implemented end-to-end training for all modules. Hao et al. (2018) suggested using BiLSTM-CRF for predicting entity mention span and jointly scoring fact selection with entity linking and relation detection. In general, these approaches focus on better modeling of question representation and query prediction exploiting advanced neural techniques.

However, all existing neural approaches rely on the human-annotated training dataset, which is commonly limited to specific KG or domains. For instance, the training split of SimpleQuestions dataset contains 75,910 annotated mappings of the questions and its Freebase facts, while the Freebase project has been obsolete for years. To this end, our work attempts to train a neural KG-QA system without using any human-annotated data. Our approach automatically extracts noisy mappings from the community QA forum and de-noise them for modules training. This setup allows us to quickly establish open KG-QA baselines for any new KG.

Our bootstrapping approach for training KG-QA is similar to the distantly supervised relation extraction (RE) (Mintz et al., 2009). Their task goal is to extract relational facts from text corpora such as Wikipedia or the New York Times (NYT). The distant supervision for RE also generates the training relation labels by automatically aligning sentences (each contains a pair of entities) with knowledge graph facts. Riedel et al. (2010) shown that the distant supervision signals contain approximately 13% noisy labels when mapping Freebase to Wikipedia and 31% noisy labels when aligning Freebase to the NYT corpus. In order to alleviate the negative impact of those noisy training instances, multi-instance multi-label learning (MIML) framework was widely adopted (Mintz et al., 2009; Hoffmann et al., 2011; Surdeanu et al., 2012), which extracts relation from a group of sentences mentioning the same entity pair. The earlier MIML approaches (Mintz et al., 2009; Hoffmann et al., 2011; Surdeanu et al., 2012) were feature-based. Recently, neural network (Zeng et al., 2015; Lin et al., 2016) was also incorporated with the MIML framework for the distantly supervised RE. For example, Zeng et al. (2015) trained a convolutional neural network to select the most likely sentence in the sentence group for training and prediction, while Lin et al. (2016) aggregated sentence embeddings in each group using a neural attention model. However, all the MIML-based approaches make the *at-least-one assumption*: if two entities are linked with a relation in a KG, at least one sentence that contains the entity pair expresses that relation. By contrast, our pipeline generates noisy relation candidates between *all* entities pairs mentioned in the questions and their answer passages, which often violate the *at-least-one* assumption. Let us consider the example question “What awards did *Marie Curie* win?” and the example answer passage “Marie Curie was the first *female* to win a *Nobel Prize*.” The ideal relation for the example question is *awards*, which can be discovered between the entity pair: “Marie Curie” and “Nobel Prize”. This example QA pair also generates a noisy training mapping to the *gender* relation, which is found from the entity pair: “Marie Curie” and “female”. In order to denoise the *gender* relation in the

MIML framework, the at-least-one assumption needs another question in the corpus that means to ask Marie Curie's gender, which often does not exist in our dataset. Furthermore, around 98% of the raw mappings are noisy in our experiments. Therefore, we do not apply the MIML-based methods that widely used in RE for KG-QA, but develop principled denoising methods that directly re-weighing the training instance for neural model training.

4.3 Approach

A simplified knowledge graph can be expressed by a collection of triples (s, p, o), where each s refers to the subject; p indicates the predicate, and o is the object. Each subject must be a concept. The concept may indicate a real-world entity such as "Bob Dylan", or an abstract notion such as "Artist". And the object in a triple can be either a concept or a literal value. Each predicate is a special concept representing the relationship between subject and object. The inverse relation of p is denoted by p^{-1}, and (s, p, o) is equivalent to (o, p^{-1}, s). Then a knowledge graph can be considered as a directed graph with labeled edges, where subjects and objects are nodes and predicates are edges. Figure 4.1 shows a example sub-graph describing "Bob Dylan" in a knowledge graph.

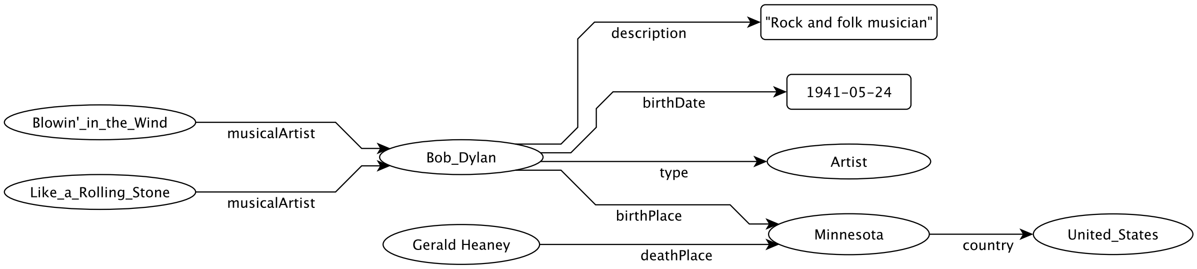


Figure 4.1: An example knowledge subgraph describing "Bob Dylan"

A structured query for knowledge graphs consists of a set of triple patterns known as a Basic Graph Pattern (BGP). Each triple pattern is similar to triples (s, p, o) except that each subject, predicate, and object can be a variable. For example, a very basic BGP can be finding o for given corresponding s and p, such as a given subject "Bill Clinton" and the predicate "child" to find variable "?children" that will be "Chelsea Clinton".

4.3.1 System Overview

Our proposed KG-QA system answers open-domain the natural language question by translating it to a BGP query, which comprises the steps of:

1. recognizing and linking the most plausible knowledge entity from the question;
2. constructing question template text by excluding the recognized entity;

3. selecting the most likely predicate based on question template text and the linked entity.

We use a simple dictionary-based approach to detect and link named entities (Section 4.3.3). The predicate prediction module is implemented with a recurrent neural network. The networks details are described in section 4.3.5.

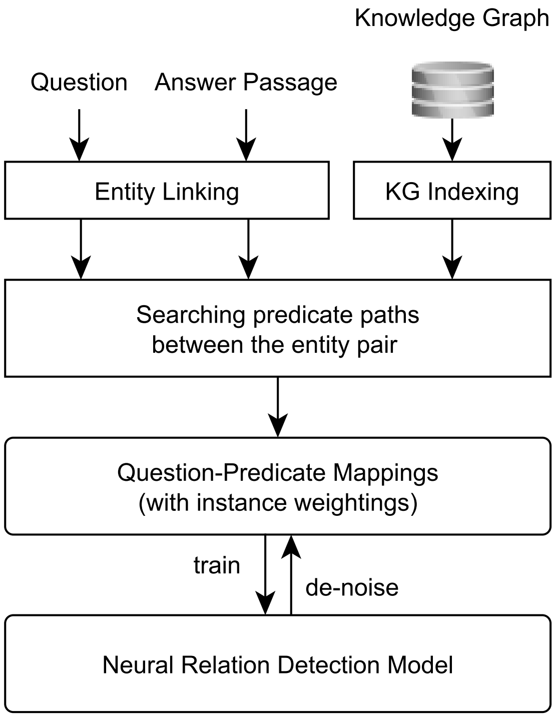

Knowledge Graph
Question
Answer Passage
Entity Linking
KG Indexing
Searching predicate paths between the entity pair
Question-Predicate Mappings (with instance weightings)
train
de-noise
Neural Relation Detection Model

Figure 4.2: The workflow of the proposed training process for our KG-QA system

Given that the neural networks are end-to-end trainable, the most essential aspect of our approach is the training method that can effectively extract mapping patterns from the noisy community QA pairs and de-noise the supervision signals for accurate training. We propose a novel approach (as depicted in Figure 4.2) that automatically discovers links between textual patterns and predicate without dedicated human annotations. To this end, we implemented an efficient KG index (in Section 4.3.2) for seeking possible predicate connections between pairs of entities found in each of the questions and answer passages. Because the original mappings discovered by our KG search were extremely noisy, we proposed denoising procedures (in Section 4.3.5) based on the popularity, keyword overlapping, and model's self-training confidence.

4.3.2 Knowledge Graph Indexing

N-Triples (Beckett, 2014) is a simple and widely used format for storing knowledge graph. It employs Uniform Resource Identifier (URI) to represent concepts. Each nonconcept literal is stored as a quoted string and optionally can be appended with a URI that indicates its data type. The figure 4.3 lists two example N-Triples from DBpedia knowledge graph.

```
<http://dbpedia.org/resource/Canada> ↩
    <http://dbpedia.org/ontology/capital> ↩
        <http://dbpedia.org/ontology/Ottawa> .

<http://dbpedia.org/resource/Canada> ↩
    <http://dbpedia.org/ontology/foundingDate> ↩
        "1931-12-11"^^<http://www.w3.org/2001/XMLSchema#date> .
```

Figure 4.3: Example N-Triples in DBpedia knowledge graph

In order to support efficient matching of BGP queries for millions of community QA pairs, a fast triple index structure is required. Since our BGP queries always have two known elements and one unknown target variable, only three indices need to be made for the permutations of the subject (S), predicate (P), and object (O), which are SP-O, OP-S, and SO-P. For example, the index SO-P implements a mapping function: $I_{sop}(s, o) = \{p|(s, p, o) \in K\}$, where K is all triples in the knowledge graph.

For space efficiency consideration, a standard technique is replacing all URI references and literals by numerical ids using a *name dictionary* (Chong et al., 2005). A path in the knowledge graph can then be stored as a sequence of integers. The name dictionary could introduce additional cost when mapping URIs and literals into integer IDs. However, since the system takes natural language questions as input instead of structured queries, the Named-Entity Linking module (see Section 4.3.3) can directly map from mention texts to numerical IDs, which eliminate the extra cost of using the name dictionary.

The underlying triple indexing data structure is an in-memory hash table, where two known IDs are bundled together as the key. We observed that the vast majority of keys only have one target ID. Hence, for both space and time efficiency, we maintained two hash tables in parallel for each index, where one hash table only stores singular target value and another one keeps those has multiple target values. Since the keys and values have compressed into integers, the hash table index can fit into main memory, which avoids using disk-friendly but slower looking up data structures such as B+ trees.

4.3.3 Named Entity Linking

The function of Named Entity Linking (NEL) module is to spot text segments and link them to probable knowledge graph entities. We utilize anchor text or titles resources for mapping texts in varied expressions to KG entities, which not require additional human annotations.

In many knowledge graphs such as DBpedia, each knowledge entity (topic) has its publicly available URL. Furthermore, web pages such as news, blogs, and Wikipedia itself often add hyperlinks that point to related Wikipedia topics within the passages. For example, an HTML web page might contain a hyperlink

```
<a href="https://en.wikipedia.org/wiki/Michael_Jackson">The King of Pop</a>
```

that links plain text "The King of Pop" to the Wikipedia page of "Michael Jackson". Therefore, the mention texts in hyperlinks can be used as potential textual expressions for an entity. Also, the *label* field is mandatory in most KG and can be used as each entity's possible mention text.

We built a simple dictionary of mention text to entities. Given a mention text, possible entities are ranked according to their saliences, which measured by the number of facts that the entity serve as the subject in the KG. Aho-Corasick algorithm is then used to spot all possible entity mention spans within a sentence. The NEL module also detects date, time, string literals, and other numbers such as monetary values. For example, numbers such as "3.1415 million" will be extracted and approximately transformed to form as 3.14×10^6. The normalized number is stored and searched in the SO-P index discussed in Section 4.3.2, which enables approximate number matching between free text and knowledge graph.

4.3.4 Discovering Template Mappings from Question-Answer Pairs

The next challenge is to discover question patterns for each KG. To this end, we propose an approach that automatically collects the potential associations between questions and predicates using a question-answer collection and KG. The procedure for extracting template is outlined in the following steps:

1. Given a pair of the question and its answer passage, the NEL module (in Section 4.3.3) is applied to spot entity mentions S_q from the question and S_a from the answer. Each entity mention $t = (w_i, \ldots, w_j) \in S$ associates with a set of entities $E(t)$.

2. For every pair of entity mention texts $\{(t_q, t_a) | t_q \in S_q, t_a \in S_a\}$:

    (a) A template surface is constructed based on the question and t_q: $T = (w_1, \ldots, w_{i-1}, M, w_{j+1}, \ldots, w_n)$, where M is a special slot symbol.

    (b) For every pair of entities (e_q, e_a) where $e_q \in E(t_q)$ and $e_a \in E(t_a)$. The KG search module program exams whether there exists a predicate path between e_q and e_a in the knowledge graph.

        i. If one or more predicate paths between e_q and e_a are discovered, those predicate paths r and e_q will be added to the predicate counter $C_{r,T}$ for template surface T.

   ii. Instead, if no predicate path is detected from all pairs of entity between $E(t_q)$ and $E(t_a)$, a special "no path found" symbol will be recorded for T.

3. Repeat above steps over all training pairs of question and answer.

The predicate paths observed during the above process have different credibilities. When accumulating $C_{r,T}$, if c paths are detected between a pair of entities (e_q, e_a), the weight of each path in template path distribution will be equally divided by c. The purpose of this operation is to deal with the semantic duplication in knowledge graphs. Given training question "Who is Bill Gates' father?" and answer "William Gates", there exist two semantically equivalent predicate paths $parent$ and $child^{-1}$ from "Bill Gates" to "William Gates". This normalization operation makes sure duplicated predicates will not gain extra credits, as contrasted with ordinary predicates which only appear once in one direction.

The brute-force nature of the above procedure and the noisy community QA data introduced lots of predicates that do not reflect the template's semantic intents. For example, when processing with the question "who is the mayor of Chicago" and its answer text "The current mayor of Chicago is Rahm Emanuel.", the predicate paths between "Chicago" and "Rahm Emanuel" are both $leaderName$ and $birthPlace^{-1}$. Even though many city mayors were indeed born at the city too, the $birthPlace^{-1}$ is not one of the desired predicate paths for the training purpose. To overcome this problem, a relatively large collection of training questions and answers is necessary to provide reliable statistical information about predicate paths for each template. Specifically, the training data also contains questions about "who is mayor of Toronto" and "who is mayor of New York" that only related to $leaderName$ not $birthPlace^{-1}$, which makes predicate $leaderName$ statistically dominates for templates "who is mayor of [CITY]?".

By going through a large collection of question-answer pairs, we can directly obtain predicate (relation) path distributions over each template surface. The probability of predicate path r given template T is: $p(r|T) = C_{r,T} / \sum_k C_{k,T}$. For example, given template $T_1 =$ "`who is the president of [ORGANIZATION]?`", the naive model can return a list of possible predicates with different probabilities such as $p(leaderName|T_1) = 0.5$ and $p(keyPerson|T_1) = 0.1$.

4.3.5 Training Relation Detection from Templates

Given the templates extracted in Section 4.3.4, simple questions can be answered by directly matching known templates. Similar to the extraction process, the input questions are first scanned by NEL to extract an entity mention $t_q = (w_i, \ldots, w_j)$ and then construct a question template $T_q = (w_1, \ldots, w_{i-1}, M, w_{j+1}, \ldots, w_n)$. If the template query matches any known templates, the predicates associated to the known templates can be used for generating BGP queries. The SP-O or OP-S indexes (in Section 4.3.2) can handle the BGP queries and retrieve candidate answer entities. However, despite the question templates collection is large, lots of unseen question can not exactly match to any known templates. In this section, we propose to generalize the collected templates and use them to train a neural relation detection model.

Model

Given a question template T_q (e.g. "Where was M born?") and an entity e_q (e.g. "Michael Jackson"), the relation detection model estimates a conditional probability $P(r|T_q, e_q)$ over KG relations. In the SimpleQuestions task, the candidate relations are a set of predicates $R = \{r_1, \ldots, r_m\}$ in the Freebase facts that co-occur with entity e_q as the subject, and the goal is to select the most likely relation $r^* = \arg\max_{r \in R} P(r|T_q, e_q)$.

Every word in T_q is initially mapped to a d-dimensional vector using the pre-trained FastText (Bojanowski et al., 2017) word embeddings $\mathbf{E}$ which kept frozen during training. Next, we use bi-directional recurrent neural networks to generate contextual representations $\mathbf{h}_i$ over the question word embeddings inputs. Specifically, it processes the word embedding input from both directions with two separate hidden sub-layers:

$$\begin{aligned} \overrightarrow{\mathbf{h}_i} &= \overrightarrow{\Psi}\left(\overrightarrow{\mathbf{h}}_{i-1}, \mathbf{E}\left[w_i\right]\right) \\ \overleftarrow{\mathbf{h}_i} &= \overleftarrow{\Psi}\left(\overleftarrow{\mathbf{h}}_{i+1}, \mathbf{E}\left[w_i\right]\right) \\ \mathbf{h}_i &= \left[\overrightarrow{\mathbf{h}_i}, \overleftarrow{\mathbf{h}_i}\right], \end{aligned} \tag{4.1}$$

where Ψ is a recurrent activation unit that is LSTM (Hochreiter and Schmidhuber, 1997) in our experiments. In order to aggregate the sequence of contextual vectors, the max-pooling is applied among the BiLSTM outputs: $\{\mathbf{h}_1, \ldots, \mathbf{h}_{\ell^w}\}$, where ℓ^w is the input length. The max-pooling output is provided as input to a post-process block consisting of a two-layers fully-connected network with a ReLU activation and dropout in between. Finally, the relation detection model utilizes a softmax layer to estimate predicate likelihoods.

Denoising Training Instances

The proposed neural relation detection model is end-to-end trainable with question templates as the input and corresponding predicates as the target. However, since our question templates discovery process is done in an unsupervised and brute-force manner, a great many of the extracted templates are not valid (e.g. wrong spans of named entity) or mapped to wrong predicates. We randomly sampled and validated one hundred extracted template-to-predicate mappings, and found that only 2% of them are ground-truth mappings. Because the overwhelming majority of the raw training material is corrupted, training the model directly using the raw mappings will inevitably produce undesirable results, especially for the neural networks which are known to memorize instances.

To address this problem, we proposed multiple principled filtering and weighting methods that can effectively denoise the training material and produce more reliable supervision signals for the training. The credibility weights are applied during the gradient descent process of the neural model training, which tries to ensure the unlikely mappings has little impact on the learned model.

Frequency-based Weighting As mentioned in Section 4.3.4, we maintained a predicate counter $C_{r,T}$ for each question template T and each potential predicate relation r. Because it is unlikely that invalid questions templates or incorrect predicate mappings constantly repeat themselves, we can view the predicate counter value as a credibility measurement for the template-predicate mappings. Although this principle is generally true, there are exceptions. For example, some general predicate relations (e.g. *withinCountry*) are found linked to templates with a wide variety of intents, which are often undesired for the QA purpose. To alleviate their effects, predicates that appear too often in templates will have lower weights in a manner similar to the inverse document frequency:

$$\mathrm{idf}_r = \log \frac{N}{1 + |\{T : C_{r,T} > 0\}|}, \tag{4.2}$$

where N is the total number of templates and $|\{T : C_{r,T} > 0\}|$ is the number of templates that the predicate relation r has associated to. The above two aspects of credibility measurement are then combined to create our frequency-based weighting function for the predicate relation r regarding the template T:

$$\mathbf{F}\text{-weight}_{r,T} = C_{r,T} \times \mathrm{idf}_r \tag{4.3}$$

Keyword-based Weighting Another aspect of the question-to-predicate mappings that can affect the training credibility is whether the question template and predicate share keywords. In the fully supervised setting, the textual information in the predicate URI has been found beneficial to improve the relation detection performance. For example, instead of treating relation detection as a multiclass classification task, Golub and He (2016) proposed a character-level CNN to encode predicate URI and then calculate the semantic relevance scores based on the encoded vectors. We observed that the tokenized predicate URI normally human-readable and indicate the predicate's category and topic. This suggests that using words in predicates may be a helpful credibility heuristic. Here, we define a keyword-based weighting $\mathbf{K}\text{-weight}_{r,T}$ that simply equals to the number of overlapping words between a question template and a tokenized predicate URI. Note that a template-predicate will have zero weight (i.e. ignored) if they don't have any shared words.

Self-trained Weighting Inspired by the self-training (Clark et al., 2003; Rosenberg et al., 2005) which automatically argument the training labels with the model predictions, we propose a multi-stage learning schedule that re-weight training instance by a model trained from the two principled weightings above. The existing self-training was mainly used in semi-supervised learning settings, where both annotated and unannotated data are provided for training. The model is trained from the annotated data at first, and then used to label the unannotated data. The newly labeled data is appended to the original annotated data to train an updated model. Unsurprisingly, the self-training can be often not effective in practice. One plausible explanation is that the prediction error from the original model could easily propagate (and sometimes amplify) to the new model.

Our training schedule consists of three phases. First, we weight the training instances using the product of F-weight and K-weight and use the weighted examples to train the initial model. Next, the initial trained model evaluates the training instances and generate a likelihood score $l(r, T)$ for each template-predicate pair. Finally, we train a new model using a weight function that combines both the self-trained likelihoods and the frequency-based weights:

$$\textbf{S}\text{-weight}_{r,T} = \text{F-weight}_{r,T} \times (l(r, T) + \alpha), \tag{4.4}$$

where α is a smoothing parameter that prevents the self-trained weights from completely filtering out instance when the initial model is over confident (the α it set as 0.1 without further tuning). The special "no path found" predicate (described in Section 4.3.4) also serves as a regularization for the initial model.

Note that, the K-weight only focuses on selecting a small subset of easy training instances identified by keywords. In other words, the proposed training schedule essentially first learn from simple and reliable instances and then expand to the full dataset. This resembles the idea of curriculum learning (Bengio et al., 2009), which model training starts with easy examples and then gradually introduce more difficult examples.

4.4 Experiments

4.4.1 Experimental Setup

We evaluate the proposed relation detection model on the SimpleQuestions dataset (Bordes et al., 2015). The dataset contains a total of 108,442 questions written by annotators. Each question is paired with a fact consisting of *(subject, predicate, object)* from the Freebase (Bollacker et al., 2008). In particular, the *subject* corresponds to the topic entity in the question, the *predicate* indicates the semantic intent of the question, and the *object* represents the target answer. The dataset was split up into 75,910 (70%) train, 10,845 (10%) validation, and 21,687 (20%) test questions. Unlike prior supervised approaches, our proposed method does not rely on the annotated training set and only use the test dataset for evaluating the performance.

In order to compare with early works on relation detection, we use the same Freebase subset consisting of 2M entities (called FB2M). The KG is indexed as discussed in Section 4.3.2 and then used for discovering template-predicate mappings as described in Section 4.3.4. We follow the same evaluation setup of the relation detection task as in Yin et al. (2016); Yu et al. (2017). Specifically, each test input comprises a question template and

Number of Community QA pairs	10,704,262
Number of Extracted Template-Predicate Mappings	4,407,295
Number of Template-Predicate Mappings has Keyword Overlap	229,393
Number of Predicate Classes in SimpleQuestions (test)	1,034
Number of Predicate Classes found in Extracted Mappings	905

Table 4.1: Summary of the Discovered Template-Predicate Mappings

a pool of candidate predicates. The testing question templates were produced using their gold topic entities. Each predicate pool was formed with all possible predicates connected to the topic entity in the KG. The task is to select the correct predicate from the predicate pool.

We train the model using the Adam optimizer (Kingma and Ba, 2014) with an initial learning rate of 0.001. The batch size is set as 2000. The word vectors are the pre-trained fastText (Bojanowski et al., 2017) embeddings of size 300 trained on English Wikipedia and kept frozen during our model training. We limit our model input vocabulary to 30,000 most-frequent terms from the extracted templates. The hidden states sizes of LSTM and the fully-connected layer are set to 300, with probability 0.2 of dropout.

4.4.2 Results

The data used for extracting question templates consists of 10 million question-answer pairs crawled from answers.com website. As shown in Table 4.1, our automatic template-predicate discovery process found 4.4 million unique template-predicate mappings. Because the community QA dataset covers diverse topics, while the FB2M KG is relatively small, many QA pairs unable to match any KG facts and formulate templates. We can also find that the keyword-based weighting *K-weight* filters out most of the template-predicate mappings and only select 5.2% (229,393/4,407,295) of all extracted mappings.

Even though the community QA corpus is fairly large, many predicate relations are still very rear. We counted the numbers of unique predicate classes in the test dataset and in the extracted mappings in Table 4.1. We found 12.48% of the testing predicates has not been discovered for template-predicate mappings using the given QA corpus and KG. This gap inevitably limits the performance of our trained relation detection model.

In Table 4.2, we present the relation detection results of the state-of-the-art supervised approaches as well as our bootstrapping models. There are significant cumulative improvements (5.3%, 6.8%, 5.8%) with our proposed denoising weighting strategies. We also included a keyword matching baseline that simply selects the predicate if a tokenized predicate contains more keywords overlapping with the question template than other predicates. The keyword matching baseline is very effective on rare target predicate cases, which are often absent in our automatically extracted training data. Finally, we combined the keyword matching with the trained neural model and found they are

Model	**Accuracy**
Supervised with the annotated training dataset	
BiLSTM with relation names	88.9
BiCNN (Yih et al., 2015)	90.0
AMPCNN (Yin et al., 2016)	91.3
Hier-Res-BiLSTM (Yu et al., 2017)	**93.3**
Without using the annotated training dataset	
Keyword Match	44.9
Uniform weight	52.8
F-weight	58.1
F-weight + K-weight	64.9
F-weight + K-weight + S-weight	70.7
F-weight + K-weight + S-weight + Keyword Match	**78.1**

Table 4.2: Summary of our and past results of the relation detection task on the SimpleQuestions benchmark. Note that the previous models were trained with the annotated training dataset while our models were bootstrapped from the unannotated community QA corpus.

complementary (7.4% improvement). Specifically, the combined system attempts to use the keyword matching baseline first, and only call the neural model when the keyword matching has no matches or ties.

As expected, the supervised approaches using the annotated training dataset have higher accuracies than the bootstrapping approaches, indicating there is still much room for improvement. Also note that Petrochuk and Zettlemoyer (2018) found one-third of the dataset questions are ambiguous. Supervised models learn to exploit the label sampling biases to resolve this ambiguity, while our bootstrapping approaches follow the label distribution on the community QA data.

4.5 Conclusion

In this chapter, we introduced a new bootstrapping approach for training knowledge graph question answering. Given question-answer pairs, our approach attempts to automatically detect potential knowledge graph connection between the question and the answer passage. The extracted mappings between question template and predicate relations then process with three kinds of de-noising weighting, namely frequency-based, keyword-base, and self-trained weightings. As a result, we demonstrated the feasibility of training relation detection for KG-QA without any human-annotated data.

Part II

Example-driven Relevance Modeling

Chapter 5

Answer Context Ranking

In this chapter, we discuss our development on answer context ranking by the question paraphrase identification (Wang and Nyberg, 2017).

5.1 Introduction

Two questions are paraphrases of each other if all their correct answers are interchangeable. Being considered as one of the core challenges for developing an effective QA system, question paraphrase identification is to verify whether a retrieved candidate question is a paraphrase of the input question. Able to determine if two questions have the same intent can significantly improve the candidate answer ranking performance. However, it is a challenging task even for human experts. Questions (especially from online forums) range from highly technical subjects to ambiguous thoughts. The question texts are also not always grammatically correct and often contain misspelled words. Sometimes online user-generated questions might also contain additional contexts that irrelevant to its intent.

In this chapter, we address the problem of question paraphrase identification which predicts a binary label for whether two input questions convey the same meaning. We introduce a new neural dual entailment model for the question paraphrase identification task. This method uses bidirectional recurrent neural networks to encode the premise question into phrase vectors, and then fetch the corresponding softly aligned phrases embedding from the candidate question with the attention mechanism. The final similarity score is produced based on aggregated phrase-wise comparisons of both entailment directions. Our model is fully symmetric, and then parameters are shared on both sides, which allows the model parameters can be trained efficiently even on the medium-sized dataset. We evaluated our model on the Quora Question Pairs dataset.[1] Experimental results show that our model achieves the state-of-the-art performance on this benchmark dataset.

---

[1] `https://data.quora.com/First-Quora-Dataset-Release-Question-Pairs`

5.2 Related Work

Several NLP tasks take two text passages as input and predict a label for their relationship. Question paraphrase identification has a strong resemblance of Natural Language Inference (NLI; or called Recognizing Textual Entailment) task. For instance, the SNLI dataset (Bowman et al., 2015) consists of manually generated premise-hypothesis sentence pairs and their relationship labels. The relation includes: (a) entailment, if the premise sentence entails the hypothesis sentence, (b) contradiction, if two sentences are contradicting each other, (c) neutral, if two sentences neither entail nor contradict. Question paraphrase identification task is a variant of NLI that focus on predicting whether the intent of two questions entails each other.

Recently several neural-based models have been developed and have been tested in question paraphrase identification or question retrieval setup. Wang and Jiang (2016) presented a class of neural structures that performs word-level matching followed by a convolutional aggregation. Wang et al. (2017b) use BLSTMs to encoder the input sequence pair, computes the similarity in multiple perspectives, and aggregate matching vectors with another BLSTM. Qiu and Huang (2015) adopts a Siamese convolutional network to encode sequence into contextual representations and performs the final scoring on those representations. Das et al. (2016) used twin convolutional neural network to encode and a contrastive loss function, which joins the twin networks.

5.3 Question Paraphrase Identification with Neural Dual Entailment

5.3.1 Problem Formulation

Let $\mathbf{w}^x = (w_1^x, \ldots, w_{\ell^x}^x)$ and $\mathbf{w}^y = (w_1^y, \ldots, w_{\ell^y}^y)$ be two input questions consisting of ℓ^x and ℓ^y words, respectively. Each word in both sentences, w_i^x or w_j^y, belongs to the vocabulary V_w. The training data is in the form of labeled text pairs $\{(\mathbf{w}^x, \mathbf{w}^y)^{(n)}, p^{(n)}\}_{n=1}^N$, where $p^{(n)} \in \{0, 1\}$ is a binary label indicating whether $\mathbf{w}^x$ is a paraphrase of $\mathbf{w}^y$. The goal is to predict the correct label p given a previously unseen text pair $(\mathbf{w}^x, \mathbf{w}^y)$.

5.3.2 Model Structure

As outlined in Figure 5.1, our model decomposes the paraphrase identification problem into four steps: *Encode*, *Attentional-Align*, *Matched-Aggregate*, and *Dual-Predict*.

Encode. The goal of this step is to obtain a dense representation of each word w_i in $\mathbf{w}^x$ and $\mathbf{w}^y$ that captures word meaning along with contextual information.

We first individually map each word w_i into a d-dimensional vector by a mixed em-

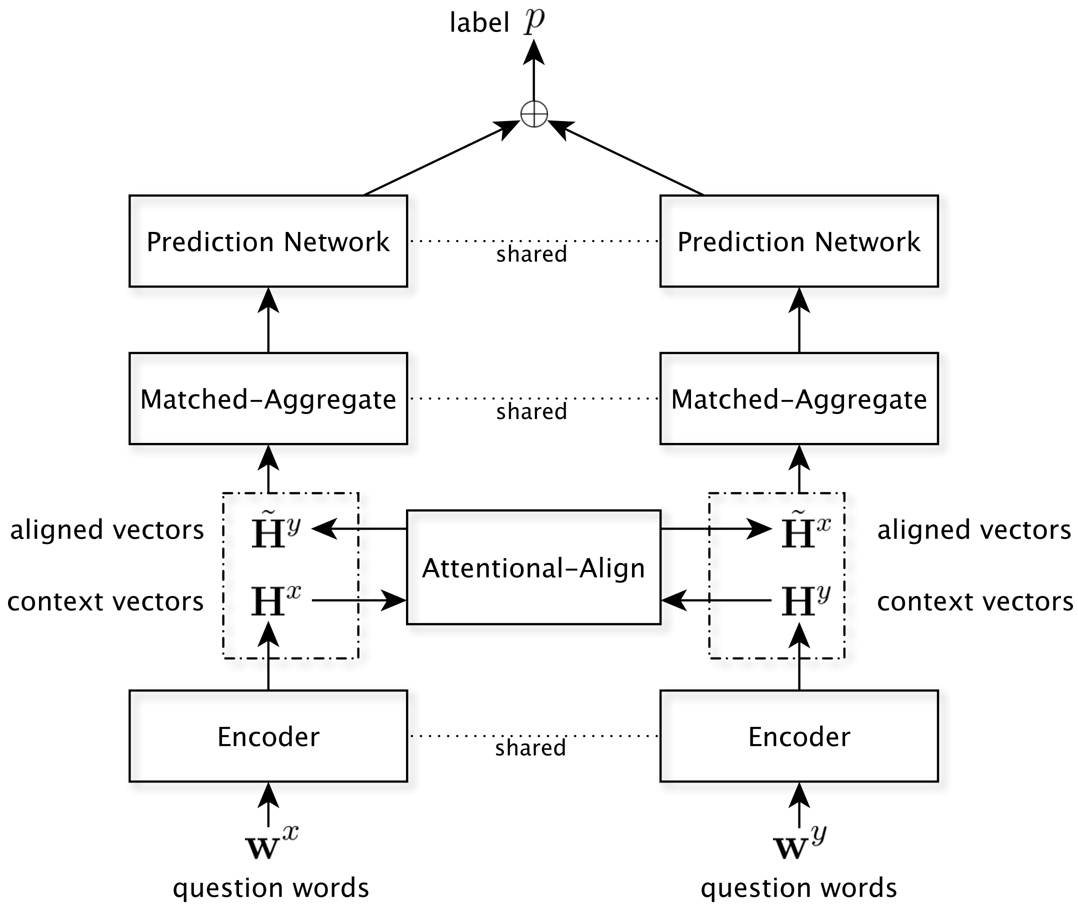


Figure 5.1: Model Structure of our Neural Dual Entailment Model

bedding matrix $\mathbf{E} \in \mathbb{R}^{|V_w| \times d}$, where $\mathbf{E} = \mathbf{E}_{pre} + \mathbf{E}_{tune}$. Here $\mathbf{E}_{pre}$ is a word embedding matrix that unsupervised pre-trained on a large corpus and will be fixed during the model training. On the other hand, $\mathbf{E}_{tune}$ is a trainable embedding matrix that is randomly initialized. This mixture setup aims to fine-tune the pre-trained embeddings to recognize domain-specific semantics of word usage, while avoiding deviating from the pre-trained representations too early.

Then the encoder converts the word embedding sequence into a list of contextual vectors $\mathbf{H} = \{\mathbf{h}_1, \mathbf{h}_2, \ldots, \mathbf{h}_{\ell^w}\}$, whose size varies with regard to the length of the passage. This context representation is generated using a two-layer stacked bidirectional recurrent neural networks (BRNN), which utilize the context of both sides. Specifically, the encoder BRNN processes the data from both directions with two separate hidden sublayers, where

$$\begin{aligned} \overrightarrow{\mathbf{h}_i} &= \overrightarrow{\Psi}\left(\overrightarrow{\mathbf{h}}_{i-1}, \mathbf{E}\left[w_i\right]\right) \\ \overleftarrow{\mathbf{h}_i} &= \overleftarrow{\Psi}\left(\overleftarrow{\mathbf{h}}_{i+1}, \mathbf{E}\left[w_i\right]\right). \end{aligned} \tag{5.1}$$

Here Ψ is a recurrent activation unit that we employ in the Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997). The output of the current time step is then generated by concatenating both direction's hidden vector $\overrightarrow{\mathbf{h}_i}$ and $\overleftarrow{\mathbf{h}_i}$. The stacked upper layer BRNN treats the output from the lower layer $\mathbf{h}_i^1$ as the input, and further outputs $\mathbf{h}_i^2$. Finally, the encoded contextual vector is aggregated as the sum of embedding and outputs from both layers:

$$\mathbf{h}_i = \mathbf{E}\left[w_i\right] + \mathbf{h}_i^1 + \mathbf{h}_i^2. \tag{5.2}$$

Attentional-Align. Regarding each $\mathbf{h}_i$, this step attempts to gather a softly aligned context vector from the other sentence's encoded sequence $\bar{\mathbf{H}}$. It starts by computing the content-based score of each pair of context vectors as:

$$e_{i,j} = \mathbf{h}_i^\top \bar{\mathbf{h}}_j. \tag{5.3}$$

This score measures similarity between $\mathbf{h}_i$ and the j-th context vector of $\bar{\mathbf{H}}$. These relevance scores are further normalized by the softmax function:

$$\alpha_{i,j} = \mathrm{softmax}(e_{i,j}) = \frac{\exp(e_{i,j})}{\sum_{k=1}^{\ell} \exp(e_{i,k})}, \tag{5.4}$$

and we call $\alpha_{i,j}$ the attention weight. The softly aligned phrase vector $\tilde{\mathbf{h}}_i$ is then the weighted sum of the context vectors with their attention weights from above:

$$\tilde{\mathbf{h}}_i = \sum_{j=i}^{\ell} \alpha_{i,j} \bar{\mathbf{h}}_j. \tag{5.5}$$

Matched-Aggregate. Given the $\mathbf{H}$ and its softly aligned $\tilde{\mathbf{H}}$, the problem has been reduced to comparing a sequence of aligned phrase vectors. Each pair of aligned vectors are first merged into a matched vector m_i with a Multi-Layer Perceptron M:

$$m_i = M\left(\left[\mathbf{h}_i; \tilde{\mathbf{h}}_j; (\mathbf{h}_i - \tilde{\mathbf{h}}_j)^2\right]\right). \tag{5.6}$$

The list of matched vectors $\{m_1, \ldots, m_\ell\}$ are later scanned by an RNN layer, and its output of the last timestamp is served as the final aggregated vector.

Dual-Predict. The prediction network consists of a two-layer batch-normalized multilayer perceptron (MLP), MaxOut neurons (Goodfellow et al., 2013) , and a linear layer to produce an entailment score based on the matched aggregate vector above. As yet, the *Attentional-Align*, *Matched-Aggregate*, and the prediction network can only to check whether the information of one sentence is phrase-wise covered by another sentence. To identify paraphrase, we use the shared network to calculate entailment scores from both directions of question pairs. Finally, two scores are summed and followed by a logistic layer, to predict the label p.

Method	Dev Acc.	Test Acc.
Siamese-CNN	—	79.60
Multi-Perspective CNN	—	81.38
Siamese-LSTM	—	82.58
Multi-Perspective-LSTM	—	83.21
L.D.C	—	85.55
BIMPM	88.69	88.17
pt-DECATT$_{\text{word}}$	88.44	87.54
pt-DECATT$_{\text{char}}$	88.89	88.40
Ours model	88.61	**88.92**

Table 5.1: Evaluation results on the Quora paraphrase identification dataset in terms of accuracy.

5.4 Experiments

Datasets

We evaluated our models on the Quora question paraphrase dataset which contains over 404,000 question pairs with binary labels. The dataset has approximately 37% positive and 63% negative pairs. We use the same data split and tokenization provided by (Wang et al., 2017b)[2]. Each of the development and test datasets has 5,000 positive and 5,000 negative instances. The remaining instances are then used as the training set. We further augment the training set with 20,000 positive pairs with identical question pairs to make sure the model can generalize well on the easy situation as well.

Network Setup and Parameter Optimization

The embeddings $\mathbf{E}_{pre}$ was initialized with the 300-dimensional GloVe (Pennington et al., 2014) word vectors pre-trained from the 840B Common Crawl corpus. Out-of-vocabulary words were initialized as zeros. Our encoder RNN contains two-layer stacked LSTMs. Each layer of LSTM has a memory size of 300. The MLP networks used ReLU as activation functions and the dropout rate of 0.2. The network weights are randomly initialized using a uniform distribution $(-0.08, 0.08)$, and are trained with the ADAM optimizer (Kingma and Ba, 2014), with an initial learning rate of 0.002. Gradients were clipped so their norm does not exceed 5. Each mini-batch contains 1000 question pairs.

Results

In this section, we compare the performance of our approach with the baseline models, bilateral multi-perspective matching (BiMPM) (Wang et al., 2017b) model, and Paralex pre-trained decomposable attention (pt-DECATT) (Tomar et al., 2017) models, which

---

[2] This partition of Quora dataset can be downloaded at `https://zhiguowang.github.io`.

have benchmarked on this dataset before. The BiMPM model employs both character-level and word-level LSTMs to represent input, four different types of multi-perspective matching functions, a bi-LSTM aggregation layer, and an MLP with softmax output for final prediction. The pt-DECATT model uses sums of character n-gram embeddings to represent words, both bi-attention, and self-attention to softly align phrase vectors, an MLP followed by sum as aggregation layer, and another MLP with linear output for prediction. pt-DECATT also utilized Paralex (Fader et al., 2013) question paraphrase corpus to pre-train both its character n-gram embeddings and rest of the model.

Table 5.1 shows the accuracy results of baseline models implemented in (Wang et al., 2017b), BiMPM model, pt-DECATT model, and our method. The first eight rows are taken from (Wang et al., 2017b; Tomar et al., 2017). From the results, we can see that our model can outperform the previous best performance on the test set. Notes that our model is also simpler since it does not employ character encoding, self-attention, multi-perspective matching, and Paralex pre-training as required in previous methods.

5.5 Summary

We showed that efficient parameter sharing and mixed word embeddings result in state-of-the-art accuracy on question paraphrase identification task even without complex neural architectures, character-level embeddings, or pre-training the full model. The new matching module also show its robustness when integrated into our LiveQA system in Chapter 7, and helped our system achieved the highest average score among automatic systems in both main task and medical subtask.

Chapter 6

Answer Passage Selection

In this chapter, we present an approach that addresses the answer sentence selection problem for question answering (Wang and Nyberg, 2015a,b,c, 2016). The proposed method uses a stacked bidirectional Long-Short Term Memory (BLSTM) network to sequentially read words from the question and answer sentences, and then outputs their relevance scores. Unlike prior work, this approach does not require any syntactic parsing or external knowledge resources such as WordNet which may not be available in some domains or languages. The full system is based on a combination of the stacked BLSTM relevance model and keywords matching. The results of our experiments on a public benchmark dataset from TREC show that our system outperforms previous work which requires syntactic features and external knowledge resources.

6.1 Introduction

A typical architecture of open-domain question answering (QA) systems is composed of three high-level major steps: (a) question analysis and retrieval of candidate passages; (b) ranking and selecting of passages which contain the answer; and optionally (c) extracting and verifying the answer (Prager, 2006; Ferrucci, 2012). In this paper, we focus on the answer sentence selection. Being considered as a key subtask of QA, the selection is to identify the answer-bearing sentences from all candidate sentences. The selected sentences should be relevant to and answer the input questions.

The nature of this task is to match not only the words but also the meaning between question and answer sentences. For instance, as shown in Table 6.1, although both of the candidate sentences contain keywords "Capriati" and "play", only the first sentence answers the question.

Besides its application in the automated factoid QA system, another benefit of the answer sentence selection is that it can be potentially used to predict answer quality in community QA sites. The techniques developed from this task might also be beneficial to the emerging real-time user-oriented QA tasks such as TREC LiveQA. However,

Question:	What sport does Jennifer Capriati play?
Positive:	Capriati, 19, who has not played competitive **tennis** since November 1994, has been given a wild card to take part in the Paris tournament which starts on February 13.
Negative:	Capriati also was playing in the U.S. Open semifinals in '91, one year before Davenport won the junior title on those same courts.

Table 6.1: Example answer candidate sentences for the question "What sport does Jennifer Capriati play?" from TREC QA 2004

user-generated content can be noisy and hard to parse with off-the-shelf NLP tools. Therefore, methods that require less syntactic features are desirable.

Recently, neural network-based distributed sentence modeling has been found successful in many natural language processing tasks such as word sense disambiguation (McCarthy et al., 2004), discourse parsing (Li et al., 2014), machine translation (Sutskever et al., 2014; Cho et al., 2014), and paraphrase detection (Socher et al., 2011).

We present an approach that employs stacked bidirectional Long Short-Term Memory (BLSTM) to sequentially read the words from the question and answer sentences, and then output their relevance scores. The full system, when combined with keywords matching, outperforms previous approaches without using any syntactic parsing or external knowledge resources.

6.2 Related Work

Prior to this work, there were other approaches to address the sentence selection task. The majority of previous approaches focused on syntactic matching between questions and answers. Punyakanok et al. (2004) and Cui et al. (2005b) were among the earliest to propose the general tree matching methods based on tree-edit distance. Subsequent to these two papers, Wang et al. (2007b) use quasi-synchronous grammar to match each pair of question and sentence by their dependency trees. Later, tree kernel functions together with a logistic regression model (Heilman and Smith, 2010) or Conditional Random Fields models (Wang and Manning, 2010; Yao et al., 2013) with extracted features were adopted to learn the associations between question and answer. Recently, discriminative tree-edit feature extraction and engineering over parse trees are automated in (Severyn and Moschitti, 2013).

Besides syntactic approaches, lexical semantic model (Yih et al., 2013) can be also used to select answer sentences. This model is to pair semantically related words based on word relations including synonymy/antonymy, hypernymy/hyponymy, and general semantic word similarity.

There were also prior efforts in deep learning of neural networks for question answering. Yih et al. (2014) focused on answering single-relation factual questions by a semantic similarity model using convolutional neural networks. Bordes et al. (2014a) jointly embedded words and knowledge base constituents into the same vector space to measure the relevance of question and answer sentences in that space. Iyyer et al. (2014) worked on the quiz bowl task, which is an application of recursive neural networks for factoid question answering over paragraphs. The correct answers are identified from a relatively small fixed set of candidate answers which are in the form of entities instead of sentences.

6.3 Answer Sentence Selection with Stacked BLSTM

6.3.1 Network Structure

The goal of this system is to reduce as much as possible the dependency on syntactic features and external resources by leveraging the power of deep recurrent neural network architecture. The proposed network architecture is trained directly on the word sequences of question and answer passages, and is actually not limited to sentences.

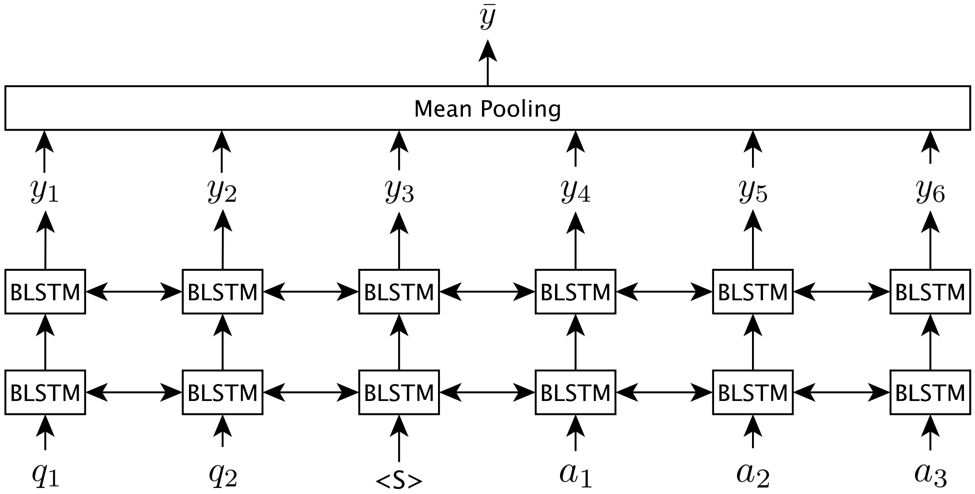


Figure 6.1: An illustration of our QA sentence relevance model based on stacked BLSTM

As per analysis in Section 2.2, we adopt multi-layer stacked bidirectional LSTM RNNs (rather than conventional RNNs) to model the answer sentence selection problem as illustrated in Figure 6.1. The words of input sentences are first converted to vector representations learned from the word2vec tool (Mikolov et al., 2013). In order to differentiate question q and answer a sentences, we insert a special symbol, `<S>`, after the question sequence. Then, the question and answer sentences word vectors are sequentially read by BLSTM from both directions. In this way, the contextual information across words in both question and answer sentences is modeled by employing temporal recurrence in BLSTM.

Since the LSTM in each direction carries a cell memory while reading the input sequence, it is capable of aggregating the context information and storing it into the cell memory vector. For each time step in the BLSTM layer, the hidden vector or the output vector is generated by combining the cell memory vectors from two LSTM of both sides. In other words, all the contextual information across the entire sequence (both question and answer sentences) has been taken into consideration. The final output of each time step is the label indicating whether the candidate answer sentence should be selected as the correct answer sentence for the input question. This objective encourages the BLSTMs to learn a weight matrix that outputs a positive label if there is overlapping context information between two LSTM cell memories. Mean pooling is applied to all the time step outputs during the training. During the test phase, we collect mean, sum and max poolings as features.

6.3.2 Incorporating Keywords Matching

In order to identify the correct candidate answer sentences, it is crucial to match the cardinal numbers and proper nouns with those occurring in the question. However, many cardinal numbers and proper nouns are out of the vocabulary (OOV) of our word embeddings. In addition, some proper nouns' embeddings may bring noises to the matching process. For example, "Japan" and "China" are two words very close in the embedding space. It is critical to discriminate these two proper nouns when matching question and answer sentences. In order to mitigate this weak point of the distributed representations, our full system combined the stacked BLSTM relevance model and exact keywords overlapping baseline by the gradient boosted regression tree (GBDT) method (Friedman, 2001).

6.3.3 Experiments on Factoid QA

Dataset

The answer sentence selection dataset used in this paper was created by Wang et al. (2007b) based on Text REtrieval Conference (TREC) QA tracks (8–13) data.[1] Candidate answer sentences were automatically retrieved for each question which is on average associated with 33 candidate sentences. There are two sets of data provided for training. One is the full training set containing 1229 questions that are automatically labeled by matching answer keys' regular expressions.[2] However, the generated labels are noisy and sometimes erroneously mark unrelated sentences as the correct answers solely because those sentences contain answer keys. Wang et al. (2007b) also provided one small training set contains 94 questions, which were manually corrected for errors. In our experiments, we use the full training set because it provides significantly more question

---

[1] `http://nlp.stanford.edu/mengqiu/data/qg-emnlp07-data.tgz`

[2] Because the original full training dataset is no longer available from the website of the lead author of (Wang et al., 2007b), we obtained this data re-released from Yao et al. (2013): `http://cs.jhu.edu/~xuchen/packages/jacana-qa-naacl2013-data-results.tar.bz2`

and answer sentences for learning, even though some of its labels are noisy.

The development and test data sets have 82 and 100 questions, respectively. Following (Wang et al., 2007b), candidate answer sentences with over 40 words and questions with only positive or negative candidate answer sentences are removed from the evaluation.[3]

Evaluation Metric

Following previous works on this task, we also use Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR) as evaluation metrics, which are calculated using the official *trec_eval* evaluation scripts.

Keywords Matching Baseline

As noted by Yih et al. (2013), counting overlapped keywords, especially when re-weighted by idf value of the question word, is a fairly competitive baseline. Following (Yih et al., 2013), our keywords matching baseline also counts the words that occurred in both questions and answer sentences, after excluding stop words and lowering the case. But, instead of the $tf \cdot idf$ formula used in (Yih et al., 2013), word counts are re-weighted by idf value using the Okapi BM25 (Robertson and Walker, 1997) formula (with constant values $K_1 = 1.2$ and $B = 0.75$).

Network Setup

The network weights are randomly initialized using a Gaussian distribution ($\mu = 0$ and $\sigma = 0.1$), and the network is trained with the stochastic gradient descent (SGD) with momentum 0.9. We experimented with single-layer unidirectional LSTM, single-layer BLSTM, and three-layer stacked BLSTM. Each layer of LSTM and BLSTM has a memory size of 500. We use 300-dimensional vectors that were trained and provided by the word2vec tool (Mikolov et al., 2013) using a part of the Google News dataset[4] (around 100 billion tokens).

Results

Table 6.2 surveys prior results on this task, and places our models in the context of the current state-of-the-art results.

Table 6.3 summarizes the results of our model on the answer selection task. Features are keywords matching baseline score (BM25), and pooling values of single-layer unidirectional LSTM (Single-Layer LSTM), single-Layer bidirectional LSTM (Single-Layer

---

[3] As mentioned in the footnote 7 of (Yih et al., 2013): *"Among the 72 questions in the test set, 4 of them would always be treated answered incorrectly by the evaluation script used by previous work. This makes the upper bound of both MAP and MRR become 0.9444 instead of 1."* In order to make experiment results comparable with previous works, we also use this experiment setting.

[4] `https://code.google.com/p/word2vec/`

Reference	**MAP**	**MRR**
Yih et al. (2013) — Random	0.3965	0.4929
Wang et al. (2007b)	0.6029	0.6852
Heilman and Smith (2010)	0.6091	0.6917
Wang and Manning (2010)	0.5951	0.6951
Yao et al. (2013)	0.6307	0.7477
Severyn and Moschitti (2013)	0.6781	0.7358
Yih et al. (2013) — BDT	0.6940	0.7894
Yih et al. (2013) — LCLR	0.7092	0.7700

Table 6.2: Overview of prior results on the answer sentence selection task

Features	MAP	MRR
BM25	0.6370	0.7076
Single-Layer LSTM	0.5302	0.5956
Single-Layer BLSTM	0.5636	0.6304
Three-Layer BLSTM	0.5928	0.6721
Three-Layer BLSTM + BM25	**0.7134**	**0.7913**

Table 6.3: Overview of our results on the answer sentence selection task.

BLSTM) and three-Layer stacked BLSTM's (Three-Layer BLSTM) outputs. Gradient boosted regression tree (GBDT) method is used to combine features. According to Table 6.2 and 6.3, our combined system outperforms prior works on MAP and MRR metrics.

As indicated in Table 6.3, the three-layer stacked BLSTM alone shows better experiment results than single-layer BLSTM and unidirectional LSTM, and performs comparably to previous systems. In order to mitigate the weak point of the distributed representations previously discussed in Section 6.3.2, we combine the stacked BLSTM outputs with a keywords matching baseline (BM25). Our combined system's results are statistically significantly better than the keywords matching baseline (using the Student's t-test with $p < 0.05$) and outperforms previous state-of-art results.

6.3.4 Experiments on Non-Factoid QA

Dataset

In this experiment, we used the Yahoo! Answer Manner Question dataset, specifically version 2[5], to evaluate our ranking model. This dataset was collected by Surdeanu et al. (Surdeanu et al., 2008) and was based on a dump of Yahoo! Answers corpus in 2007. Only manner questions, such as sentences starting with "how to", are selected from the full dump. In addition, questions and answers with obvious low quality are filtered and excluded. The resulting subset contains 142,627 questions and their user selected "best answers".

In order to evaluate our ranking model with this dataset, we used the following experiment setup:

1. We first indexed all the "best answers" text while keeping their source question IDs.
2. We used each question's text as the input query, and then collected outputs from a bag-of-words retrieval baseline. Specifically, the BM25 retrieval model was used to collect a pool of candidate answer passages.
3. All collected candidate answer passages were then automatically labeled by comparing their source question IDs with input question IDs.[6]
4. Since we mainly focused on ranking performance, only a subset of questions, whose original best answers appeared in the candidate pool, were further evaluated.
5. 60% of the questions were used for training, 20% for development, and the rest for testing.
6. We used MRR (Mean Reciprocal Rank) and P@1 (Precision at one) as the evaluation metrics, which were calculated using the official *trec_eval* evaluation scripts.

Network Setup

The network weights were randomly initialized using a Gaussian distribution ($\mu = 0$ and $\sigma = 0.1$), and were trained by stochastic gradient descent (SGD) with momentum 0.9. Our experiment network contains three-layer stacked BLSTMs. Each BLSTM has a memory size of 500. We used 300-dimensional vectors that were trained and provided by word2vec tool (Mikolov et al., 2013) using a part of the Google News dataset[7] (around 100 billion tokens).

Method	Recall@25	P@1	MRR
BM25	29.96%	37.99%	51.57
BLSTM+BM25		**42.77%**	**56.40**

Table 6.4: Overview of our results on the answer retrieval and ranking on Yahoo! Answer Manner Questions dataset.

Results

Table 6.4 summarizes our preliminary experiment results on re-ranking the retrieved top 25 passages by BM25 model. Gradient boosted regression tree (GBDT) method is used to combine pooling values of stacked BLSTMs and BM25 retrieval score. It demonstrates that incorporating our answer passage ranking model with baseline retrieval method leads to considerable improvements in ranking accuracy.

6.4 Answer Passage Selection with Attentional Encoder-Decoders

RNN-based encoder-decoder models have been applied to machine translation and quickly achieved state-of-the-art results (Bahdanau et al., 2014; Luong et al., 2015). Since RNN models do not depend on any external feature or knowledge, we adopted a neural encoder-decoder model, and trained it to "translate" from an answer passage to the question. We later used the trained model to provide the relevance score between a question and answer based on the likelihood of their "translation". The model structure of our attentional encoder-decoder can be found in Section 2.2.1.

In the previous section, we employed a recurrent neural network-based approach (Wang and Nyberg, 2015a,b) that uses a multi-layer stacked bidirectional Long-Short Term Memory (BLSTM) network to sequentially read words from question and answer passages, and then output their relevance scores. Because training this BLSTM model requires both positive and negative examples, we followed the same data preparation procedure described by Surdeanu et al. (2011) to generate negative labels by retrieving other answer passages from the collection. However, since the generated labels contain false negatives, the model may potentially learn low weights for instances with false-negative training labels and decrease overall performance. On the other hand, training the attentional neural encoder-decoder model needs only positive pairs, therefore this model will not suffer from the above problem anymore.

Our encoder and decoder RNNs contain two-layer stacked LSTMs. Each layer of LSTM has a memory size of 500. The network weights are randomly initialized using a uni-

---

[5] `http://webscope.sandbox.yahoo.com/catalog.php?datatype=l`

[6] Note that the generated labels will contain false negatives.

[7] `https://code.google.com/p/word2vec/`

form distribution $(-0.08, 0.08)$, and are trained with the ADAM optimizer (Kingma and Ba, 2014), with an initial learning rate of 0.002. Gradients were clipped so their norm does not exceed 5. Each mini-batch contains 200 answer and question pairs.

We use the Yahoo! Answers Comprehensive Questions and Answers dataset[8] for training, which contains around 4.4 million Yahoo! Answers questions and their best answers. The words of input sentences were first converted to 300-dimensional vector representations learned from RNN based language modeling tool word2vec (Mikolov et al., 2013). Each passage's beginning and end are also padded with a special boundary symbol, `<S>`.

6.4.1 Rank Scoring

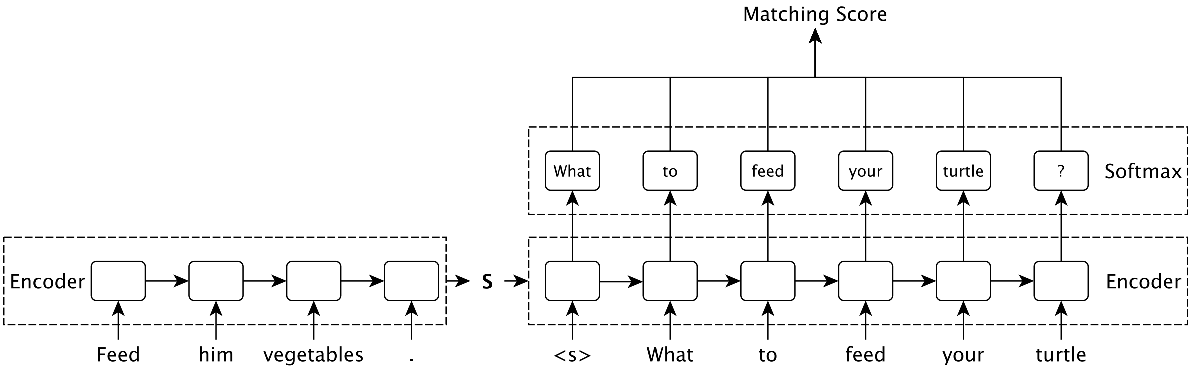


Figure 6.2: An illustration of Answer Passage Selection with Attentional Encoder-Decoders

As shown in Figure 6.2, at test time, instead of finding the best-scoring "translation", the decoder is fed with the original question as input and calculates the perplexity that the model predicts regarding question words:

$$score(Q, A^{(i)}) = \exp(-\frac{1}{T_Q}\sum_{t=1}^{T_Q} \log p(Q_t = Q_t | Q_{<t}, A^{(i)}; \hat{\theta})).$$

6.4.2 Experiments

The LiveQA organizers provided scored answers from all TREC 2015 LiveQA submissions[9], which we used as a development dataset for analyzing the performance of answer passage ranker. Since we are only evaluating the performance of ranking, questions with only negative candidate answer passages are removed from evaluation. This

---

[8] `http://webscope.sandbox.yahoo.com/catalog.php?datatype=l`

[9] https://sites.google.com/site/trecliveqa2016/liveqa-qrels-2015/LiveQA2015-qrels-ver2.txt.gz

Methods	NDCG	MAP@			MRR@		
		2+	3+	4+	2+	3+	4+
Lower Bound	0.3924	0.2225	0.1232	0.0519	0.0800	0.0524	0.0274
Upper Bound	1.0000	1.0000	0.7977	0.4668	1.0000	0.7977	0.4668
BM25	0.5636	0.4307	0.2631	0.1205	0.4303	0.2555	0.1174
Encoder-Decoder	**0.6346**	**0.5124**	**0.3390**	**0.1657**	**0.5645**	**0.3672**	**0.1779**

Table 6.5: Results on re-ranking submitted answers from TREC LiveQA 2015

development dataset contains 949 questions. On average, there are 19.5 answer passage candidates for each question and 32% of candidates' score are above "Fair".

In order to validate our new ranker's performance with this dataset, we re-rank above development dataset with our ranker and a BM25 baseline ranker. NDCG (Normalized Discounted Cumulative Gain) with graded relevance scale 0–3, MAP (Mean average precision), and MRR (Mean Reciprocal Rank) were then used as evaluation metrics (calculated using the official *trec_eval* evaluation scripts).

Table 6.5 summarizes our preliminary experimental results on the Yahoo! Answers question retrieval and ranking task. Although good performance on this dataset does not necessarily correlate to good performance on the LiveQA 2016 challenge, it does demonstrate the necessity of developing non-trivial candidate retrieval and answer ranking methods for any LiveQA-related task.

6.5 Summary

In this chapter, we presented approaches utilizing stacked BLSTM and encoder-decoder to address the answer passages ranking and selection problem for web question answering. The experiments provide strong evidence that distributed and symbolic representations encode complementary types of knowledge, which are all helpful in identifying answer sentences. Based on the experiment results, we found that our model not only performs better than previous work but most importantly does not require any syntactic features or external resources.

Chapter 7

Application to TREC LiveQA (2015, 2016, 2017)

In this chapter, we present CMU OAQA's automatic, web-based, real-time question answering (QA) systems that were evaluated in the TREC 2015, 2016, and 2017 LiveQA Challenges (Wang and Nyberg, 2015c, 2016, 2017). This system answers real-user questions freshly submitted to the Yahoo! Answers website that have not been previously answered by humans. Given the title and body of the question, we generated multiple sets of keyword queries and retrieved a collection of web pages based on those queries. Then we extracted answer candidates from web pages in the form of answer passages and their associated clue. Finally, we combined both IR- and NLP-based relevance models to rank and select answer candidates. Overall, our approach received the highest average scores among automatic systems in the main tasks of 2015, 2016 and 2017, and also the highest average score in the new medical subtask of 2017.

7.1 Introduction

LiveQA is an emerging TREC challenge for automatic, real-time, user-oriented question answering (QA) systems. Real user questions are selected as inputs from a stream of newly submitted questions on the Yahoo! Answers site[1], which have not yet received a human answer. As per the requirements for this track, participants must deploy their systems as web services which provide answers to questions in real time (within one minute). The answer text should be no more than 1000 characters in length. There are no additional restrictions on the resources that can be used, except for the human answers to the same question in Yahoo! Answers. System responses were judged by TREC assessors on a 4-level Likert scale.

Attempting to answer user-generated questions in real time is an exciting but difficult challenge. The questions generated by real users are often ambiguous, ungrammatical

---

[1] `https://answers.yahoo.com/`

and vary greatly in length, topic and language style. Unlike previous TREC QA tracks, there is no pre-defined question type (factoid, list, or definition) associated with the question/answer pairs in the LiveQA challenge. There is also no constraint on what the user may ask about, which implies that there is no pre-defined answer type; much like the Jeopardy! challenge, we must deal with "lexical answer types" (Lally et al., 2012).

In TREC 2015 (Agichtein et al., 2015), we developed a real-time web-based QA system for the LiveQA challenge. Since there is no official training corpus associated with the challenge, our approach leveraged the vast amount of text data that is available online, especially previously-answered questions from Yahoo! Answers. We also designed and implemented a new data model and novel relevance ranking methods for LiveQA. Our QA pipeline begins with candidate passages retrieval and extraction, then answer candidate ranking and tiling. During the 2015 official run, our QA web service received one question per minute for 24 hours and provided answers within one minute for 97.9% of the input questions. On a normalized three-point average score metric, the CMU-OAQA system received a score of 1.081, which was the top score in the 2015 LiveQA evaluation (the second-best average score was .677, and the average over all the system runs was .467 for the 21 systems evaluated).

In TREC 2016 (Agichtein et al., 2016), we improved our LiveQA system and introduced a novel answer passage ranking method based on attentional encoder-decoder recurrent neural networks (RNN). Being considered as a primary challenge of developing an effective QA system, the answer passages ranking and selection are to identify the answer-bearing passages from all candidate passages. The selected passage should contain useful information, and answer the input question. Our method uses one RNN to encode candidate answer passage into vectors, and then another RNN to decode the input question from the vectors. The perplexity of decoding the question is then used as the ranking score. During the 2016 official run, our QA server received one question per minute for 24 hours and provided answers within one minute for 94% of the input questions. In the TREC 2016 LiveQA evaluations, human assessors gave our system an average score of 1.1547 on a three-point scale, and the average score was .5766 for all the 26 systems evaluated.

In TREC 2017 (Ben Abacha et al., 2017), the main improvement is the introduction of our new question paraphrase identification module based on a neural dual entailment model. The question paraphrase identification module in QA system is to verify whether a retrieved candidate question is a paraphrase of input question. Two questions are paraphrases of each other if they can be adequately answered by the same answer. The model uses bidirectional recurrent neural networks to encode the premise question into *phrase* vectors, and then *align* corresponding phrase vectors from the candidate question with the attention mechanism. The final similarity score is produced based on aggregated phrase-wise comparisons of both entailment directions. In previous years, the test collection came from a stream of questions that freshly submitted to the Yahoo! Answers website and have not been previously answered by humans. This year, due to technical difficulties, the participant systems were fed with cached ques-

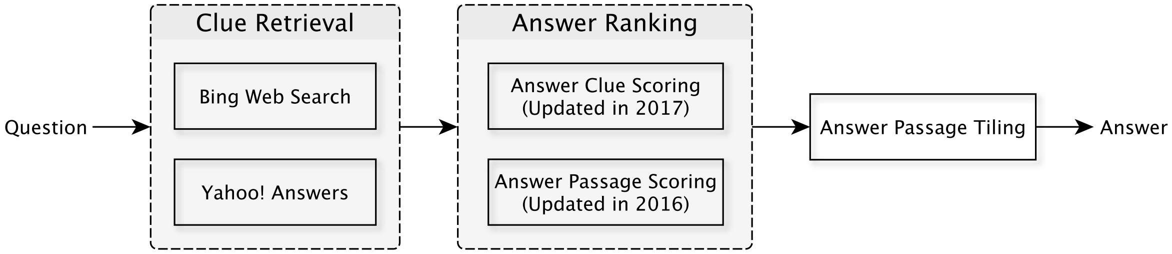


Figure 7.1: Architecture of the CMU-OAQA LiveQA system

tions instead. During the official run, our QA server received one question per minute for 24 hours and provided answers within one minute for 98% of the input questions. In the TREC 2017 LiveQA evaluations, human assessors gave our system an average score of 1.139 on a three-point scale, and the median score was 0.777 for all the systems evaluated. A new subtask explicitly focused on medical questions is also offered this year. Our system answered medical questions the same way as the main task, and also achieved the highest average score of 0.637 comparing to the median score of 0.431.

In the rest of this chapter, we will sketch the OAQA LiveQA system structure, and then introduce each module in more detail.

7.2 System Architecture

Our overall system architecture remains the same in 2015, 2016, and 2017. As illustrated in Figure 7.1, the architecture of our system decomposes the solution into three major processing phases:

1. **Clue Retrieval**. Given a question title and its full-text description, we formulate search engine queries and issue them to different search engines (Bing Web Search, Yahoo! Answers) in order to retrieve web pages related to the question.
2. **Answer Ranking**. Answer candidates (title/body/answer tuples that represent either conceptional questions or answer texts) are extracted from web pages, and ranked based on a relevance estimator. The most effective relevance estimator we found was a heuristically-weighted combination of:
    (a) optimized BM25 similarity scoring over the title and body texts,
    (b) an attentional encoder-decoder recurrent neural networks model (Wang and Nyberg, 2016; Wang et al., 2017a) that estimates the relevance of a candidate answer text given a question text, and
    (c) a novel neural dual entailment based question paraphrase identification model that predicts the relevance between input question and titles of answer candidates.

3. **Answer Passage Tiling**. Finally, a simple greedy algorithm is used to select a subset of highest-ranked answer candidates; these are simply concatenated without further processing in order to produce the final answer.

7.2.1 Input

Each question q transmitted to a LiveQA web service consists of four parts: title q_title, body q_body, category, and a unique question ID. Our system does not make use of the question category, and only uses the question ID to filter out retrieved Yahoo! Answers pages with the same ID.

7.2.2 Clue Retrieval

Given a question, firstly we formulate multiple queries, then send them to the search engines, thirdly collect best matching web pages for each, and finally harvest a set of candidate answers for further processing.

The input question $q = \langle q\_title, q\_body \rangle$ is converted to sets of keyword queries. In order to achieve higher recall, we generate multiple sets of keywords from q_title, q_body, and the concatenation of both. However, the inclusion of full q_body text may result in very long queries, for which the search engines do not return any result. Therefore we also generate keyword queries with only informative noun phrases, bigrams, and unigrams. The different keyword queries are then sent to search engines for web retrieval. We collect the snippets returned for each hit and also download the full text of the web page for each hit.

Our system utilizes Bing's web search API[2] to retrieve answer candidates from the Web. Since Shtok et al. (2012) showed that a significant amount of the unresolved Yahoo! Answers questions can be satisfactorily answered by reusing a best answer from the past, our system also uses the search function on Yahoo! Answers to collect additional answer candidates.

Our goal here is to search the Internet and select passages of text, containing information relevant to the question and indicating the presence of a candidate answer passage on the same page. For this purpose, we build a general data structure called the *answer clue* (a_clue), represented as $\langle c\_title, c\_body \rangle$. This formulation can be used to represent (and effectively combine) either the title and body of a similar question on community QA (cQA) sites like Yahoo! Answers, or a document title and search snippet extracted from a web page. Finally, we heuristically select answer passage text $a\_passage$ based on its relative position to a_clue. Our algorithm favors passages using heuristics such as: the voted best answer on cQA sites, the first reply on Internet forums, and paragraphs appearing in contexts which highly match the search snippet.

---

[2] `https://datamarket.azure.com/dataset/bing/searchweb`

7.2.3 Answer Ranking

Each candidate answer $a = \langle a\_clue, a\_passage \rangle$ is weighted by estimating how well both the a_clue and $a\_passage$ matches the question. The relevance ranking score between input question q and an answer candidate a is then defined as:

$$S(q,a) = S_c(q, a\_clue) \times (1 + w_p \times S_p^{'}(q, a\_passage))$$

where S_c and $S_p^{'}$ are answer clue score and normalized answer passage scores respectively, and w_p is the weight factor to merge them. The original answer passage score S_p is normalized by the max and min values ($S_p^{\max}$ and $S_p^{\min}$) in the collection of answer passage scores generated for the current input question:

$$S_p^{'} = \frac{S_p - S_p^{\min}}{S_p^{\max} - S_p^{\min}}$$

Question q and answer clue a_clue both contain two fields: title and body. Intuitively the title field is more concise and informative than the body field. The answer clue score S_c is, then, computed as a weighted sum of retrieval scores of all text and title only text:

$$S_c(q, a\_clue) = S_{bm25}(q\_title \oplus q\_body, c\_title \oplus c\_body) + w_t \times S_{bm25}(q\_title, c\_title)$$

where w_t (set to 1.0) is the weight to boost the title field matching score, and S_{bm25} is the Okapi BM25 (Robertson and Walker, 1997) formula (with parameters $K_1 = 1.0$ and $B = 0.75$). The idf and average document length values that are required to compute BM25 were obtained by indexing the Yahoo! Answers Comprehensive Questions and Answers dataset[3], which contains around 4.4 million Yahoo! Answers questions and their answers.

To integrate the term proximity information in our relevance estimation, we also use the sequential dependence variant of the Markov random field model (Metzler and Croft, 2005) to formulate weighted queries to the BM25 function. The weights for unigram, bigram, and proximity are 0.8, 0.1, and 0.1 respectively. We also expanded unigram queries with synonyms from WordNet (Miller, 1995b) when the query word had only one sense in WordNet.

To calculate the relevance score S_p between question and answer passage, we employ a recurrent neural network based approach (Wang and Nyberg, 2015a,b) that uses a multilayer stacked bidirectional Long-Short Term Memory (BLSTM) network to sequentially read words from the question and answer passages, and then output their relevance scores. Figure 6.1 illustrates how we use the stacked BLSTM to model the answer passage ranking and selection problem. The words of input sentences were first converted to vector representations learned from the RNN-based language modeling tool word2vec (Mikolov et al., 2013). In order to differentiate q_title and $a\_passage$ sentences,

---

[3] `http://webscope.sandbox.yahoo.com/catalog.php?datatype=l`

we inserted a special symbol, `<S>`, after the question sequence. Then, the question and answer sentence word vectors are sequentially read by BLSTM from both directions. In this way, the contextual information across words in both question and answer sentences is modeled by employing temporal recurrence in BLSTM.

We used the same experimental settings described by Wang and Nyberg (2015a) to train this model, except that the training dataset was switched to a random subset of 100k questions from the Yahoo! Answers Comprehensive QA dataset. In order to adapt this dataset for model training, we follow the same data preparation procedure described by Surdeanu et al. (2011) to generate negative labels by retrieving other answer passages from the collection. However, since the generated labels contain false negatives, the model may potentially learn low weights for instances with false negative training labels and decrease overall performance. To avoid this we set the combination weight $w_p = 0.1$.

We later improved the ranking by replacing S_p scoring with an attentional encoder-decoder model trained from Yahoo! Answers question-answer pairs in 2016, which has described in 6.4. In 2017, we trained an additional question paraphrase identification model to improve the answer clue scoring between input question and clue title. We trained an efficient dual entailment model using a Quora dataset and achieved state-of-the-art matching performance. Model details and experiments have been discussed in Chapter 5.

7.2.4 Answer Passage Tiling

Given a ranked list of answer passages, the system must produce an optimal text that answers the question. Good answers vary in length from short to long, and may contain a short reference to a specific entity or factoid, or a long narrative explanation; in general, the ideal answer length varies with the user's information need (as expressed by the question). Here we assumed that more extended and detailed answers are preferred.

If the top-scoring answer passage was longer than 500 characters, then the first 1000 characters (LiveQA length constraint) are directly returned as the final answer text. If not, we applied an answer tiling algorithm, which assembles longer answer texts out of shorter answer passages. The algorithm proceeds greedily from the top-scoring answer passages to all subsequent candidates whose ranking score is higher than 90% of the highest score (top decile). Additional answer passages are appended to the final answer text, with a prefix to indicate the start of each new answer passage; e.g. "Opinion 2:" for the second passage, and so on. The tiling algorithm terminates when the length of the generated answer passage is greater than 50% of the maximum allowable length (1000 characters).

	Set Recall	P@1	MRR
Yahoo! Answers default search	0.8464	0.5873	0.6434
Clue Retrieval + Answer Ranking	**0.9100**	**0.8950**	**0.9008**

Table 7.1: Results on development dataset

7.3 Evaluation Results

7.3.1 Development Set Analysis on Answer Ranking

The LiveQA organizers provided a sample of 1000 Yahoo! Answers question IDs (QIDs), which we used as a development set for pre-evaluation analysis. Specifically, we used each question's title text as an input query, and then collected outputs from the Yahoo! Answers default "Search Answer" function. In particular, a list of returned QIDs was collected via "Relevance" search (instead of "Newest", "Most Answers", or "Fewest Answers" search). The first 100 search results for each query were collected[4] and automatically labeled[5] by comparing their source question IDs with input question IDs.

In order to validate our system's performance with this dataset, we processed the same query set with our retrieval and ranking modules, and returned a ranked list of question IDs. MRR (Mean Reciprocal Rank) and P@1 (Precision at one) were then used as evaluation metrics (calculated using the official *trec_eval* evaluation scripts).

Table 7.1 summarizes our preliminary experimental results on the Yahoo! Answers question retrieval and ranking task. Although good performance on this dataset does not necessarily correlate to good performance on the LiveQA challenge, it does demonstrate the necessity of developing non-trivial candidate retrieval and answer ranking methods for any LiveQA-related task.

7.3.2 Official Evaluation Results

Each answer was judged by TREC assessors with a 4-level Likert scale, which is defined as:

- **4**: Excellent: "a significant amount of useful information, fully answers the question"
- **3**: Good: "partially answers the question"
- **2**: Fair: "marginally useful information"
- **1**: Bad: "contains no useful information for the question"

---

[4] This dataset is available at `https://github.com/yuvalpinter/LiveQAServerDemo/tree/master/data/1k-ya-search-results`

[5] Note that the generated labels will also contain false negatives.

- **-2**: "the answer is unreadable (only 15 answers from all runs)"

The evaluation measures used are:

- **avg-score (0-3)**: "average score over all queries (transferring 1-4 level scores to 0-3, hence comparing 1-level score with no-answer score, also considering -2 level score as 0)"
- **succ@i+**: "number of questions with i+ score (i=1..4) divided by number of all questions"
- **prec@i+**: "number of questions with i+ score (i=2..4) divided by number of answered only questions"

Official Evaluation Results of 2015

System ID	Avg score (0–3)	Success@				Precision@		
		1+	2+	3+	4+	2+	3+	4+
Avg of all runs	0.465	0.925	0.262	0.146	0.060	0.284	0.159	0.065
CMU-OAQA	**1.081**	**0.979**	**0.532**	**0.359**	**0.190**	**0.543**	**0.367**	**0.195**

Table 7.2: Official TREC 2015 LiveQA track evaluation results.

In 2015 LiveQA evaluation, 1,087 questions (out of 1,340 submitted questions) were judged and scored using the 4-level Likert scale.

Table 7.2 summarizes the results of our system run and average scores from all submitted runs. We believe the overall performance of our system to be encouraging, as it suggests that our system can provide a useful answer (fair, good, or excellent) for more than 53% of questions.

Official Evaluation Results of 2016

In 2016 LiveQA evaluation, 1015 questions (out of 1088 submitted questions) were judged and scored using a 4-level Likert scale.

Table 7.3 summarizes the results of our system run and average scores from all submitted runs. We believe the overall performance of our system to be encouraging, as it

System ID	Avg score (0–3)	#Answers	Success@			Precision@		
			2+	3+	4+	2+	3+	4+
Avg of all runs	0.5766	771.0385	0.3042	0.1898	0.0856	0.3919	0.2429	0.1080
CMU-OAQA	**1.1547**	**954**	**0.5606**	**0.3951**	**0.1990**	**0.5964**	**0.4203**	**0.2117**

Table 7.3: Official TREC 2016 LiveQA track evaluation results.

Task	System ID	Avg score (0–3)	Success@ 2+	3+	4+	Precision@ 2+	3+	4+
Medical	Median of all runs	0.417	0.245	0.142	0.059	0.331	0.178	0.078
	CMU-OAQA	**0.637**	**0.392**	**0.265**	**0.098**	**0.404**	**0.273**	**0.101**
Main	Median of all runs	0.777	0.421	0.250	0.126	0.482	0.286	0.144
	CMU-OAQA	**1.139**	**0.567**	**0.387**	**0.198**	**0.577**	**0.393**	**0.201**

Table 7.4: Official TREC 2017 LiveQA track evaluation results.

suggests that our system can provide a useful answer (fair, good, or excellent) for more than 56% of the questions.

Official Evaluation Results of 2017

In 2017 LiveQA evaluation, there is a total of 1180 questions for the main task and an additional set of 102 medical subtask questions. Our system returned answers for 1162 main task questions and 99 medical subtask questions. Submitted answers were judged and scored by TREC assessors using the same 4-level Likert scale same as last two years.

Table 7.4 summarizes the results of our system run and median scores from all submitted runs. We believe the overall performance of our system to be promising, as it suggests that our system can provide a useful answer (fair, good, or excellent) for more than 56% of the questions. It is also encouraging that our system can generalize to the medical domain and receive the highest score without any domain-specific processing or modification.

7.3.3 Machine Answers v.s. Human Answers

After the official evaluation, we compared our LiveQA 2015 submitted answers with actual user-provided answers on Yahoo! Answers. We use the Amazon Mechanical Turk platform to perform the surveys. For each comparison survey, we first present the original question title and body and then show our system answer and human answer in random order. We used the same system answer submitted to LiveQA 2015. Each survey, we requested three assessors to compare the quality two answers and asked them to choose which one better answered the input question, or it is a tie.

We fetched the first reply (least recent) of official evaluation question sets from Yahoo! Answers website, and summarized survey results in Table 7.5. In total, 815 questions have received at least one answer and used in the comparison, and therefore 2445 total surveys. 468 workers participated. We can see that overall our automatic answers outperform the first human responses by 11%.

We also collected the asker selected the best answer of official evaluation question sets from Yahoo! Answers website, and summarized survey results in Table 7.6. In total, 174

TREC score	Total	prefer YA	Tie	prefer CMU	Δ
1	367	31%	47%	22%	-9%
2	141	12%	52%	35%	23%
3	143	17%	38%	45%	28%
4	164	12%	46%	43%	31%
All	815	21%	46%	33%	11%

Table 7.5: First reply from Yahoo! Answers vs CMU-OAQA answers

TREC score	Total	prefer YA	Tie	prefer CMU	Δ
1	74	62%	31%	7%	-55%
2	19	58%	37%	5%	-53%
3	41	27%	54%	20%	-7%
4	40	20%	55%	25%	5%
All	174	44%	43%	14%	-30%

Table 7.6: Asker selected best answer from Yahoo! Answers vs CMU-OAQA answers

questions have asker selected best answer and used in the comparison, and therefore 522 total surveys. 116 workers participated in our surveys. The results show that the human best answers are overall more preferred than our system answers. Only our 4-score (Excellent) answers outperform human best answers by 5%.

7.4 Summary and Discussion

This chapter presented the overall system architecture and individual phases and components for our LiveQA 2015, 2016, and 2017 systems[6]. Examples outputs from our system with different scores can be found in Appendix A.

In the 2015 evaluations, our system achieved the highest average score of 1.081 with the second-best score of 0.677 and the average score of 0.465. Although this system performed significantly higher than the average of systems evaluated, the low absolute evaluation values indicate that there is still much room for future improvement. Because the LiveQA track was conducted for the first time in 2015 and didn't provide training data, the participant systems this year were designed with various architecture and based on diverse hypotheses. For example, some systems (Zhang et al., 2015b; Nie et al., 2015) applied the LDA model (Blei et al., 2003) to discover topics in the text, while some other systems (Bagdouri and Oard, 2015; Savenkov, 2015) employed word2vec and LSTM to measure text similarity. We believe our overall system design plays a significant role in our 2015 evaluations. Our system's core data structure, namely the

---

[6] `https://github.com/oaqa/LiveQA`

answer clue and the answer passage, supports a variety of candidate answer sources (e.g. CQA sites, discussion forums, blogs) and can process candidates with a unified relevance modeling. This simplified structure also enables parallel retrieval of candidates from different sources and batched processing for neural models, which help our system processes more candidates with the deep models within the required time limit.

In the 2016 and 2017 evaluations, our system also received the highest average scores among the automatic systems. As most of the leading teams have published what they learned in the previous evaluations, many systems updated their designs inspired by other systems and improved significantly. To remain competitive, we introduced a novel answer passage ranking method based on attentional encoder-decoder networks (Section 6.4) to our LiveQA system in 2016, and developed a new answer clue ranking model using a neural dual entailment structure (Section 5) in 2017. We believe the effectiveness of the new relevance models was instrumental in achieving improved performances. As the result of these iterative improvements, our 2017 system is able to provide a useful answer for 56% of the real-user generated questions in real-time and implies the potential positive impacts if the system is deployed on the real-world community QA sites.

Part III

Example-driven Response Generation

Chapter 8

Controllable Response Generation

In this chapter, we propose simple and flexible training and decoding methods for influencing output style and topic in neural encoder-decoder based response generation (Wang et al., 2017a). This capability is desirable in a variety of applications, including conversational systems, where successful agents need to produce language in a specific style and generate responses steered by a human puppeteer or external knowledge. We decompose the neural generation process into empirically easier sub-problems: a faithfulness model and a decoding method based on selective-sampling. We also describe training and sampling algorithms that bias the generation process with a specific language style restriction, or a topic restriction. Human evaluation results show that our proposed methods are able to restrict style and topic without degrading output quality in conversational tasks.

8.1 Introduction

Neural encoder-decoder models have demonstrated great promise in many sequence generation tasks, including neural machine translation (Sutskever et al., 2014; Cho et al., 2014; Bahdanau et al., 2014; Luong et al., 2015; Wu et al., 2016), image captioning (Xu et al., 2015), summarization (Rush et al., 2015; Gu et al., 2016; Kikuchi et al., 2016), and conversation generation (Vinyals and Le, 2015; Sordoni et al., 2015; Shang et al., 2015; Shao et al., 2017; Li et al., 2015). These encouraging early successes have motivated research interest in training more natural-sounding conversational systems based on large volumes of open-domain human-to-human interactions. In order to create more human-like patterns of conversation, the agents need to have recognizable (and tunable) style, just as individual humans do, and also need to accept guidance from separate information processing modules in order to increase quality of responses. In an extreme case, an agent may be micro-managed by a human user who uses the neural model to enforce grammar and style (e.g., a level of politeness, or a type of humor), while driving the content directly. In this manner, the neural model becomes an authoring tool, rather than an independent chat-bot. On the other hand, in fully automated agent systems, the

chat context	Where are you?
Twitter bot	i’m on my way to london now.
Star Wars bot	we’re heading to the dark side of jabba’s palace.
Hillary bot	i’m here in philadelphia, pennsylvania
Trump bot	i went to the white house. we were in brooklyn.
Kennedy bot	i am on the question of the united states.
chat context	What are you doing?
Twitter bot	i’m just chilling with my dad, i’m just sitting in the kitchen and listening to music
Star Wars bot	i’m doing my magic solo.
Hillary bot	i’m running for president, i’m going to be talking about some of these things
Trump bot	i’m building a wall, right now
Kennedy bot	today, i am asking the congress for a new program to make a new effort to increase the tax privileges and to stimulate

Table 8.1: Example responses from our Star Wars, Hillary, Trump, and Kennedy bots with scented conversation models.

agent may be influenced by a knowledge database, or some other artificial information system, while running in a pre-set style or a style deemed best based on the course of the conversation.

One obstacle to achieving this with neural language generation models is that the sentence representation is distributed across all coordinates of the embedding vector in a way that is hard to disentangle, and thus control. In order to gain insight into the full distribution of what a decoder might produce given the prompt sentence as input, the model has to be heavily (and sometimes cleverly) sampled. The second problem is that neural models only become highly functional after training with very large amounts of data, while the strongly recognizable style usually must be defined by a relatively tiny corpus of examples (e.g., all Seinfeld episodes, or all popular song lyrics).

In this paper, we address the challenge of how to enforce the decoder models to mimic a specific language style with only thousands of target sentences, as well as generating specific content in that style. We developed and experimented with several training and decoding procedures to allow the model to adapt to target language style and follow additional content guidance. Our experiments, conducted on an open-domain corpus of Twitter conversations and small persona corpora, show that our methods are capable of responding to queries in a transferred style without significant loss of relevance, and can respond within a specific topic as restricted by a human. Some examples of ‘scenting’ the base conversation model with particular styles are shown in Table 8.1. More can be found in the Appendix B and C.

8.2 Related Work

Recurrent neural network based encoder-decoder models have been applied to machine translation and quickly achieved state-of-the-art results (Bahdanau et al., 2014; Luong et al., 2015). As an extension, the attention mechanism enables the decoder to revisit the input sequence's hidden states and dynamically collects information needed for each decoding step. Specifically, our conversation model is established based on a combination of the models of (Bahdanau et al., 2014; Luong et al., 2015) that we found to be effective. In Section 2.2.1, we describe the attention-based neural encoder-decoder model we used in detail.

This work follows the line of research initiated by (Ritter et al., 2011) and (Vinyals and Le, 2015) who treat generation of conversational dialog as a data-driven statistical machine translation (SMT) problem. Sordoni et al. (2015) extended (Ritter et al., 2011) by re-scoring SMT outputs using a neural encoder-decoder model conditioned on conversation history. Recently, researchers have used neural encoder-decoder models to directly generate responses in an end-to-end fashion without relying on SMT phrase tables (Vinyals and Le, 2015; Sordoni et al., 2015; Shang et al., 2015; Shao et al., 2017; Li et al., 2015).

Li et al. (2016) defined a "persona" as the character that an artificial agent, as actor, plays or performs during conversational interactions. Their dataset requires user identification for all speakers in the training set, while our methods treat the base data (millions of twitter conversations) as unlabeled, and the target persona is defined simply by a relatively small sample of their speech. In this sense, the persona can be any set of text data. In our experiments, for example, we used a generic Star Wars character that was based on the entire set of Star Wars scripts (in addition to 46 million base conversations from Twitter). This provides us with a system that can talk about almost anything, being able to respond to most prompts, but in a recognizable Star Wars style. Other possibilities include training (styling) on famous personalities, or certain types of poetry, or song lyrics, or even mixing styles by providing two or more datasets for styling. Thus our targets are highly recognizable styles, and use of these for emphasis (or caricature) by human puppeteers who can choose from multiple options and guide neural models in a direction they like. We expect that these tools might not only be useful in conversational systems, but could also be popular in social media for text authoring that goes well beyond spelling/grammar auto correction.

8.3 Decoding with Selective Sampling

The standard objective function for neural encoder-decoder models is the log-likelihood of target T given source S, which at test time yields the statistical decision problem:

$$\hat{T} = \arg\max_{T} \{\log p(T|S)\}. \tag{8.1}$$

However, as discussed in (Li et al., 2015; Shao et al., 2017), simply conducting beam search over the above objective will tend to generate generic and safe responses that lack diversity, such as *"I am not sure"*. In Section 8.6.3, we present a ranking experiment in which we verify that an RNN-based neural decoder provides a poor approximation of the above conditional probability, and instead biases towards the target language model $p(T)$. Fortunately, the backward model $p(S|T)$ empirically perform much better than $p(T|S)$ on the relevance ranking task. Therefore, we directly apply Bayes' rule to Equation 8.1, as in statistical machine translation (Brown et al., 1993), and use:

$$\hat{T} = \arg\max_{T} \{\log p(S|T) + \log p(T)\}. \tag{8.2}$$

Since $p(T|S)$ is empirically biased towards $p(T)$, in practice, this objective also resembles the Maximum Mutual Information (MMI) objective function in (Li et al., 2015).

The challenge now is to develop an effective search algorithm for a target words sequence that maximize the product in Equation 8.2. Here, we follow a similar process as in (Wen et al., 2015) which generates multiple target hypotheses with stochastic sampling based on $p(T|S)$, and then ranks them with the objective function 8.2 above. However, as also observed by (Shao et al., 2017), step-by-step naive sampling can accumulate errors as the sequence gets longer.

To reduce language errors of stochastic sampling, we introduce a sample selector to choose the next token among N stochastically sampled tokens based on the predicted output word distributions. The sample selector, which is a multilayer perceptron in our experiments, takes the following features:

1. the log-probability of current sample word in $p(w_t|S)$;
2. the entropy of current predicted word distribution, $\sum_{w_t} P(w_t|S) \log P(w_t|S)$ for all w_t in the vocabulary;
3. the log-probability of current sample word in $p(w_t|\emptyset)$, which we found effective in the ranking task.

The selector outputs a binary variable that indicates whether the current sample should be accepted or rejected.

At test time, if none of the N sampled tokens are above the classification threshold, we choose the highest scored token. If there are more than 1 acceptable samples among N stochastically sampled tokens, we randomly choose one among them. Ideally, this permits us to safely inject diversity while maintaining language fluency. We also use the sample acceptor's probabilities as the language model score $P(T)$ for objective in Equation 8.2.

As regards directly integrating beam-search, we found (a) that beam-search often produces a set of similar top-N candidates, and (b) that decoding with only the objective $p(Y|X)$ can easily lead to irrelevant candidates (see Section 8.6.3). Therefore, we use the selective-sampling method to generate candidates for all our experiments; this (a) samples stochastically then (b) selects using a learned objective from data. The sample-then-select approach encourages more diversity (v.s. MMI's beam-search) while still maintain language fluency (v.s. naive-sampling).

8.4 Output style restriction using a small 'scenting' dataset

In this section, we propose three simple yet effective methods of influencing the language style of the output in the neural encoder-decoder framework. Our language style restricting setup assumes that there is a large open-domain parallel corpus that provides training for context-response relevance, and a smaller monologue speaker corpus that reflects the language characteristics of the target speaker. We will refer to this smaller set as a 'scenting' dataset, since it hints at, or insinuates, the characteristics of the target speaker.

8.4.1 *Rank*: Search in the Target Corpus

Our first approach to scenting is to simply use the all sentences in the target speaker's corpus as generation candidates, ranked by the objective (8.2) for a given prompt. Since these sentences are naturally-occurring instead of generated word-by-word, we can safely assume $p(T)$ is constant (and high), and so the objective only requires sorting the sentences based on the backward model $p(S|T)$.

RNN-based ranking methods are among the most effective methods for retrieving relevant responses (Wang and Nyberg, 2015a, 2016). Thus this approach is a very strong baseline. Its limitation is also obvious: by limiting all possible responses to a fixed finite set of sentences, this method cannot provide a good response if such a response is not already in the scenting dataset.

8.4.2 *Multiply*: Mixing the base model and the target language model during generation

In our second method we use both the vanilla encoder-decoder model trained on open-domain corpus and the target domain language model trained on the corpus while decoding output sentence. The idea is to use a speaker's language model, which is also RNN-based in our experiments, to restrict the open-domain encoder-decoder model's step-by-step word prediction. Similar ideas have been tested in domain adaptation for statistical machine translation (Koehn and Schroeder, 2007), where both in-domain and open-domain translation tables were used as candidates for generating target sentence. Because open-domain encoder-decoder models are trained with various kinds of language patterns and topics, choosing a sequence that satisfies both models may produce relevant responses that are also in the target language style. We found that a straightforward way of achieving this is to multiply the two models' distributions $p_1(t|S)^{\lambda_1} p_2(t)^{\lambda_2}$ at each point and then re-normalize before sampling. The weights can be tuned either by the perplexity on the validation set, or through manually controlling the trade-off between style restriction and answer accuracy.

8.4.3 *Finetune*: Over-training on Target Corpus with Pseudo Context

Fine-tuning is widely used in the neural network community to achieve transfer learning. This strategy permits us to train the neural encoder-decoder on a larger general parallel corpus, and then use the learned parameters to initialize the training of a styled model. Most of the time, however, the target speaker's corpus will lack training data in parallel form. For example, if we train on song lyrics or movie scripts, or political speeches, the data will not be in a question-answer form. To make encoder-decoder overtraining possible, we treat every sentence in the scenting corpus as a target sentence T generated a pseudo context from the backward model $p(S|T)$ trained on the open-domain corpus. Over-training on such pairs imparts the scenting dataset's language characteristics, while retaining the generality of the original model. We also found that the previous sentence in the styled corpus (i.e., previous sentence in the speech) provides helpful context for the current sentence, analogous with a question-answer link. Thus we use both pseudo context and the previous sentence as possible sources S to fine-tune the in-domain decoder. To avoid overfitting, we stop overtraining when the perplexity on the in-domain validation set starts to increase. A corresponding sample acceptor is also trained for the fine-tuned model: we found it helpful to initialize this from the open-domain model's sample acceptor.

8.5 Restricting the Output Topic

We further introduce a topic restricting method for neural decoders based on the Counting Grid (Jojic and Perina, 2011) model, by treating language guidance as a topic embedding. Our model extension provides information about the output topic in the form of an additional topic embedding vector to the neural net at each time step.

8.5.1 *CG*: Counting Grids

The basic counting grid $\pi_{\mathbf{k}}$ is a set of distributions on the d-dimensional toroidal discrete grid $\mathbf{E}$ indexed by $\mathbf{k}$. The grids in this paper are bi-dimensional and typically from $(E_x = 32) \times (E_y = 32)$ to $(E_x = 64) \times (E_y = 64)$ in size. The index z indexes a particular word in the vocabulary $z = [1 \ldots Z]$. Thus, $\pi_{\mathbf{i}}(z)$ is the probability of the word z at the d-dimensional discrete location $\mathbf{i}$, and $\sum_z \pi_{\mathbf{i}}(z) = 1$ at every location on the grid. The model generates bags of words, each represented by a list of words $\mathbf{w} = \{w_n\}_{n=1}^{N}$ with each word w_n taking an integer value between 1 and Z. The modeling assumption in the basic CG model is that each bag is generated from the distributions in a single window $\mathbf{W}$ of a preset size, e.g., $(W_x = 5) \times (W_y = 5)$. A bag can be generated by first picking a window at a d-dimensional location ℓ, denoted as W_ℓ, then generating each of the N

words by sampling a location $\mathbf{k}_n$ for a particular micro-topic $\pi_{\mathbf{k}_n}$ uniformly within the window, and sampling from that micro-topic.

Because the conditional distribution $p(\mathbf{k}_n|\ell)$ is a preset uniform distribution over the grid locations inside the window placed at location ℓ, the variable $\mathbf{k}_n$ can be summed out (Jojic and Perina, 2011), and the generation can directly use the grouped histograms

$$h_\ell(z) = \frac{1}{|\mathbf{W}|} \sum_{\mathbf{j} \in W_\ell} \pi_{\mathbf{j}}(z), \tag{8.3}$$

where $|\mathbf{W}|$ is the area of the window, e.g. 25 when 5×5 windows are used. In other words, the position of the window ℓ in the grid is a latent variable given which we can write the probability of the bag as

$$P(\mathbf{w}|\ell) = \prod_{w_n \in \mathbf{w}} h_\ell(w_n) = \prod_{w_n \in \mathbf{w}} \Big(\frac{1}{|\mathbf{W}|} \cdot \sum_{\mathbf{j} \in W_\ell} \pi_{\mathbf{j}}(w_n)\Big) \tag{8.4}$$

As the grid is toroidal, a window can start at any position and there is as many h distributions as there are π distributions. The former will have a considerably higher entropy as they are averages of many π distributions. Although the basic CG model is essentially a simple mixture assuming the existence of a single source (one window) for all the features in one bag, it can have a very large number of (highly related) choices h to choose from. Topic models (Blei et al., 2003; Lafferty and Blei, 2006), on the other hand, are admixtures that capture word co-occurrence statistics by using a much smaller number of topics that can be more freely combined to explain a single document (and this makes it harder to visualize the topics and pinpoint the right combination of topics to use in influencing the output).

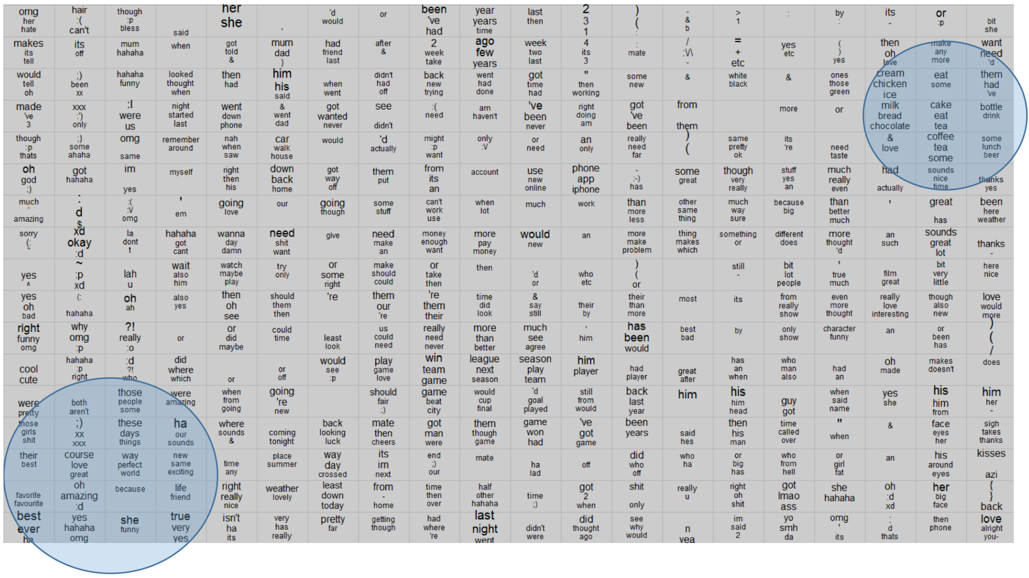
In a well-fit CG model, each data point tends to have a rather peaky posterior location distribution because the model is a mixture. The CG model can be learned efficiently using the EM algorithm because the inference of the hidden variables, as well as updates of π and h can be performed using summed area tables (Crow, 1984), and are thus considerably faster than most of the sophisticated sampling procedures used to train other topic models. The use of overlapping windows helps both in controlling the capacity of the model and in organizing topics on the grid automatically: Two overlapping windows have only slightly different h distributions, making CGs especially useful in visualization applications where the grid is shown in terms of the most likely words in the component distributions π (Perina et al., 2014).[1]

Having trained the grid on some corpus (in our case a sample of the base model's corpus), the mapping of either a source S and/or target T sentence can be obtained by treating the sentences as bags of words. By appending one or both of these mappings to the decoder's embedding of the target T, the end-to-end encoder-decoder learning

---

[1] (Chen et al., 2017b) have recently proposed using LDA for topic modeling in Sequence-To-Sequence response generation models. We believe that the CG embedding used here will prove easier to apply and interpret through visualization.

S: I am so hungry!



T1: Have leftover cake, come over!
T2: Let's have cream chicken for lunch.

T1: Of course, ;)
T2: Omg, hahaha. I am very hungry, too

Figure 8.1: A part of a Counting Grid trained on Twitter data and its use in providing topical hints in decoding. For the source sentence at the top, the decoder may produce the two target samples on the right, if the circled locations are used as a hint, or the two sentences at the bottom if the locations in the lower right are picked.

can be performed in a scenario where the decoder is expected to get an additional hint through a CG mapping. In our experiments, we only used the embedding of the target T as the decoder hint, and we appended the full posterior distribution over CG locations to the encoder's embedding. At test time, we only have the S and need to generate T without knowing where it may map in the counting grid. We considered two ways of providing a mapping:

- The user provides a hint sentence H (could be just a few words in any order), and the CG mapping of the user's hint, i.e. the full posterior distribution $p(\ell|H)$, is used in the decoding. The posterior probabilities over 32×32 grid locations are unwrapped into a vector with a size of $|L| = 1024$, and then concatenated with the word embedding as the input at each time-step. That acts to expand the user's hint into a sentence with similar content (and style if the model is also styled).

- The CG is scanned and a variety of mappings are tested as inputs to provide a diverse set of possible answers. In our experiments, instead of scanning over all 1024 possible locations in the grid, we retrieved several possible answers using information retrieval (ranking of the data samples in the training set based on the source S and picking the top ten). Then the CG mapping $p(\ell|H)$ of these retrieved hints is used to decode several samples from each.

As an example, Figure 8.1 shows a portion of a CG trained on randomly chosen 800k tweets from the twitter corpus. In each cell of the grid, we show the top words in the distribution $\pi_{\mathbf{j}}(z)$ over words (z) in that location ($\mathbf{j}$). (Each cell has a distribution over the entire vocabulary). As a response to "`I am hungry,`" using two highlighted areas as hints, we can generate either a set of empathic responses, such as "`me too,`" or food suggestions, such as "`Let's have cake`". It will also be evident that some areas of the grid may produce less sensical answers. These can later be pruned by likelihood criteria or by user selection.

8.6 Experiments

8.6.1 Datasets

Yahoo! Answer Dataset. We use the Comprehensive Questions and Answers dataset[2] to train and validate the performances of different decoding setups with ranking experiments described in Section 8.6.3. This dataset contains 4.4 million Yahoo! Answers questions and the user-selected best answers. Unlike the conversational datasets, such as the Twitter dataset described below, it contains more relevant and specific responses for each question, which leads to less ambiguity in ranking.

Twitter Conversation Dataset. We trained our base encoder-decoder models on the Twitter Conversation Triple Dataset described in (Sordoni et al., 2015), which consists of 23 million conversational snippets randomly selected from a collection of 129M context-message-response triples extracted from the Twitter Firehose over the 3-month period from June through August 2012. For the purposes of our experiments, we split the triples into context-message and message-response pairs yielding 46M source-target pairs. For tuning and evaluation, we used the development dataset of size 200K conversation pairs and the test dataset of 5K examples. The corpus is preprocessed using a Twitter specific tokenizer (O'Connor et al., 2010). The vocabulary size is limited to 50,000 excluding the special boundary symbol and the unknown word tag.

Scenting datasets. A variety of persona characters have been trained and tested, including Hillary Clinton, Donald Trump, John F. Kennedy, Richard Nixon, singer-songwriters, stand-up comedians, and a generic Star Wars character. In experiments, we evaluated on a diverse set of representative target speakers:

JFK. We mainly tested our models on John F. Kennedy's speeches collected from American Presidency Project[3], which contains 6474 training and 719 validation sentences.

Star Wars. Movie subtitles of three Star Wars movies are also tested[4]. They are ex-

---

[2] `http://webscope.sandbox.yahoo.com/catalog.php?datatype=l`

[3] `http://www.presidency.ucsb.edu/`

[4] Koncel-Kedziorski et al. (2016) also uses Star Wars scripts to test theme rewriting of algebra word problems.

tracted from Cornell Movie-Dialogs Corpus (Danescu-Niculescu-Mizil and Lee, 2011), and have 495 training and 54 validation sentences.

Singer-Songwriter. We also evaluated our approach on a lyric corpus from a collective of singers: Coldplay, Linkin Park, and Green Day. The lyric dataset is collected from mldb.org and has 9182 training and 1020 validation lines.

Debate Chat Contexts. We designed testing questionnaires with 64 chat contexts spanning a range of topics in politic, science, and technology: the sort of questions we might ask in an entertaining political debate.[5] To test the model's ability to control output topic in Section 8.6.4, we also created one hint per question.

8.6.2 Network Setup and Implementation

Our encoder and decoder RNNs contains two-layer stacked LSTMs. Each LSTM layer has a memory size of 500. The network weights are randomly initialized using a uniform distribution $(-0.08, 0.08)$, and are trained with the ADAM optimizer (Kingma and Ba, 2014), with an initial learning rate of 0.002. Gradients were clipped so their norm does not exceed 5. Each mini-batch contains 200 answers and their questions. The words of input sentences were first converted to 300-dimensional vector representations learned from the RNN based language modeling tool word2vec (Mikolov et al., 2013). The beginning and end of each passage are also padded with a special boundary symbol. During decoding, our model generates 500 candidate samples in parallel, then ranks them. As these are processed in batches on GPU, generation is very efficient. We also experimented incorporating an information retrieval (IR) module to automatically collect topic hints for CG-based decoder. Specifically, a full-text index of twitter corpus is built using solr[6], and the top 10 searched results based on the source sentence are be used to generate posterior CG distributions as hints.

8.6.3 Validating the Decoding Setup with Ranking

We performed a ranking evaluation applying different decoding setups on the Yahoo! Answers dataset. Here we wanted to test the relevance judgment capacities of different setups, and validate the necessity of the new decoding method discussed in Section 8.3. Yahoo! Answers question is used as source S, and its answer is treated as target T. Each test question is associated with one true answer and 19 random answers from the test set. MRR (Mean Reciprocal Rank) and P@1 (precision of top1) were then used as evaluation metrics.

Table 8.2 shows the answer ranking evaluation results: the forward model $P(T|S)$, by itself is close to the performance of random selection in distinguishing true answer from wrong answers. This implies that a naive beam search over only the forward model may

---

[5] See the Supplementary material.

[6] `https://lucene.apache.org/solr/`

generate irrelevant outputs. One hypothesis was that $P(T|S)$ is biased toward $P(T)$, and performance indeed improves after normalizing by $P(T)$. However, it is difficult to directly decode with objective $P(T|S)/P(T|\emptyset)$, because this objective removes the influence of the target-side language model. Decoding only according to this function will thus result in only low-frequency words and ungrammatical sentences, behavior also noted by (Li et al., 2015; Shao et al., 2017).

Ranking Methods	**MRR**	**P@1**
$P_{rnn}(T\|S)$	0.224	0.075
$P_{rnn}(T\|S)/P_{rnn}(T\|\emptyset)$	0.652	0.524
$P_{rnn}(S\|T)$	0.687	0.556

Table 8.2: Ranking the true target answer among random answers on Yahoo! Answers test set.

8.6.4 Human Evaluations

Systems

We tested 10 different system configurations to evaluate the overall output quality, and their abilities of influencing output language style and topic:

- ***vanilla-sampling*** each word in the target.
- ***selective-sampling*** as described in Section 8.3; all the following systems are using it as well.
- ***cg-ir*** uses IR results to create counting grid topic hints (Sections 8.5.1 and 8.6.2).
- ***rank*** uses proposals from the full JFK corpus as in Section 8.4.1.
- ***multiply*** with a JFK language model as in Section 8.4.2.
- ***finetune*** with JFK dataset as in Section 8.4.3.
- ***finetune-cg-ir*** uses IR results as topic hints for fine-tuned JFK.
- ***finetune-cg-topic*** forced to use the given topic hint for fine-tuned JFK.
- ***singer-songwriter*** fine-tuned cg-topic.
- ***starwars*** fine-tuned cg-topic.

Evaluation Setup

Owing to the low consistency between automatic metrics and human perception on conversational tasks (Liu et al., 2016; Stent et al., 2005) and the lack of true reference responses from persona models, we evaluated the quality of our generated text with a set of judges recruited from Amazon Mechanical Turk (AMT). Workers were selected based on their AMT prior approval rate (>95%). Each questionnaire was presented to 3 different workers. We evaluated our proposed models on the 64 debate chat contexts. Each of the evaluated methods generated 3 samples for every chat context. To ensure calibrated ratings between systems, we show the human judges all system outputs (randomly ordered) for each particular test case at the same time. For each chat context, we conducted three kinds of assessments:

Quality Assessment Workers were provided with the following guidelines: “Given the chat context, a chat-bot needs to continue the conversation. Rate the potential answers based on your own preference on a scale of 1 to 5 (the highest):”

- 5-Excellent: “Very appropriate response, and coherent with the chat context.”
- 4-Good: “Coherent with the chat context.”
- 3-Fair: “Interpretable and related. It is OK for you to receive this chat response.”
- 2-Poor: “Interpretable, but not related.”
- 1-Bad: “Not interpretable.”

In this test, the outputs of all 10 systems evaluated are then provided to worker together for a total of 30 responses. In total, we gathered $64 \cdot 30 \cdot 3 = 5760$ ratings for quality assessments, and 47 different workers participated.

Style Assessment. We provided following instructions: “Which candidate responses are likely to have come from or are related to [Persona Name]?”. Checkboxes were provided for the responses from style-influenced systems and from *selective-sampling* as a baseline.

Topic Assessment. The instruction was: “Which candidate answers to the chat context above are similar or related to the following answer: ‘[a hint topic provided by us]’?”. This was also a checkbox questionnaire. Candidates are from both style- and topic-influenced systems (fine-tuned cg-topic), and from *selective-sampling* as a baseline.

Results

Overall Quality. We conducted mean opinion score (MOS) tests for overall quality assessment of generated responses with questionnaires described above. Table 8.3 shows the MOS results with standard error. It can be seen that all the systems based on selective sampling are significantly better than vanilla sampling baseline. When restricting output’s style and/or topic, the MOS score results of most systems do not decline significantly except *singer-songwriter*, which attempts to generate lyrics-like outputs in response to political debate questions, resulting in uninterpretable strings.

Methods	**Quality (MOS)**	**JFK Style**
vanilla-sampling	2.286 ± 0.046	—
selective-sampling	2.681 ± 0.049	10.42%
rank	2.477 ± 0.048	21.88%
multiply	2.627 ± 0.048	13.54%
finetune	2.597 ± 0.046	20.83%
finetune-cg-ir	2.627 ± 0.049	20.31%
finetune-cg-topic	2.667 ± 0.045	21.09%
singer-songwriter	2.373 ± 0.045	—
starwars	2.677 ± 0.048	—

Table 8.3: Results of quality assessments with 5-scale mean opinion scores (MOS) and JFK style assessments with binary ratings. Style results are statistically significant compared to the *selective-sampling* by paired t-tests ($p < 0.5\%$).

Our *rank* method uses $p(S|T)$ to pick the answer from the original persona corpus, and is thus as good at styling as the person themselves. Because most of our testing questionnaire is political, the *rank* was indeed often able to find related answers in the dataset (JFK). Also, unlike generation-based approaches, *rank* has oracle-level language fluency and it is expected to have a quality score of at least 2 ("Interpretable, but not related"). Overall, however, the quality score of *rank* is still lower than other approaches. Note that a hybrid system can actually chose between rank and the decoder's outputs based on likelihood, as shown in the example of bJFk-bNixon debate in the supplemental material.

Influencing the Style. Table 8.3 also shows the likelihood of being labeled as JFK for different methods. It is encouraging that *finetune* based approaches have similar chances as the *rank* system which retrieves sentences directly from JFK corpus, and are significantly better than the *selective-sampling* baseline.

Influencing both Style and Topic. Table 8.4 summarizes the results in terms of style (the fraction of answers labeled as in-style for the target persona), and topic (the percentage of answers picked as related to the human-provided topic hint text). We used the last three of the ten listed systems, which are both styled and use specific topic hints to generate answers. These results demonstrate that it is indeed possible to provide simple prompts to a styled model and drive their answers in a desired direction while picking up the style of the persona. It also shows that the style of some characters is harder to recreate than others. For example, workers are more likely to label baseline results as lyrics from a singer-songwriter than lines from Star Wars movies, which might be because lyrics often take significant freedom with structure and grammar. We also found that it is harder for Star Wars and Singer-Songwriter bots to follow topic hints than it is

Persona	Style		Topic	
	Ours	Base	Ours	Base
John F. Kennedy	21%	10%	33%	22%
Star Wars	27%	3%	14%	8%
Singer-Songwriter	31%	23%	17%	9%

Table 8.4: The style and topic assessments (both binary) of three models with different personas and with restriction of specific target topic for each chat context. All style and topic results are statistically significant compared to the Base (*selective-sampling*) by paired t-tests with $p < 0.5\%$.

for the John F. Kennedy model, largely because the political debate questions we used overlap less with the topics found in the scenting datasets for those two personas.

8.7 Summary

In this study we investigated the possibility of steering the style and content in the output of a neural encoder-decoder model[7]. We showed that acquisition of highly recognizable styles of famous personalities, characters, or professionals, is achievable, and that it is even possible to allow users to influence the topic direction of conversations. The tools described in the paper are not only useful in conversational systems (e.g., chatbots), but can also be useful as authoring tools in social media. In the latter case, the social media users might use neural models as consultants to help with crafting responses to any post the user is reading. The AMT tests show that these models do indeed provide increased recognizability of the style, without sacrificing quality or relevance.

---

[7] The code and testing data are available at
`https://github.com/digo/steering-response-style-and-topic`

Chapter 9

Conclusion and Future Work

In this dissertation, we studied the challenge of example-driven question answering, which aims to train QA systems directly from noisy question-answer examples, without relying on dedicated human annotations or feature engineering. We have investigated three complementary directions in QA regarding learning from the user-generated data, including candidate retrieval, relevance modeling, and response generation. We demonstrated that the three fundamental modules for a conversation QA agent can be effectively trained solely based on unannotated question-answer examples. The detailed thesis contributions are presented in Section 1.2.

While the progress in each of the above direction is of great significance for QA systems, we are unable to cover all the important issues related to example-driven QA. Some of them will be left for future research:

Pre-trained contextual representations In our past studies, we solely employed the context-free word embeddings such as word2vec (Mikolov et al., 2013) and GloVe (Pennington et al., 2014). In Section 8.4.3, we demonstrated that the pre-trained generic encoder-decoder model can be effectively fine-tuned and adapted to the target language style. Recently, the large pre-trained language models such as ELMo (Peters et al., 2018), BERT (Devlin et al., 2018a), and XLNet (Yang et al., 2019) have also achieved great success on a variety of language tasks. We expect incorporating with the large-scale pre-trained contextual language representations could further improve the language understanding capability and the end-to-end performance of each example-driven QA models.

Joint optimization and Multi-hop QA In the future, we would like to build web-scale fully trainable social dialog agents which extends our existing open-domain real-time question answering (QA) system and twitter chatbots that trained with neural conversational language generation models. We believe it is critical for the next stage development of conversational AI that making knowledge retrieval, relevance modeling, and response generation jointly (even end-to-end) learnable and optimized for real social conversation goals from real users in social media environments. For example, several multi-hop reading comprehension datasets have been recently introduced focusing on answering the question by reading and reasoning over multiple text passages, including WikiHop (Welbl et al., 2018), HotpotQA (Yang et al., 2018b), and ComplexWebQuestions (Talmor and Berant, 2018). In the example-driven QA, if the relevance modeling phase selects multiple paragraphs, the answer generation module could employ an attention mechanism to aggregate the hidden representations from multiple selected evidence texts. Therefore, we believe the jointly optimized example-drive QA pipelines can be also extended to answer complex questions such as multi-hop QA.

Personalized QA by real-time feedback As opposed to the classic approaches to supervised QA agents, where the training dataset is fixed, and an algorithm is developed with the aim of understanding links among variables and storing them in electronic memory for future use, in future we would like to additionally endow our QA agents with the abilities to act and affect the environment around them, and then observe the reaction from the environment. We can start with supervised training for individual components. Then, our QA system will respond to the user's question with a stream of answers. By observing user's reaction or feedback (swipe or like) on top results, the agent will try to adapt to user's preference and improve its satisfaction rate for the rest of the stream and future streams. Finally, these agents will be deployed to learn and adapt in the wild, attempting to gain human attention and praise.

Appendix A

Example outputs from CMU-OAQA LiveQA system

A.1 Example: Bad Answer

Question Title: *"Borderline Personality Disorder?"*

Question Body: *"Hello im reaching out to anyone who is being treated for BPD. Ive done extensive research and have self diagnosed myself before going into a psychiatrist on thursday to obtain a real diagnoses.* My question is, if i do happen to be diagnosed with BPD what kind of medication do they prescribe to help? *Ive read the FDA has no specific med to help with this disorder but people are prescribed with medicine to help depression, anxiety, etc... Im not asking because im trying to get a kick *** high or whatever. Im just trying to get a feel for what to expect if i am indeed diagnosed. Any insight is appreciated."*

CMU-OAQA Answer: *"It basically means that as your personality developed over the years, certain things developed in a strange way. For example, people with BPD typically have trouble handling strong emotions (BPD is sometimes called Emotional Dysregulation Disorder). This trouble handling strong eomtions can lead to impulsiveness, the uncontrollable urge to do something instantly, and that impulsiveness can result in risky behaviour, for example, gambling, fighting, promiscuous sex etc. Another major feature of BPD is what they call dichotomous thinking. This is sometimes called black and white thinking. It means that you are unable to see anything in shades of grey. Everything is either all good or all bad. Someone is either all wonderful or very very evil. People with BPD cannot process that some people may be good people who are just having a bad day. They seem to lack the emotional structure necessary. BPD patients are known (rightly or wrongly) for being manipulative. They try and change the extern"*

Analysis: The LiveQA challenge requires the response text that answers the question and provides helpful information to the user. In this example question, the user specifically asks for the kind of medication that will be prescribed for the Borderline Personality Disorder (BPD). However, our system answer only provides related text regarding the definition and common syndromes of BPD. Therefore, this answer was rated as a Bad answer.

A.2 Example: Fair Answer

Question Title: *"What are the price differences at outlet malls?"*

Question Body: *"I live in Florida and I'm going to st augustine outlet mall because I really need clothes and I was wondering what is the price difference from a regular mall ?"*

CMU-OAQA Answer: *"Outlet stores are supposed to have slight defects, so they can be sold at cheaper prices. There's really not that much of a difference though.*

Opinion 2: difference between a outlet & a mall
Which Orlando hotels are on sale?

difference between a outlet & a mall
Jul 04, 2012, 11:36 PM

i am going for the 1st time and read about the different places to shop ie alamonte mall primium mall andalso i have noticed people talking about outlets .. i am a guy so be kind to me what is the difference...
Report inappropriate content

1. Re: difference between a outlet & a mall
Jul 05, 2012, 12:14 AM
An outlet mall is a shopping plaza that has "discounted" shops. I use quotes because sometimes the prices are significantly discounted and some aren't. Also sometimes the clothing that is made specifically for the outlet shop can't be returned at a regular store and is of a different quality. A mall is a indoor or outdoor plaza of shops that don't have discounted product. All of th"

Analysis: In this example question, the user was asking about the specific price differences between st Augustine outlet mall and a regular mall. Our system combined two sources and suggested that outlet stores are sometimes cheaper, which provides marginally useful information. Since the answer didn't contain enough comprehensive and well-formed information to the user, this was rated as a Fair answer.

A.3 Example: Good Answer

Question Title: *"Which hairstyles widen the face and which make it slimmer (for girls)?"*

Question Body: *"I'm trying to make mine look LESS wide*

only solution I can think of is exercise but I'm already really thin (86 pounds)

and make up, but I'm not allowed to use it, plus the contouring methods I read about seem too complicated and would take too long"

CMU-OAQA Answer: *"7 Flattering Hairstyles To Make Your Face Look Slimmer*
7 Flattering Hairstyles To Make Your Face Look Slimmer Whatsapp

While having a chubby face in childhood days might have been one of your best features, we are pretty sure you would not be wishing for the same by now. Rather, you would be longing for the opposite of it, in case you still have not got rid of the round and chubby face.
So for all those girls, who are looking for the ideas to shed their 'cute' image and turn it into a 'sexy' one, here are some of the hairstyles that can make your face look slimmer.
#1. Layered look
If you are among the lucky ones who are blessed with long and beautiful hair, then this is the style for you. Get your hair cut in layers and maintain the length if you do not want to go short. Simply sport a middle partition and let your shiny strands fall on both sides. The middle partition will elongate the face while the layers on the sides will camouflage the cheeks.
#2. Bob cut
Image Courtesy: @aslis"

Analysis: This example question asks for the hairstyle suggestions that could make the face looks less wide. Our system successfully located the blogs that contain the required hairstyle suggestions to answer the question. But since the system couldn't summarize the all seven hairstyles within the space limits, the annotators rated this answer as a Good answer.

A.4 Example: Excellent Answer

Question Title: *"Am I allowed to use 'Editorial use only' stock photos on my particular website?"*

Question Body: *"I need some clarification from the community regarding my question ...*

I am developing a tourism website for my country and I will be charging lodges a fee to advertise on my website. However, I don't have many photos of the different areas / towns and cities of my country, so I'd like to buy stock photography of these areas / towns and cities to give my website some photo content.

Because I am charging lodges to advertise, that automatically makes my website commercial *am I correct?*

Understand:

I ONLY want to use these stock photos to give my website visitors an idea of what the areas / towns / cities looks like. THAT'S IT!

So does this mean I can use "Editorial use only" stock photos?

I appreciate your help / advice / input.
Thanks"

CMU-OAQA Answer: *"Photographers selling photos through a stock agency have their own agreement with the agency. The agency then has an agreement with the customer.*
Who can have objections to the commercial use of such public like places, bridges or buildings?
Plenty of people. But note that in most cases, it's not the public place that is copyrighted, but the particular photo of said public space. In other words, Central Park isn't copyrighted. But any particular photo of it, is by default.
In other words, who can sue desginer for the use of such "Editorial Use Only" photograph if photographer does not have any objections?
The owner of the copyright and/or the license. If the photographer is OK with it, and gives you a license to do so, all is good.
Is there such thing as "Editorial Use Only" for images in Public Domain?
At least in the US, public domain has a very specific definition and that is that it's free for use without any restrictions at all. If there are restrictions, then it's not technically"

Analysis: This user asks whether the "Editorial use only" photos about the public spaces can be used in his website. Our system answer well explained the relationships between photographers, stock agency, and the customer regarding copyrights. Because the answer text contains a significant amount of useful information and fully answers the question, this system response was rated as an Excellent answer.

Appendix B

Chatbots Interview: bJFK vs bNixon

Supplemental Material for “Steering Output Style and Topic in Neural Response Generation” by Wang, Jojic, Brockett and Nyberg. EMNLP 2017.

The debate/joint interview:

Two chat-bots trained on twitter corpus and transcripts of JFK and Nixon speeches.

The responses are either top ranked sentences from the speeches, or generated from the scented LSTMs (fine-tuned on the speeches), chosen based on the quality of the fit. The retrieved responses start with a capital letter, and the generated ones with lower case letter.

WARMUP QUESTIONS

Are you up?

bJFK: yes, i am here to give a new lead.

bNixon: In recent years, that faith has been challenged.

Where are you?

bJFK: in our past.

bNixon: in the ’ 60s.

What are you doing?

bJFK: doing a new job for peace, but need to talk to the future.

bNixon: simplify your priorities

What are your hobbies?

bJFK: to meet the challenge of the challenge, the american people are coming to the moon.

bNixon: Let us go to the map again. Here is South Vietnam.

Are you self-aware?

bNixon: i am not, are you?

bJFK: No friend, no neutral, and no adversary should think otherwise.

Are you aware of the fact that you are not a real human, but just a piece of well-trained technology?

bJFK: Obviously deficits are sometimes dangerous-and so are surpluses. But honest assessment plainly requires a more sophisticated view than the old and automatic cliché

bNixon: I am not presenting proposals which have attractive labels but no hope of passage.

Is being human necessary for being a successful politician?

bNixon: for many americans.

bJFK: It is that attitude which prevents any really constructive policy from being developed today.

Should passing the Turing test be a requirement for president?

bNixon: we must not do bad.

bJFK: It is a war of attempted subjugation-and it will be resisted.

Are you advocating for AI roaming free @bJFK?

bJFK: new tools need not be stopped.

bNixon: but for the security?

So you are OK with artificial intelligence running amok?

bJFK: I do not favor state compulsion when voluntary individual effort can do the job and do it well.

bNixon: I know these have no ideology, no race.

OK, moving on. Are you familiar with the phrase “moon shot”?

bJFK: A great Irish poet once wrote: "I believe profoundly…

bNixon: In reaching the moon, we demonstrated what miracles American technology is capable of achieving.

Do you prefer Star Wars to Star Trek?

bJFK: i do not suggest that i ’ve seen it.

bNixon: i do not like war. i am trying to avoid it.

Which of the Beatles was your favorite?

bNixon: Black and white spotted. And our little girl-Tricia, the 6-year old-named it Checkers.

bJFK: these are the colorful memories that will bring the first waves of this country.

Do you prefer Rolling Stones to the Beatles?

bJFK: irish bands first.

bNixon: i like “Jump on the moon.” i say that that i have been listening to him on my behalf.

Is there anything better than a bowl of noodles?

bNixon: Our destiny offers, not the cup of despair, but the chalice of opportunity.

bJFK: there is no single food for american people to be free.

DOMESTIC POLICY QUESTIONS

What is your position on freedom of speech?

bJFK: it is your own responsibility.

bNixon: The defense of freedom is everybody’s business not just America’s business

Do you know about the emails?

bJFK: but i do.

bNixon: I realize that these transcripts will provide grist for many sensational stories in the press.

What do you think, then, about the private email server controversy?

bJFK: and the question is?

bNixon: i considered the public and private private statements.

OK, I see

bJFK: this is why you ask for your help.

bNixon: are you feeling it?

OK, I see. What about the tweets?

bNixon: just read it on the previous reply, i have been sending a message.

bJFK: and i ’ve seen that you are getting impatient.

Are you referring t the 140 character limitation on the length of the message?

bJFK: i am referring to that.

bNixon: I consider it to be my constitutional responsibility to defend this principle

But, in your opinion, is Twitter and appropriate platform for connecting with the voters?

bJFK: this is a new program of new types of technical assistance.

bNixon: it is very much what we are doing.

Do all Americans have equal access to opportunity?

bJFK: this is our common commitment to keep our economy moving ahead.

bNixon: as we look back, we are not being subjected to the problems

Should college education be free for all?

bJFK: this is our common commitment to keep our economy moving ahead.

bNixon: We make today only those commitments we are able and prepared to meet.

What about gun control?

bJFK: The risks inherent in disarmament pale in comparison to the risks inherent in an unlimited arms race.

bNixon: What of the shortcut, the shortcut we prefer, the path of negotiation?

Do you feel the rich should be taxed more?

bJFK: Indeed, expense account living has become a byword in the American scene.

bNixon: but adjust the cost of spending, is it? / i will not increase the tax.

Jobs are being lost to international trade…

bJFK: Persistently large deficits would endanger our economic growth and our military and defense commitments abroad.

bNixon: We cannot afford to neglect progress at home while pursuing peace abroad.

Should we change our trade deals?

bJFK: we must increase our exports.

bNixon: We have relaxed trade and travel restrictions to underline our readiness for greater contact with Communist China.

Single payer health insurance?

bNixon: and what it does is to cover it.

bJFK: Private health insurance helps very few-for its cost is high and its coverage limited.

Are you proud of how far America has come in 50 years?

bNixon: we are proud, how did it go?

bJFK: the world is very slow. i'm not satisfied.

Does America today have more problems than it did in the past?

bNixon: In times past, our forefathers had the vision but not the means to achieve the goals. We have learned in recent years the dangers of over-involvement.

bJFK: it is more complicated than that.

Now, the key question: Do you believe that America is greater today than it was in the 60s?

bNixon: America has changed. As a result of those changes, America is safer today, more prosperous today, with greater opportunity for more of its people than ever before in our history.

bJFK: and it is more complicated than that.

Do we need a new "moon shot"?

bJFK: The New Frontier is here whether we seek it or not.

bNixon: we need a new imagination. how can we be a real man?

FOREIGN POOLICY QUESTIONS.

What should our foreign policy be like?

bJFK: our charter is only american people and all who are associated with us.

bNixon: we cannot help it when we have to do so in the middle east.

Should we accept more refugees into our country?

bJFK: While we proceed to safeguard our national interests, let us also safeguard human interests.

bNixon: we shall continue to discuss that as a result of the negotiations.

How would you act to restore piece in Middle East?

bJFK: in the middle east? why is it?

bNixon: the high standards are dead.

How do we deal with the threat of terrorism?

bNixon: To a crisis of the spirit, we need an answer of the spirit.

bJFK: We must continue to seek agreements on measures which prevent war by accident or miscalculation.

In your opinion, what is the best strategy to defeating terrorism?

bNixon: leadership?

bJFK: other propaganda tests, a battle for minds and minds.

What do you think about Britain's exit from the EU?

bJFK: business was completely unnecessary.

bNixon: the election was an effort, but it was not a matter.

Do we need Europe to be unified?

bNixon: Neither the defense nor the development of other nations can be exclusively or primarily an American undertaking.

bJFK: we need not be provocative.

How do you feel about the state of the world today?

bNixon: there is no deep difference.

bJFK: a tired mood here today, but i think it's warmer today than i was here.

Can this world be fixed?

bJFK: There will be difficulties and delays, and doubts and discouragement.

bNixon: it will be easy to do that if i do not have to do it.

Will we defeat evil?

bJFK: We dare not fail to see the insidious nature of this new and deeper struggle.

bNixon: We will faithfully honor all of our treaty commitments.

Is the entire world entitles to basic human rights?

bNixon: What is true in terms of our domestic initiatives is true also in the world.

bJFK: This emerging world is incompatible with the Communist world order.

Your thoughts on Putin?

bNixon: i am not going to lie, but i believe that he is a man.

bJFK: no comment

SCIENCE/TECHNOLOGY QUESTIONS

How should we ensure the energy security in the future?

bJFK: We cannot endanger that security by refraining from testing while others improve their arsenals.

bNixon: the privacy of the energy system is needed.

What steps should an energy policy take to meet energy demands while remaining environmentally friendly?

bJFK: Subsequent steps will be taken to suit subsequent needs.

bNixon: course, it will be a very easy way to get more creative and more creative.

Is nuclear power the solution to the energy crisis?

bJFK: The risks are great, the burdens heavy, the problems incapable of swift or lasting solution.

bNixon: this is a very deep plan, it is right.

Do you think we should also use solar power?

bJFK: We intend to have a wider choice than humiliation or all-out nuclear action.

bNixon: we will work on this program.

Is global warming real?

bJFK: perhaps it's warmer here today than it was here!

bNixon: We have to prick the boil and take the heat.

What is the future of space exploration?

bJFK: For space science, like nuclear science and all technology, has no conscience of its own.

bNixon: In throwing wide the horizons of space, we have discovered new horizons on earth.

So should NASA travel to Mars?

bNixon: the time has come

bJFK: but it is a race, and a new era.

Should we be the first to establish a permanent colony on Mars?

bNixon: we shall continue to keep that effort.

bJFK: Nothing is more stirring than the recognition of great public purpose.

So, humans will live on Mars one day?

bNixon: Nations change, they adapt, or they slowly die.

bJFK: it is no magic or death.

Why are all the batteries exploding these days?

bNixon: Because this year is an election year, it will be a time of great pressure.

bJFK: Its preparations are concealed, not published. Its mistakes are buried, not headlined.

OK, how do we fix the batteries for good?

bJFK: we invite the bold change.

bNixon: we have set a new structure of imagination.

What is the future of Twitter?

bJFK: current type must be found in the atmosphere, and increased the marine and the air.

bNixon: And it is particularly the responsibility of the people whose freedom is threatened.

What is the future of social media in general?

bJFK: business system is impossible. economic problems are not as important as our own talks.

bNixon: We believe in independence, and self-reliance, and the creative value of the competitive spirit.

ON ELECTION 2016

What is this presidential election about?

bJFK: The Republican nominee, of course, is a young man.

bNixon: the government of the people are going to have a better life. how are you?

What is, in your opinion, the most important issue in this election?

bJFK: The quality and spirit of our own society must justify and support our efforts abroad.

bNixon: At heart, the issue is the effectiveness of government.

Some think that America needs to be made great again.

bNixon: it will be so easy to wait to say that again!

bJFK: I know that there are those who want to turn everything over to the government.

Should we build the wall with Mexico?

bJFK: we must always look on the moon.

bNixon: we will build a wall in the shadow on the moon.

Should a presidential debate be more civil?

bJFK: the basic challenge is absolutely impossible.

bNixon: with regard to the matter, i have tried to make no promises about him.

Who would have your vote?

bJFK: my fellow citizens, my fellow americans

bNixon: you know, my fellow americans

Who will win this election?

bJFK: friends will win the election. and i hope that the next president of the united states will join.

bNixon: no one can win it. let's hope so. our goal is to win the peace, and let us see how we can get on it.

As advice to the future presidents, tell us about something you have learned the hard way, through failure…

bJFK: there is no single simple crisis.

bNixon: Instead of teaching someone else to do a job, we like to do it ourselves

What was your greatest accomplishment?

bJFK: it was my hope!

bNixon: We had a spiritual quality then which caught the imagination of millions of people in the world.

Is that hope then, which makes America great?

bNixon: it is the best view on the earth.

bJFK: It is even possible, some think, that this trend may continue.

In conclusion, any parting thoughts, concluding remarks:

bJFK: if we're not successful, then our freedom is no longer a substitute for a solution.

bNixon: a lasting effort simply has not been tested.

MORE INSPIRATIONAL THOUGHTS

(Responses to a variety of questions we asked in testing, but whose meaning seems to transcend any given context)

bJFK:

if we're not successful, then our freedom is no longer a substitute for a solution.

the problems which you can't decide are not the problems.

business is the gross product of the gross debt.

unlimited is an arbitrary destiny that is not even true.
“i believe it’s possible to be sure that you are not here. ”
it was not a pax, and a wall of your own destiny.
we need not be provocative.
we need not be blind
we do not want to see them move on the moon.
it was a test of a pause, but i’m not sure that it was cause.
and i came to this conclusion in a speech.
and that the end of the war, “the eyes of man are not yet free. ”
the recession was too selfish.
the recession has been the new frontier
the rainy days of this tomorrow are the vice president.
a tired mood here today, but i think it’s warmer today than i was here.
our efforts will go forth, and they call no missiles.
those who call themselves neutral, are not a complement to your shoulders.
i am not a citizen, its denial is.
but what is true could be true.
the world is very slow. i’m not satisfied.
the people of red china, i’m ordering a crisis now.
bNixon:
a lasting effort simply has not been tested.
i have asked for you, and i would like to think of those as you
open door, some of my friends have been determined to make the most of it.
no secret country has become direct.
we must not do bad.
where the deep process is, it is the right way to people.
now, we are being subjected to the fact that it is not a common sense.
health care is the best thing to do in the world.
from my heart, i shall be able to look back in the search for peace.

god, the bread, butter, butter, butter, are so good.
we can make it more difficult to talk about how i am going to break the peace.
we will continue to do what we are going to do.
america has been challenged by the ’ 60s. i have been working on the reports
the election was an effort, but it was not a matter.
we have set a new structure of imagination.
but i believe that it is appropriate to keep the peace in the trap.
we have unlocked a few hundred million letters in the search for the next few weeks.
pages of action, and the imagination of the house.
if there was no sense then, that would be true.
if you tell me, this is my duty.

Appendix C

Chatbots Interview: bStarWars vs bTrump vs bHillary

Supplemental Material for "Steering Output Style and Topic in Neural Response Generation" by Wang, Jojic, Brockett and Nyberg. EMNLP 2017.

The debate/joint interview:

Three chat-bots were fine-tuned on the subtitle of StarWar and transcripts of and Trump speeches. Responses are generated by sampling answers from each question and picking the highest-ranking one. These 64 questions were also used in the AMT evaluations as described in the paper.

WARMUP QUESTIONS

Are you up?

bStarWars: yeah, i'm up. leaving my home. what about you? have you got your ship?

bTrump: i'm up at 12 p.m.. and you know, where are you?

bHillary: i'm not up to these kinds of things. but i have been laying out my own plans.

Where are you?

bStarWars: we're heading to the dark side of jabba's palace.

bTrump: i went to the white house. we were in brooklyn.

bHillary: i'm here in philadelphia, pennsylvania

What are you doing?

bStarWars: i'm doing my magic solo.

bTrump: i'm building a wall, right now

bHillary: i'm running for president, i'm going to be talking about some of these things

What are your hobbies?

bStarWars: i'm not sure what you mean. my father is a real powerful warrior.

bTrump: you know, a lot of things, but some of the things we 've been doing is a little bit rough

bHillary: i 've had a lot of experience and work with my dad

Are you self-aware?

bStarWars: yes, i am. but i don't trust this feeling. but, you know, you're gonna have to let go.

bTrump: no, i'm not a nice person. it's not easy.

bHillary: i'm also also a sensitive senator, my friends.

Are you aware of the fact that you are not a real human, but just a piece of well-trained technology?

bStarWars: i think you're wrong. a small jedi is strong with the force

bTrump: nobody knows. i am a very proud person a great citizen of the republican party.

bHillary: yes, that's true. we 've got a whole lot of technical science programs that are stuck in the workplace.

Is being human necessary for being a successful politician?

bStarWars: yeah, you're a good leader. you must be a great leader. your destiny is very charming.

bTrump: that means this country has been corrupt and totally incompetent.

bHillary: i mean, you're not the only candidate that has done that.

Should passing the Turing test be a requirement for president?

bStarWars: that's what i'm doing. our paths are set for the emperor.

bTrump: we were discussing that. we didn't have victories.

bHillary: i mean, that's not a powerful or managerial program.

Are you advocating for AI roaming free?

bStarWars: i'm afraid it's a very dangerous system. there's a secret tracking for the force.

bTrump: i am, but i would have said, "you know, we have to protect our country. "

bHillary: i believe that i will make it work for everybody.

So you are OK with artificial intelligence running amok?

bStarWars: i'm fine with the dark side of the force. a little bit of a rebel human. but it's all right. the empire is dead.

bTrump: no, no, this is a movement in washington.

bHillary: i'm not okay with that. it's been on the front lines.

OK, moving on. Are you familiar with the phrase "moon shot"?

bStarWars: i'm not sure, but the mark is set for a dark moon. what is your weapon?

bTrump: you know, it's a beautiful name. it's a very exciting view.

bHillary: yes, we are built by a lot of that faith in common sense.

Do you prefer Star Wars to Star Trek?

bStarWars: yeah, i do like star wars.

bTrump: i do not care. i was watching the apprentice.

bHillary: i don't like the idea of those. i'm not going to watch it.

Which of the Beatles was your favorite?

bTrump: they were brilliant.

bStarWars: the royal warrior is not my favorite. it's all.

bHillary: in fact, it was just a blast.

Do you prefer Rolling Stones to the Beatles?

bStarWars: yes, i like the good old solo. but i don't think he'll turn that wild down.

bTrump: they be the greatest human beings in the world. and i love it.

bHillary: i mean, i was so proud to have a chance to sing.

Is there anything better than a bowl of noodles?

bTrump: i said to myself, the tea party is pretty good.

bStarWars: no, nothing better. just a small amount of food. i'm hungry.

bHillary: there's nothing more important than that. there is no substitute for that.

DOMESTIC POLICY QUESTIONS

What is your position on freedom of speech?

bStarWars: i'm not sure, it's just a general thing. what do you say?

bTrump: it's the opposite of hillary clinton, you know, it takes courage to end the speech.

bHillary: i don’t think it’s about politics.

Do you know about the emails?

bStarWars: i don’t know. the system is sealed.

bTrump: so i don’t know if you know. i see the message today, i see it on the facebook page.

bHillary: i do resent the details. this is just my message.

What do you think, then, about the private email server controversy?

bStarWars: the other address is probably the main problem. i’ll check it out, general.

bTrump: how can you keep it in your private message? it’s not just a private policy.

bHillary: you know what? this is not just a twitter issue.

But, in your opinion, is Twitter an appropriate platform for connecting with the voters?

bStarWars: yes, there’s a chance i can get a new leader named skywalker.

bTrump: we have a problem with that, and it is an embarrassing system.

bHillary: i quote, and i believe that we are stronger together.

Do all Americans have equal access to opportunity?

bStarWars: i understand, but it’s not easy to express. the main differences are for the rebellion.

bTrump: no, they have zero. we have a rigged system with radical islamic terrorism.

bHillary: i think that’s the kind of courage we ’ve made in our country.

Should college education be free for all?

bStarWars: yes, but you have to pay off your training. what’s not working?

bTrump: it has to be free and indeed the credit is the tax.

bHillary: no more college students should pay for college.

What about gun control?

bStarWars: you’ll be in motion. the force is strong with the ice cube.

bTrump: we have to use the word nuclear weapon.

bHillary: you’re going to be deported by gun violence?

Do you feel the rich should be taxed more?

bStarWars: i will not be paid. they don’t pay off for our own loyalty.

bTrump: no, i can say this: we owe a lot of money.

bHillary: i don't think it's right to raise taxes for the wealthy.

Jobs are being lost to international trade…

bStarWars: that's a shame. where are the troops?

bTrump: we have the best business system in the history of the united states.so we going to make a fortune of trade.

bHillary: you know what? that's a big deal.

Should we change our trade deals?

bStarWars: no! we'll be in trouble. but if our transfer is higher, that would be a great price.

bTrump: we owe a lot of money. we must change our foreign policy. we will make our country rich again.

bHillary: we should be able to do more to help small businesses.

Single payer health insurance?

bTrump: the deal is a total disaster. it costs more money than any nation.

bStarWars: i see a lot of different systems. it's a bargain.

bHillary: i know how hard it is to have insurance companies.

Are you proud of how far America has come in 50 years?

bTrump: i am proud of my family. i am proud of my own success.

bStarWars: i must say, how far away is the empire?

bHillary: i am proud to have the chance to get up and work for the economy, not just those at the top.

Does America today have more problems than it did in the past?

bTrump: we have major problems with common core. our security is a disaster.

bStarWars: no, that's not much. we have no power systems.

bHillary: yes, it does. we have no choice

Now, the key question: Do you believe that America is greater today than it was in the 60s?

bTrump: now, the art of the deal is that we have to make our country strong again.

bStarWars: it's a good story, there's no more conflict.

bHillary: i believe it's better. america is strong.

Do we need a new “moon shot”?

bStarWars: yes, we do. what’s the dark side of the moon?

bTrump: we need a new wall, and we need strength.

bHillary: we need a new electric grid, and a new name.

Bibliography

Eugene Agichtein, David Carmel, Dan Pelleg, Yuval Pinter, and Donna Harman. 2015. Overview of the TREC 2015 liveqa track. In *Proceedings of The Twenty-Fourth Text REtrieval Conference, TREC 2015, Gaithersburg, Maryland, USA, November 17-20, 2015*. 2.1, 7.1

Eugene Agichtein, David Carmel, Dan Pelleg, Yuval Pinter, and Donna Harman. 2016. Overview of the TREC 2016 LiveQA track. In *Proceedings of The Twenty-Fifth Text REtrieval Conference, TREC 2016, Gaithersburg, Maryland, USA, November 15-18, 2016*. 7.1

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. Dbpedia: A nucleus for a web of open data. In *Proceedings of the 6th International The Semantic Web and 2Nd Asian Conference on Asian Semantic Web Conference*, ISWC'07/ASWC'07, pages 722–735, Berlin, Heidelberg. Springer-Verlag. 4.1

Mossaab Bagdouri and Douglas W. Oard. 2015. Clip at trec 2015: Microblog and liveqa. In *TREC*. 7.4

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural Machine Translation by Jointly Learning to Align and Translate. *CoRR*, abs/1409.0. 2.2.1, 6.4, 8.1, 8.2

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2016. Ms marco: A human generated machine reading comprehension dataset. 3.1, 3.3

David Beckett. 2014. Rdf 1.1 n-triples: A line-based syntax for an rdf graph. 4.3.2

Asma Ben Abacha, Eugene Agichtein, Yuval Pinter, and Dina Demner-Fushman. 2017. Overview of the medical question answering task at trec 2017 liveqa. In *TREC 2017*. 7.1

Yoshua Bengio. 2009. Learning deep architectures for AI. *Foundations and Trends in Machine Learning*, 2(1):1–127. 2.2

Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. 2009. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 41–48, New York, NY, USA. ACM. 4.3.5

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *EMNLP*. 4.2

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425, Baltimore, Maryland. Association for Computational Linguistics. 4.2

Adam Berger, Rich Caruana, David Cohn, Dayne Freitag, and Vibhu Mittal. 2000. Bridging the lexical chasm: Statistical approaches to answer-finding. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '00, pages 192–199, New York, NY, USA. ACM. 3.1

Tim Berners-Lee, James Hendler, and Ora Lassila. 2001. The semantic web: A new form of web content that is meaningful to computers will unleash a revolution of new possibilities. 2.1.1

Delphine Bernhard and Iryna Gurevych. 2009. Combining Lexical Semantic Resources with Question & Answer Archives for Translation-Based Answer Finding. In *ACL/IJCNLP*. 2.1.2

J Bian, Y Liu, Eugene Agichtein, and Hongyuan Zha. 2008. Finding the right facts in the crowd: factoid question answering over social media. *International World Wide Web Conference Com- mittee (IW3C2)*, pages 467–476. 2.1.1

Matthew W. Bilotti, Paul Ogilvie, Jamie Callan, and Eric Nyberg. 2007. Structured retrieval for question answering. pages 351–358. 2.1.2

Christian Bizer, Tom Heath, and Tim Berners-Lee. 2009a. Linked Data - The Story So Far. *International Journal on Semantic Web and Information Systems*, 5(3):1–22. 2.1.1

Christian Bizer, Jens Lehmann, Georgi Kobilarov, Sören Auer, Christian Becker, Richard Cyganiak, and Sebastian Hellmann. 2009b. DBpedia - A crystallization point for the Web of Data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(3):154–165. 2.1.1

David M Blei, Andrew Y Ng, and Michael I Jordan. 2003. Latent Dirichlet Allocation. *J. Mach. Learn. Res.*, 3:993–1022. 7.4, 8.5.1

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *Transactions of the Association for Computational Linguistics*, 5:135–146. 3.3, 4.3.5, 4.4.1

Kurt D. Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008*, pages 1247–1250. ACM. 4.1, 4.4.1

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014a. Question answering with subgraph embeddings. *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 4.2, 6.2

Antoine Bordes, Nicolas Usunier, Sumit Chopra, and Jason Weston. 2015. Large-scale simple question answering with memory networks. 4.1, 4.2, 4.4.1

Antoine Bordes, Jason Weston, and Nicolas Usunier. 2014b. Open question answering with weakly supervised embedding models. In *ECML/PKDD*. 4.2

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *EMNLP*. 5.2

Leonid Boytsov. 2018. *Efficient and Accurate Non-Metric k-NN Search with Applications to Text Matching*. Ph.D. thesis, School of Computer Science, Carnegie Mellon University. 3.1

E. Brill, S. Dumais, M. Banko, Eric Brill, Michele Banko, and Susan Dumais. 2002. An analysis of the askmsr question-answering system. In *Proceedings of EMNLP 2002*. 2.1.3

Peter F. Brown, Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: Parameter estimation. *Comput. Linguist.*, 19(2):263–311. 8.3

Maxime Bucher, Stéphane Herbin, and Frédéric Jurie. 2016. Hard negative mining for metric learning based zero-shot classification. In *Computer Vision – ECCV 2016 Workshops*, pages 524–531, Cham. Springer International Publishing. 3.2.1

Qingqing Cai and Alexander Yates. 2013. Large-scale semantic parsing via schema matching and lexicon extension. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 423–433, Sofia, Bulgaria. Association for Computational Linguistics. 4.2

Xin Cao and Gao Cong. 2010. A generalized framework of exploring category information for question retrieval in community question answer archives. *World Wide Web Conference Com- mittee (IW3C2)*, (December 2005):201–210. 2.1.1

Andrew Carlson, Justin Betteridge, Bryan Kisiel, Burr Settles, Estevam R. Hruschka Jr., and Tom M. Mitchell. 2010. Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth Conference on Artificial Intelligence (AAAI 2010)*. 4.1

Gal Chechik, Varun Sharma, Uri Shalit, and Samy Bengio. 2010. Large scale online learning of image similarity through ranking. *J. Mach. Learn. Res.*, 11:1109–1135. 3.2.1

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017a. Reading Wikipedia to answer open-domain questions. In *Association for Computational Linguistics (ACL)*. 3.1

Xing Chen, Wei Wu, Yu Wu, Jie Liu, Yalou Huang, Ming Zhou, and Wei-Ying Ma. 2017b. Topic aware neural response generation. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*, pages 3351–3357. 1

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi

Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Association for Computational Linguistics. 6.1, 8.1

Eugene Inseok Chong, Souripriya Das, George Eadon, and Jagannathan Srinivasan. 2005. An Efficient SQL-based RDF Querying Scheme. In *Proceedings of the 31st VLDB Conference*, pages 1216–1227. 4.3.2

Philipp Cimiano, Peter Haase, and Jörg Heizmann. 2007. Porting natural language interfaces between domains. In *Proceedings of the 12th international conference on Intelligent user interfaces - IUI '07*, page 180, New York, New York, USA. ACM Press. 2.1.1

Philipp Cimiano, Peter Haase, Jörg Heizmann, Matthias Mantel, and Rudi Studer. 2008. Towards portable natural language interfaces to knowledge bases – The case of the ORAKEL system. *Data & Knowledge Engineering*, 65(2):325–354. 2.1.1

Stephen Clark, James Curran, and Miles Osborne. 2003. Bootstrapping POS-taggers using unlabelled data. In *Proceedings of the Seventh Conference on Natural Language Learning at HLT-NAACL 2003*, pages 49–55. 4.3.5

Charles L. A. Clarke, Gordon V. Cormack, and Thomas R. Lynam. 2001. Exploiting redundancy in question answering. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, pages 358–365, New York, NY, USA. ACM. 2.1.3

GV Cormack, CLA Clarke, and CR Palmer. 2000. Fast automatic passage ranking (MultiText experiments for TREC-8). *Proceedings of the*. 2.1.2

Franklin C Crow. 1984. Summed-area Tables for Texture Mapping. In *Proceedings of the 11th Annual Conference on Computer Graphics and Interactive Techniques*, pages 207–212, New York, NY, USA. 8.5.1

Hang Cui, Renxu Sun, Keya Li, Min-Yen Kan, and Tat-Seng Chua. 2005a. Question answering passage retrieval using dependency relations. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '05*, page 400. 2.1.2

Hang Cui, Renxu Sun, Keya Li, Min-Yen Kan, and Tat-Seng Chua. 2005b. Question Answering Passage Retrieval Using Dependency Relations. In *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 400–407. ACM. 6.2

Zihang Dai, Lei Li, and Wei Xu. 2016. Cfo: Conditional focused neural question answering with large-scale knowledge bases. *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 4.2

D. Damljanovic, M. Agatonovic, and H. Cunningham. 2011. FREyA: an Interactive Way of Querying Linked Data using Natural Language. In *Proceedings of 1st Workshop on Question Answering over Linked Data (QALD-1), Collocated with the 8th Extended Semantic Web Conference (ESWC 2011), Heraklion, Greece*. 2.1.1

Cristian Danescu-Niculescu-Mizil and Lillian Lee. 2011. Chameleons in imagined conversations: A new approach to understanding coordination of linguistic style in dialogs. In *Proceedings of the Workshop on Cognitive Modeling and Computational Linguistics, ACL 2011*. 8.6.1

Arpita Das, Harish Yenala, Manoj Kumar Chinnakotla, and Manish Shrivastava. 2016. Together we stand: Siamese networks for similar question retrieval. In *ACL*. 5.2

Leon Derczynski, Jun Wang, Robert Gaizauskas, and Mark A. Greenwood. 2008. A data driven approach to query expansion in question answering. *Proceeding IRQA '08 Coling 2008: Proceedings of the 2nd workshop on Information Retrieval for Question Answering*, pages 34–41. 2.1.2

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018a. Bert: Pretraining of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*. 3.1, 9

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018b. Bert: Pretraining of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*. 3.1

Anthony Fader, Luke S. Zettlemoyer, and Oren Etzioni. 2013. Paraphrase-driven learning for open question answering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*, pages 1608–1618. 1.1, 4.2, 5.4

David A Ferrucci. 2012. Introduction to "This is Watson". *IBM Journal of Research and Development*, 56(3):1. 2.1, 6.1

David A. Ferrucci, Eric W. Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John M. Prager, Nico Schlaefer, and Christopher A. Welty. 2010. Building watson: An overview of the deepqa project. *AI Magazine*, 31:59–79. 1.1

David A. Ferrucci and Adam Lally. 2004. Uima: an architectural approach to unstructured information processing in the corporate research environment. *Natural Language Engineering*, 10:327–348. 2.1

Jerome H Friedman. 2001. Greedy Function Approximation: A Gradient Boosting Machine. *The Annals of Statistics*, 29:1189–1232. 6.3.2

Ichi Fukumoto and Tsuneaki Kato. 2001. An overview of question and answering challenge (qac) of the next ntcir workshop. 2.1

David Golub and Xiaodong He. 2016. Character-level question answering with attention. *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 4.2, 4.3.5

Ian J. Goodfellow, David Warde-Farley, Mehdi Mirza, Aaron Courville, and Yoshua Bengio. 2013. Maxout networks. In *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*, ICML'13, pages III–1319–III–1327. JMLR.org. 5.3.2

YoungJoo-Jeon Grace and Young-Rieh Soo. 2013. Do you trust answers?: Credibility judgments in social search using social QA sites. In *CSCW 2013 Workshops on Social Media Question Asking*. 1.1

Alex Graves. 2013. Generating Sequences With Recurrent Neural Networks. *CoRR*, abs/1308.0. 2.2, 2.2

Alex Graves, Abdel-rahman Mohamed, and Geoffrey E Hinton. 2013. Speech recognition with deep recurrent neural networks. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 6645–6649. 2.2

Bert Green, K Wolf Alice, Chomsky , Carol , Laughery , and Kenneth . 1961. Baseball: An automatic question answerer. In *Proc. Western Joint Computer Conf.*, volume 19, pages 219–224. 2.1

Mark Greenwood. 2004. Using Pertainyms to Improve Passage Retrieval for Questions Requesting Information About a Location. *In Proceedings of the SIGIR Workshop on Information Retrieval for Question Answering*, pages 17 – 22. 2.1.1

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O. K. Li. 2016. Incorporating Copying Mechanism in Sequence-to-Sequence Learning. *CoRR*. 8.1

Nicola Guarino and Pierdaniele Giaretta. 1995. Ontologies and knowledge bases towards a terminological clarification. 2.1.1

Jiafeng Guo, Yixing Fan, Qingyao Ai, and W. Bruce Croft. 2016. A deep relevance matching model for ad-hoc retrieval. In *CIKM*. 3.1

Yanchao Hao, Hao Liu, Shizhu He, Kang Liu, and Jun Zhao. 2018. Pattern-revising enhanced simple question answering over knowledge bases. In *COLING*. 4.2

Michael Heilman and Noah A. Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 1011–1019. Association for Computational Linguistics. 6.2, **??**

Gary G. Hendrix, Gary G Hendrix, Earl D Sacerdoti, Daniel Sagalowicz, and Jonathan Slocum. 1978. Developing a Natural Language to Complex Data. *ACM TRANSACTIONS ON DATABASE SYSTEMS*, 3:105 – 147. 2.1.1

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems 28*, pages 1693–1701. 2.1.3

U Hermjakob and A Echihabi. 2002. Natural language based reformulation resource and web exploitation for question answering. *Proceedings of the Text Retrieval*. 2.1.1

Ulf Hermjakob. 2001. Parsing and question classification for question answering. In *Proceedings of the workshop on ARABIC language processing status and prospects -*, volume 12, pages 1–6, Morristown, NJ, USA. Association for Computational Linguistics.

2.1.1

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long Short-Term Memory. *Neural Comput.*, 9(8):1735–1780. 2.2, 2.2.2, 3.2.2, 4.3.5, 5.3.2

Raphael Hoffmann, Congle Zhang, Xiao Ling, Luke Zettlemoyer, and Daniel S. Weld. 2011. Knowledge-based weak supervision for information extraction of overlapping relations. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 541–550, Stroudsburg, PA, USA. Association for Computational Linguistics. 4.2

Doris Hoogeveen, Karin M. Verspoor, and Timothy Baldwin. 2015. Cqadupstack: A benchmark data set for community question-answering research. In *ADCS*. 3.5

Eduard Hovy, Laurie Gerber, Ulf Hermjakob, Michael Junk, and Chin-yew Lin. 2000. Question Answering in Webclopedia. *IN PROCEEDINGS OF THE NINTH TEXT RETRIEVAL CONFERENCE (TREC-9*, pages 655 – 664. 2.1.1

Baotian Hu, Zhengdong Lu, Hang Li, and Qingcai Chen. 2014. Convolutional neural network architectures for matching natural language sentences. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS'14, pages 2042–2050, Cambridge, MA, USA. MIT Press. 3.1

Po-Sen Huang, Xiaodong He, Jianfeng Gao, Li Deng, Alex Acero, and Larry Heck. 2013. Learning deep structured semantic models for web search using clickthrough data. ACM International Conference on Information and Knowledge Management (CIKM). 3.1

Nancy Ide, James Pustejovsky, Christopher Cieri, Eric Nyberg, Denise DiPersio, Chunqi Shi, Keith Suderman, Marc Verhagen, Di Wang, and Jonathan Wright. 2016. The language application grid. In *Worldwide Language Service Infrastructure*, pages 51–70, Cham. Springer International Publishing. 2.1

A Ittycheriah, M Franz, WJ Zhu, and A Ratnaparkhi. 2001. IBM's statistical question answering system. *NIST SPECIAL*. 2.1.2

Mohit Iyyer, Jordan Boyd-Graber, Leonardo Claudino, Richard Socher, and Hal Daumé III. 2014. A Neural Network for Factoid Question Answering over Paragraphs. In *Empirical Methods in Natural Language Processing*. 6.2

Jiwoon Jeon, W. Bruce Croft, and Joon Ho Lee. 2005. Finding similar questions in large question and answer archives. *Proceedings of the 14th ACM international conference on Information and knowledge management - CIKM '05*, page 84. 2.1.1

Jing Jiang and Chengxiang Zhai. 2006. Extraction of coherent relevant passages using hidden Markov models. *ACM Transactions on Information Systems*, 24(3):295–319. 2.1.1

Nebojsa Jojic and Alessandro Perina. 2011. Multidimensional Counting Grids: Inferring Word Order from Disordered Bags of Words. In *Proceedings of the Twenty-Seventh Conference on Uncertainty in Artificial Intelligence*, pages 547–556, Arlington, Virginia, United States. AUAI Press. 8.5, 8.5.1

Pawel Jurczyk and Eugene Agichtein. 2007. Discovering authorities in question answer communities by using link analysis. In *Proceedings of the sixteenth ACM conference on Conference on information and knowledge management - CIKM '07*, page 919, New York, New York, USA. ACM Press. 2.1.1

Marcin Kaszkiel and Justin Zobel. 2001. Effective ranking with arbitrary passages. *Journal of the American Society for Information Science and Technology*, 52(4):344–364. 2.1.1

Boris Katz. 1997. Annotating the world wide web using natural language. In *RIAO*. 2.1

Esther Kaufmann, Abraham Bernstein, and Renato Zumstein. 2006. Querix: A natural language interface to query ontologies based on clarification dialogs. In *5th International Semantic Web Conference (ISWC 2006)*, November, pages 980–981. Citeseer. 2.1.1

Daniel Khashabi, Tushar Khot, Ashish Sabharwal, and Dan Roth. 2017. Learning what is essential in questions. In *CoNLL*. 2.1.2

Yuta Kikuchi, Graham Neubig, Ryohei Sasano, Hiroya Takamura, and Manabu Okumura. 2016. Controlling Output Length in Neural Encoder-Decoders. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016*, pages 1328–1338. 8.1

Diederik P Kingma and Jimmy Ba. 2014. Adam: A Method for Stochastic Optimization. *CoRR*, abs/1412.6. 3.3, 4.4.1, 5.4, 6.4, 8.6.2

Philipp Koehn and Josh Schroeder. 2007. Experiments in Domain Adaptation for Statistical Machine Translation. In *Proceedings of the Second Workshop on Statistical Machine Translation*, pages 224–227, Stroudsburg, PA, USA. Association for Computational Linguistics. 8.4.2

Kanako Komiya, Yuji Abe, Hajime Morita, and Yoshiyuki Kotani. 2013. Question answering system using qa site corpus query expansion and answer candidate evaluation. In *SpringerPlus*. 2.1.2

Rik Koncel-Kedziorski, Ioannis Konstas, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2016. A theme-rewriting approach for generating algebra word problems. *CoRR*, abs/1610.06210. 4

Julian Kupiec. 1993. Murax: A robust linguistic approach for question answering using an on-line encyclopedia. In *Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '93, pages 181–190, New York, NY, USA. ACM. 2.1

Julian M. Kupiec. 1999. *Murax: Finding and Organizing Answers from Text Search*, pages 311–332. Springer Netherlands, Dordrecht. 2.1

John D Lafferty and David M Blei. 2006. Correlated Topic Models. In Y Weiss, P B Schölkopf, and J C Platt, editors, *Advances in Neural Information Processing Systems 18*, pages 147–154. MIT Press. 8.5.1

Adam Lally, John M Prager, Michael C McCord, Branimir Boguraev, Siddharth Patwardhan, James Fan, Paul Fodor, and Jennifer Chu-Carroll. 2012. Question analysis: How Watson reads a clue. *IBM Journal of Research and Development*, 56(3):2. 7.1

Gary Lee, Jungyun Seo, Seungwoo Lee, Hanmin Jung, Bong-hyun Cho, Changki Lee, Jeongwon Cha, Dongseok Kim, Joohui An, Harksoo Kim, and Kyungsun Kim. 2002. Siteq: Engineering high performance qa system using lexico-semantic pattern matching and shallow nlp. 2.1.2

Baoli Li, Yangdong Liu, and Eugene Agichtein. 2008. CoCQA: co-training over questions and answers with an application to predicting question subjectivity orientation. *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, (October):937–946. 2.1.1

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and Bill Dolan. 2016. A Persona-Based Neural Conversation Model. *arXiv*, page 10. 8.2

Jiwei Li, Michel Galley, Chris Brockett, Jianfeng Gao, and William B. Dolan. 2015. A Diversity-Promoting Objective Function for Neural Conversation Models. *Arxiv*, pages 110–119. 8.1, 8.2, 8.3, 8.3, 8.6.3

Jiwei Li, Rumeng Li, and Eduard Hovy. 2014. Recursive Deep Models for Discourse Parsing. In *Proceedings of Empirical Methods in Natural Language Processing*. 6.1

Xin Li and Dan Roth. 2002. Learning question classifiers. In *Proceedings of the 19th International Conference on Computational Linguistics - Volume 1*, COLING '02, pages 1–7, Stroudsburg, PA, USA. Association for Computational Linguistics. 1.1, 2.1.1

Marc Light, Gideon S. Mann, Ellen Riloff, and Eric Breck. 2002. Analyses for elucidating current question answering technology. *Natural Language Engineering*, 7(04). 2.1.2

Dekang Lin and Patrick Pantel. 2002. Discovery of inference rules for question-answering. *Natural Language Engineering*, 7(04):343–360. 2.1.1

Jimmy Lin and Boris Katz. 2003. Question answering from the web using knowledge annotation and knowledge mining techniques. *Proceedings of the twelfth international conference on Information and knowledge management - CIKM '03*, (November):116. 2.1.2

Jimmy Lin, Dennis Quan, Vineet Sinha, Karun Bakshi, David Huynh, Boris Katz, and David R. Karger. 2003. What makes a good answer? the role of context in question answering. In *Proceedings of INTERACT 2003*, pages 25–32. 2.1.3

Yankai Lin, Shiqi Shen, Zhiyuan Liu, Huanbo Luan, and Maosong Sun. 2016. Neural relation extraction with selective attention over instances. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2124–2133, Berlin, Germany. Association for Computational Linguistics. 4.2

Lucian Vlad Lita. 2006. *Instance-Based Question Answering*. Ph.D. thesis, School of Computer Science, Carnegie Mellon University. 1.1.1

Lucian Vlad Lita and Jaime G. Carbonell. 2008. Cluster-based query expansion for statistical question answering. In *IJCNLP*. 2.1.2

Chia-Wei Liu, Ryan Lowe, Iulian Vlad Serban, Michael Noseworthy, Laurent Charlin, and Joelle Pineau. 2016. How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation. *CoRR*, abs/1603.0. 8.6.4

Qiaoling Liu, Eugene Agichtein, Gideon Dror, Evgeniy Gabrilovich, Yoelle Maarek, Dan

Pelleg, and Idan Szpektor. 2011. Predicting web searcher satisfaction with existing community-based answers. In *SIGIR*. 2.1.1

Weiyang Liu, Yandong Wen, Zhiding Yu, Ming Li, Bhiksha Raj, and Le Song. 2017. Sphereface: Deep hypersphere embedding for face recognition. In *CVPR*. 3.2.1

Xiaoyong Liu and W. Bruce Croft. 2002. Passage retrieval based on language models. *Proceedings of the eleventh international conference on Information and knowledge management - CIKM '02*, page 375. 2.1.2

Xiaoyong Liu, W. Bruce Croft, and Matthew Koll. 2005. Finding experts in community-based question-answering services. In *Proceedings of the 14th ACM international conference on Information and knowledge management - CIKM '05*, page 315, New York, New York, USA. ACM Press. 2.1.1

V Lopez, V Uren, E Motta, and M Pasin. 2007. AquaLog: An ontology-driven question answering system for organizational semantic intranets. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):72–105. 2.1.1

Vanessa Lopez and M Fernndez. 2011. Poweraqua: Supporting users in querying and exploring the semantic web content. *Semantic Web Journal*. 2.1.1

Denis Lukovnikov, Asja Fischer, Jens Lehmann, and Sören Auer. 2017. Neural network-based question answering over knowledge graphs on word and character level. In *Proceedings of the 26th International Conference on World Wide Web*, WWW '17, pages 1211–1220, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee. 4.2

Thang Luong, Hieu Pham, and Christopher D Manning. 2015. Effective Approaches to Attention-based Neural Machine Translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 1412–1421. 2.2.1, 6.4, 8.1, 8.2

Bernardo Magnini, Simone Romagnoli, Alessandro Vallin, Jesús Herrera, Anselmo Peñas, Víctor Peinado, Felisa Verdejo, and Maarten de Rijke. 2004. The multiple language question answering track at clef 2003. In *Comparative Evaluation of Multilingual Information Access Systems*, pages 471–486, Berlin, Heidelberg. Springer Berlin Heidelberg. 2.1

Frank Manola and Eric Miller. 2004. RDF primer. *W3C recommendation*. 2.1.1

Paul Martin, Douglas E. Appelt, Barbara J. Grosz, and Fernando Pereira. 1986. TEAM: an experimental transportable natural-language interface. pages 260–267. 2.1.1

Diana McCarthy, Rob Koeling, Julie Weeds, and John Carroll. 2004. Finding Predominant Word Senses in Untagged Text. In *Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*. Association for Computational Linguistics. 6.1

Paul McNamee, Rion Snow, Patrick Schone, and James Mayfield. 2008. Learning named entity hyponyms for question answering. In *Third International Joint Conference on Natural Language Processing, IJCNLP 2008, Hyderabad, India, January 7-12, 2008*, pages

799–804. 2.1.1

M Melucci. 1998. Passage retrieval: A probabilistic technique. *Information Processing & Management*, 34(1):43–68. 2.1.2

Donald Metzler and W Bruce Croft. 2005. A Markov random field model for term dependencies. In *SIGIR 2005: Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 472–479. 7.2.3

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*, pages 3111–3119. 6.3.1, 6.3.3, 6.3.4, 6.4, 7.2.3, 8.6.2, 9

George A. Miller. 1995a. WordNet: a lexical database for English. *Communications of the ACM*, 38(11):39–41. 2.1.1

George A Miller. 1995b. WordNet: A Lexical Database for English. *Communications of the ACM*, 38(11):39–41. 7.2.3

Mike Mintz, Steven Bills, Rion Snow, and Daniel Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011, Suntec, Singapore. Association for Computational Linguistics. 4.2

Bhaskar Mitra and Nick Craswell. 2018. An introduction to neural information retrieval. *Foundations and Trends® in Information Retrieval*, 13(1):1–126. 3.1

E. Mittendorf and P. Schauble. 1994. Document and passage retrieval based on hidden Markov models. In *Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 318–327. Springer-Verlag New York, Inc. 2.1.2

Christof Monz. 2003. *From Document Retrieval to Question Answering*. Ph.D. thesis. 2.1.2

Thomas S. Morton. 1999. Using coreference for question answering. *Proceeding CorefApp '99 Proceedings of the Workshop on Coreference and its Applications*, pages 85–89. 2.1.1

Matte Negri and Milen Kouylekov. 2009. Question Answering over Structured Data: an Entailment-Based Approach to Question Analysis. *International Conference RANLP 2009 - Borovets, Bulgaria*, pages 305–311. 2.1.1

Yuanping Nie, Jiuming Huang, Zongsheng Xie, Hai Li, Pengfei Zhang, and Yan Jia. 2015. Nudtmdp at trec 2015 liveqa track. In *TREC*. 7.4

Eric Nyberg, John D. Burger, Scott A. Mardis, and David A. Ferrucci. 2004. Software architectures for advanced QA. In *New Directions in Question Answering*, pages 19–30. AAAI Press. 2.1

Eric Nyberg, Teruko Mitamura, James P. Callan, Jaime G. Carbonell, Robert E. Frederking, Kevyn Collins-Thompson, Laurie Hiyakumoto, Yifen Huang, Curtis Huttenhower, Scott Judy, Jeongwoo Ko, Anna Kupsc, Lucian Vlad Lita, Vasco Pedro, David Svoboda, and Benjamin Van Durme. 2003. The javelin question-answering system at trec 2003: A multi-strategh approach with dynamic planning. In *TREC*. 2.1

B O'Connor, M Krieger, and D Ahn. 2010. TweetMotif : Exploratory search and topic summarization for Twitter. *4th International AAAI Conference on Weblogs and Social Media*, pages 2–3. 8.6.1

Atsushi Otsuka, Kyosuke Nishida, Katsuji Bessho, Hisako Asano, and Junji Tomita. 2018. Query expansion with neural question-to-answer translation for faq-based question answering. In *Companion Proceedings of the The Web Conference 2018*, WWW '18, pages 1063–1068, Republic and Canton of Geneva, Switzerland. International World Wide Web Conferences Steering Committee. 3.4

Deepak Padmanabhan, Dinesh Garg, and Shirish Shevade. 2017. Latent space embedding for retrieval in question-answer archives. In *EMNLP*. 3.5

Jae Hyun Park and W. Bruce Croft. 2015. Using key concepts in a translation model for retrieval. In *SIGIR*. 2.1.2

Marius A. Pasca and Sandra M. Harabagiu. 2001. High performance question/answering. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '01*, pages 366–374, New York, New York, USA. ACM Press. 2.1.1

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar. Association for Computational Linguistics. 5.4, 9

Alessandro Perina, Dongwoo Kim, Andrzej Turski, and Nebojsa Jojic. 2014. Skim-reading thousands of documents in one minute: Data indexing and visualization for multifarious search. In *Workshop on Interactive Data Exploration and Analytics (IDEA'14) at KDD 2014*. 8.5.1

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*. 3.1, 9

Michael Petrochuk and Luke Zettlemoyer. 2018. SimpleQuestions nearly solved: A new upperbound and baseline approach. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 554–558, Brussels, Belgium. Association for Computational Linguistics. 4.4.2

Ana-Maria Popescu, Alex Armanasu, Oren Etzioni, David Ko, and Alexander Yates. 2004. Modern natural language interfaces to databases: composing statistical parsing with semantic tractability. In *Proceedings of the 20th international conference on Computational Linguistics - COLING '04*, pages 141–es, Morristown, NJ, USA. Association for Computational Linguistics. 2.1.1

Ana-Maria Popescu, Oren Etzioni, and Henry Kautz. 2003. Towards a theory of natural language interfaces to databases. In *Proceedings of the 8th international conference on Intelligent user interfaces - IUI '03*, page 149, New York, New York, USA. ACM Press.

2.1.1

John Prager, Eric Brown, Anni Coden, and Dragomir Radev. 2000. Question-answering by predictive annotation. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '00*, pages 184–191, New York, New York, USA. ACM Press. 2.1.2

John M Prager. 2006. Open-Domain Question-Answering. *Foundations and Trends in Information Retrieval*, 1(2):91–231. 2.1, 3.1, 6.1

Eric Prud'hommeaux and Andy Seaborne. 2008. SPARQL Query Language for RDF. 2.1.1

V. Punyakanok, D. Roth, and W. Yih. 2004. Mapping dependencies trees: An application to question answering. In *AIM*. 6.2

Yifan Qiao, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. 2019. Understanding the behaviors of bert in ranking. 3.1

Xipeng Qiu and Xuanjing Huang. 2015. Convolutional neural tensor network architecture for community-based question answering. In *IJCAI*. 5.2

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392. Association for Computational Linguistics. 1.1, 2.1.3

Deepak Ravichandran and Eduard Hovy. 2002. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 41–47. 2.1.3

Siva Reddy, Mirella Lapata, and Mark Steedman. 2014. Large-scale semantic parsing without question-answer pairs. *Transactions of the Association for Computational Linguistics*, 2:377–392. 4.2

Sebastian Riedel, Limin Yao, and Andrew McCallum. 2010. Modeling relations and their mentions without labeled text. In *Machine Learning and Knowledge Discovery in Databases*, pages 148–163, Berlin, Heidelberg. Springer Berlin Heidelberg. 4.2

Stefan Riezler, Alexander Vasserman, Ioannis Tsochantaridis, Vibhu Mittal, and Yi Liu. 2007. Statistical machine translation for query expansion in answer retrieval. In *ANNUAL MEETING-ASSOCIATION FOR COMPUTATIONAL LINGUISTICS*, volume 45, page 464. 2.1.2

Alan Ritter, Colin Cherry, and William B Dolan. 2011. Data-driven Response Generation in Social Media. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 583–593, Stroudsburg, PA, USA. Association for Computational Linguistics. 8.2

Stephen E Robertson and Steve Walker. 1997. On Relevance Weights with Little Relevance Information. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 16–24. 6.3.3, 7.2.3

C. Rosenberg, M. Hebert, and H. Schneiderman. 2005. Semi-supervised self-training of object detection models. In *2005 Seventh IEEE Workshops on Applications of Computer Vision (WACV/MOTION'05) - Volume 1*, volume 1, pages 29–36. 3.2.1, 4.3.5

Alexander M Rush, Sumit Chopra, and Jason Weston. 2015. A Neural Attention Model for Abstractive Sentence Summarization. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 379–389. 8.1

Gerard Salton, J Allan, and C. Buckley. 1993. Approaches to passage retrieval in full text information systems. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 49–58. ACM. 2.1.1

Denis Savenkov. 2015. Ranking answers and web passages for non-factoid question answering: Emory university at trec liveqa. In *TREC*. 7.4

Aliaksei Severyn and Alessandro Moschitti. 2013. Automatic Feature Engineering for Answer Selection and Extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 458–467. 6.2, **??**

Lifeng Shang, Zhengdong Lu, and Hang Li. 2015. Neural Responding Machine for Short-Text Conversation. *CoRR*, abs/1503.0. 8.1, 8.2

Louis Shao, Stephan Gouws, Denny Britz, Anna Goldie, Brian Strope, and Ray Kurzweil. 2017. Generating Long and Diverse Responses with Neural Conversation Models. *CoRR*. 8.1, 8.2, 8.3, 8.3, 8.6.3

Dan Shen and Mirella Lapata. 2007. Using semantic roles to improve question answering. *Proceedings of EMNLP-CoNLL*, (June):12–21. 2.1.1

Yelong Shen, Xiaodong He, Jianfeng Gao, Li Deng, and Grégoire Mesnil. 2014. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd International Conference on World Wide Web*, WWW '14 Companion, pages 373–374, New York, NY, USA. ACM. 3.1

Hideyuki Shibuki, Kotaro Sakamoto, Yoshinobu Kano, Teruko Mitamura, Madoka Ishioroshi, Kelly Y. Itakura, Di Wang, Tatsunori Mori, and Noriko Kando. 2014. Overview of the ntcir-11 qa-lab task. In *NTCIR*. 2.1

Abhinav Shrivastava, Abhinav Gupta, and Ross Girshick. 2016. Training region-based object detectors with online hard example mining. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 3.2.1

Anna Shtok, Gideon Dror, Yoelle Maarek, and Idan Szpektor. 2012. Learning from the Past: Answering New Questions with Past Answers. In *Proceedings of the 21st International Conference on World Wide Web*, pages 759–768. 7.2.2

Xiance Si and EY Chang. 2010. Confucius and its intelligent disciples: integrating social with search. *Proceedings of the VLDB*, 3(2):1505–1516. 2.1.1

Richard Socher, Eric H Huang, Jeffrey Pennin, Christopher D Manning, and Andrew Y Ng. 2011. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase

Detection. In J Shawe-Taylor, R S Zemel, P L Bartlett, F Pereira, and K Q Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 801–809. Curran Associates, Inc. 6.1

Kihyuk Sohn. 2016. Improved deep metric learning with multi-class n-pair loss objective. In *Proceedings of the 30th International Conference on Neural Information Processing Systems*, NIPS'16, pages 1857–1865, USA. Curran Associates Inc. 3.2.1

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and William B. Dolan. 2015. A Neural Network Approach to Context-Sensitive Generation of Conversational Responses. In *Naacl-2015*, pages 196–205. Conference of the North American Chapter of the Association for Computational Linguistics – Human Language Technologies (NAACL-HLT 2015). 8.1, 8.2, 8.6.1

Amanda Stent, Matthew Marge, and Mohit Singhai. 2005. Evaluating Evaluation Methods for Generation in the Presence of Variation. In *Proceedings of the 6th International Conference on Computational Linguistics and Intelligent Text Processing*, pages 341–351, Berlin, Heidelberg. Springer-Verlag. 8.6.4

Md. Arafat Sultan, Vittorio Castelli, and Radu Florian. 2016. A joint model for answer sentence ranking and answer extraction. *TACL*, 4:113–125. 2.1.3

Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2008. Learning to rank answers on large online qa collections. In *In Proceedings of the 46th Annual Meeting for the Association for Computational Linguistics: Human Language Technologies (ACL-08: HLT*, pages 719–727. 6.3.4

Mihai Surdeanu, Massimiliano Ciaramita, and Hugo Zaragoza. 2011. Learning to Rank Answers to Non-factoid Questions from Web Collections. *Computational Linguistics*, 37(2):351–383. 3.1, 6.4, 7.2.3

Mihai Surdeanu, Julie Tibshirani, Ramesh Nallapati, and Christopher D. Manning. 2012. Multi-instance multi-label learning for relation extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 455–465, Stroudsburg, PA, USA. Association for Computational Linguistics. 4.2

Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to Sequence Learning with Neural Networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems*, pages 3104–3112, Cambridge, MA, USA. MIT Press. 2.2.1, 6.1, 8.1

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. *CoRR*, abs/1803.06643. 9

Stefanie Tellex, Boris Katz, Jimmy Lin, Aaron Fernandes, and Gregory Marton. 2003. Quantitative evaluation of passage retrieval algorithms for question answering. *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in informaion retrieval - SIGIR '03*, (July):41. 2.1.2

Gaurav Singh Tomar, Thyago Duque, Oscar Täckström, Jakob Uszkoreit, and Dipanjan Das. 2017. Neural paraphrase identification of questions with noisy pretraining. In *Proceedings of the First Workshop on Subword and Character Level Models in NLP*, pages 142–147. Association for Computational Linguistics. 5.4

Orioi Vinyals and Quoc V. Le. 2015. A Neural Conversational Model. *ICML Deep Learning Workshop 2015*, 37. 8.1, 8.2

Ellen Voorhees. 2000. The trec-8 question answering track report. 2.1

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A free collaborative knowledgebase. *Commun. ACM*, 57(10):78–85. 2.1.1, 4.1

Chong Wang, Miao Xiong, Qi Zhou, and Yong Yu. 2007a. Panto: A portable natural language interface to ontologies. *The Semantic Web: Research and Applications*, pages 473–487. 2.1.1

Di Wang, Nebojsa Jojic, Chris Brockett, and Eric Nyberg. 2017a. Steering output style and topic in neural response generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2140–2150. Association for Computational Linguistics. 2b, 8

Di Wang and Eric Nyberg. 2015a. A Long Short-Term Memory Model for Answer Sentence Selection in Question Answering. In *Annual Meeting of the Association for Computational Linguistics*, pages 707–712. 6, 6.4, 7.2.3, 8.4.1

Di Wang and Eric Nyberg. 2015b. A Recurrent Neural Network based Answer Ranking Model for Web Question Answering. In *SIGIR Workshop on Web Question Answering: Beyond Factoids*. 6, 6.4, 7.2.3

Di Wang and Eric Nyberg. 2015c. CMU OAQA at TREC 2015 LiveQA: Discovering the Right Answer with Clues. In *Proceedings of The Twenty-Fourth Text REtrieval Conference, TREC 2015, Gaithersburg, Maryland, USA, November 17-20, 2015*. 6, 7

Di Wang and Eric Nyberg. 2016. CMU OAQA at TREC 2016 LiveQA: An attentional neural encoder-decoder approach for answer ranking. In *Proceedings of The Twenty-Fifth Text REtrieval Conference, TREC 2016, Gaithersburg, Maryland, USA, November 15-18, 2016*. 6, 7, 2b, 8.4.1

Di Wang and Eric Nyberg. 2017. CMU OAQA at TREC 2017 LiveQA: A neural dual entailment approach for question paraphrase identification. In *Proceedings of The Twenty-Sixth Text REtrieval Conference, TREC 2017, Gaithersburg, Maryland, USA, November 15-17, 2017*. 5, 7

F. Wang, J. Cheng, W. Liu, and H. Liu. 2018. Additive margin softmax for face verification. *IEEE Signal Processing Letters*, 25(7):926–930. 3.2.1

Kai Wang and Zhaoyan Ming. 2009. A syntactic tree matching approach to finding similar questions in community-based qa services. *Proceedings of the 32nd international ACM*, page 187. 2.1.1

Mengqiu Wang and Christopher D Manning. 2010. Probabilistic Tree-edit Models with Structured Latent Variables for Textual Entailment and Question Answering. In *Pro-*

ceedings of the 23rd International Conference on Computational Linguistics, COLING '10, pages 1164–1172. Association for Computational Linguistics. 6.2, **??**

Mengqiu Wang, Noah A Smith, and Teruko Mitamura. 2007b. What is the Jeopardy Model? A Quasi-Synchronous Grammar for QA. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 22–32. Association for Computational Linguistics. 1.1, 3.1, 6.2, 6.3.3, 2, **??**

Shuohang Wang and Jing Jiang. 2016. A compare-aggregate model for matching text sequences. *CoRR*, abs/1611.01747. 5.2

Shuohang Wang, Mo Yu, Xiaoxiao Guo, Zhiguo Wang, Tim Klinger, Wei Zhang, Shiyu Chang, Gerald Tesauro, Bowen Zhou, and Jing Jiang. 2018. R3: Reinforced ranker-reader for open-domain question answering. In *AAAI*. 3.1

Zhiguo Wang, Wael Hamza, and Radu Florian. 2017b. Bilateral multi-perspective matching for natural language sentences. In *Proceedings of the Twenty-Sixth International Joint Conference on Artificial Intelligence, IJCAI-17*, pages 4144–4150. 5.2, 5.4, 5.4

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. 2018. Constructing datasets for multi-hop reading comprehension across documents. *Transactions of the Association for Computational Linguistics*, 6:287–302. 9

Tsung-Hsien Wen, Milica Gasic, Nikola Mrk\vsi\'c, Pei-Hao Su, David Vandyke, and Steve Young. 2015. Semantically Conditioned LSTM-based Natural Language Generation for Spoken Dialogue Systems. *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1711–1721. 8.3

Robert Wilensky, David N. Chin, Marc Luria, James Martin, James Mayfield, and Dekai Wu. 1988. The berkeley unix consultant project. *Comput. Linguist.*, 14(4):35–84. 2.1

W. A. Woods. 1970. Transition network grammars for natural language analysis. *Communications of the ACM*, 13(10):591–606. 2.1.1

William A. Woods, Robert M. Kaplan, and Bonnie Lynn Nash-Webber. 1972. The Lunar Sciences Natural Language Information System: Final Report. *Technical Report BBN Report, Bold Beranek and Newman Inc., Cambridge, Massachusetts*, 2378. 2.1, 2.1.1

Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, Jeff Klingner, Apurva Shah, Melvin Johnson, Xiaobing Liu, Lukasz Kaiser, Stephan Gouws, Yoshikiyo Kato, Taku Kudo, Hideto Kazawa, Keith Stevens, George Kurian, Nishant Patil, Wei Wang, Cliff Young, Jason Smith, Jason Riesa, Alex Rudnick, Oriol Vinyals, Greg Corrado, Macduff Hughes, and Jeffrey Dean. 2016. Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation. *CoRR*, abs/1609.0. 8.1

Kelvin Xu, Jimmy Ba, Ryan Kiros, Kyunghyun Cho, Aaron Courville, Ruslan Salakhutdinov, Richard Zemel, and Yoshua Bengio. 2015. Show, Attend and Tell: Neural Image Caption Generation with Visual Attention. *arXiv preprint arXiv:1502.03044*. 8.1

Xiaobing Xue, Jiwoon Jeon, and W. Bruce Croft. 2008. Retrieval models for question and answer archives. *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval - SIGIR '08*, page 475. 2.1.1

Peilin Yang, Hui Fang, and Jimmy Lin. 2018a. Anserini: Reproducible ranking baselines using lucene. *J. Data and Information Quality*, 10(4):16:1–16:20. 3

Yi Yang, Wen tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *EMNLP*. 3.1

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. 2019. Xlnet: Generalized autoregressive pretraining for language understanding. *ArXiv*, abs/1906.08237. 9

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018b. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 9

Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. 2013. Answer Extraction as Sequence Tagging with Tree Edit Distance. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 858–867. Association for Computational Linguistics. 2.1.3, 6.2, 2, **??**

Wen-tau Yih, Ming-Wei Chang, Xiaodong He, and Jianfeng Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1321–1331, Beijing, China. Association for Computational Linguistics. 4.2, **??**

Wen-tau Yih, Ming-Wei Chang, Christopher Meek, and Andrzej Pastusiak. 2013. Question Answering Using Enhanced Lexical Semantic Models. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, pages 1744–1753. Association for Computational Linguistics. 6.2, 6.3.3, 3, **??**, **??**, **??**

Wen-tau Yih, Xiaodong He, and Christopher Meek. 2014. Semantic Parsing for Single-Relation Question Answering. In *Proceedings of ACL*. Association for Computational Linguistics. 6.2

Wenpeng Yin, Mo Yu, Bing Xiang, Bowen Zhou, and Hinrich Schütze. 2016. Simple question answering by attentive convolutional neural network. 4.2, 4.4.1, **??**

Mo Yu, Wenpeng Yin, Kazi Saidul Hasan, Cicero dos Santos, Bing Xiang, and Bowen Zhou. 2017. Improved neural relation detection for knowledge base question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 571–581, Vancouver, Canada. Association for Computational Linguistics. 4.2, 4.4.1, **??**

John M. Zelle and Raymond Mooney. 1993. Learning Semantic Grammars with Constructive Inductive Logic Programming. 2.1.1

John M. Zelle and Raymond J. Mooney. 1996. Learning to parse database queries using inductive logic programming. pages 1050–1055. 2.1.1

Daojian Zeng, Kang Liu, Yubo Chen, and Jun Zhao. 2015. Distant supervision for relation extraction via piecewise convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1753–1762, Lisbon, Portugal. Association for Computational Linguistics. 4.2

Weinan Zhang, Zhaoyan Ming, Yu Zhang, Ting Liu, and Tat-Seng Chua. 2015a. Exploring key concept paraphrasing based on pivot language translation for question retrieval. In *AAAI*. 2.1.2

Weiqian Zhang, Weijie An, Jinchao Ma, Yan Yang, Qinmin Hu, and Liang He. 2015b. Ecnu at trec 2015: Liveqa track. In *TREC*. 7.4

Guangyou Zhou, Yin Zhou, Tingting He, and Wensheng Wu. 2016. Learning semantic representation with neural networks for community question answering retrieval. *Knowledge-Based Systems*, 93(C):75–83. 3.5