QUESTION ANSWERING USING DEEP NEURAL NETWORKS: SINGLE TURN AND BEYOND

SOUVIK KUNDU

NATIONAL UNIVERSITY OF SINGAPORE

2020

QUESTION ANSWERING USING DEEP NEURAL NETWORKS: SINGLE TURN AND BEYOND

SOUVIK KUNDU
(B.E., Jadavpur University)

A THESIS SUBMITTED FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF COMPUTER SCIENCE
SCHOOL OF COMPUTING
NATIONAL UNIVERSITY OF SINGAPORE

2020

Supervisor:
Professor Ng Hwee Tou

Examiners:
Professor Lee Wee Sun
Associate Professor Kan Min Yen
Professor Giuseppe Riccardi (University of Trento)

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in this thesis.

This thesis has also not been submitted for any degree in any university previously.

Souvik Kundu

May 13, 2020

Acknowledgements

I would like to extend my utmost gratitude to my advisor, Hwee Tou Ng, for his valuable guidance, motivation and criticism throughout the work of this thesis. He has made this research not only feasible, but more importantly, enjoyable with the correct balance of feedback and independence, and with the great conversations about all aspects of this work.

Next, I would like to thank the board of examiners, Professor Lee Wee Sun, Associate Professor Kan Min Yen, and Professor Giuseppe Riccardi for their valuable feedback.

The acknowledgement would not be complete without mentioning my friends and colleagues in the NUS NLP Group. It was a great pleasure working with them, and I appreciate their help and support.

I dedicate this thesis to my family, who offered unconditional love, for their endless support and encouragement.

Contents

Abstract

In this thesis, we tackle the task of question answering (QA). Machine comprehension (MC) systems mimic the process of reading comprehension (RC) by answering questions after understanding a natural language text. We first present a deep neural network model for MC-based QA. We develop an end-to-end question-focused multi-factor attention network for answer extraction. Our proposed multi-factor attentive encoding helps to aggregate relevant evidence by using a tensor-based multi-factor attention mechanism. Due to the proposed question-focused attention pointing mechanism, it also learns to focus on the important question words to encode the aggregated question vector. Our proposed model achieves significant improvements over prior work on three large-scale challenging QA datasets.

The second task that we have tackled is nil-aware answer extraction for machine reading comprehension. For a given question, the associated passage might not contain any valid answer. These questions are defined as nil questions. Most of the recently proposed neural models do not consider nil questions, although it is crucial for a practical QA system to be able to determine whether a text passage contains a valid answer for a question. We focus on developing models that extract an answer for a question if and only if the

associated text passage contains a valid answer. Otherwise, they are expected to return *Nil* as answer. We propose a nil-aware answer extraction framework that is capable of returning *Nil* or a text span from the associated passage as an answer in a single step. In addition to our proposed MC model, we show that our proposed framework can be easily integrated with several other recently proposed QA models developed for reading comprehension and can be trained in an end-to-end fashion. The proposed framework decomposes pieces of evidence into relevant and irrelevant parts and then combines them to infer the existence of any answer. Experiments show that the integration of our proposed framework significantly outperforms several strong baseline systems that use pipeline or threshold-based approaches.

Finally, we focus on developing a system for multi-turn conversational question answering. Recently proposed conversational question answering systems lack the ability to ask a follow-up clarification question when a given question is underspecified. In this work, we focus on developing a conversational question answering system that can predict the answer to a question in a conversation, and has the ability to ask a follow-up clarification question if the question is underspecified. We propose a pipeline approach which consists of an answer prediction model and a follow-up question generation model. The answer prediction model is based on a dual co-attention network while the follow-up question generation model is based on a sequence-to-sequence neural network enhanced with a copying mechanism. Experiments on the ShARC dataset show the effectiveness of the proposed system.

List of Tables

List of Figures

Chapter 1

Introduction

In many science fiction movie scenes, we witness people talking to machines in natural language, just like two human beings participating in a natural conversation. Such a dialog agent powered by artificial intelligence has always been the dream of researchers during the last several decades.

In 2011, the natural language processing (NLP) community witnessed a breakthrough when IBM developed its DeepQA system, called Watson. Watson (Ferrucci et al., 2010) beat the biggest all-time winner Brad Rutter, and the longest championship record-holder Ken Jennings. The driving forces behind Watson were advances in NLP, information retrieval (IR), machine learning, knowledge representation, and reasoning. Despite this success, research on answering complex questions and developing intelligent conversational question answering systems is still in the early stages.

Large-scale question answering (QA) systems such as DeepQA rely on multiple sources to answer questions. Besides Wikipedia, it also relies on

knowledge bases (KBs), dictionaries, news articles, books, etc. As a result, such systems heavily rely on information redundancy among the sources to answer correctly. Having a single knowledge source forces the model to be very precise while searching for an answer, since the evidence needed to answer a question might appear only once. This challenge thus encourages research in the ability of a machine to read, a key motivation for the machine comprehension subfield (Hirschman et al., 1999; Ng et al., 2000) and the creation of a number of machine reading comprehension datasets such as CNN/Daily Mail (Hermann et al., 2015), CBT (Hill et al., 2016), SQuAD (Rajpurkar et al., 2016), NewsQA (Trischler et al., 2017), TriviaQA (Joshi et al., 2017), RACE (Lai et al., 2017), SearchQA (Dunn et al., 2017), etc.

1.1 Scope of the Thesis

Machine reading comprehension (RC) can be of several types. For instance, in multi-choice RC (Lai et al., 2017), a system needs to identify the correct choice given a passage, a question, and a set of candidate answer choices. In cloze-style RC (Hermann et al., 2015), a system needs to find the correct answer from the associated passage given a fill-in-the-blank question. An answer is often a single word in this case. In other cases such as SQuAD, NewsQA, TriviaQA, and SearchQA, a system needs to extract a span of text as the answer from the associated passage given a question. In this case, the questions are not necessarily seeking factual answers and the answers can be of any length. In this thesis, we focus on RC that requires a model to extract a span of text as answer. We also develop models to tackle the scenario when

the associated passage does not contain any valid answer for a given question. We call this task nil-aware answer span extraction. In this case, the system is expected to return *Nil* as the answer.

An example is given as follows:

> *(CNN) – An American woman died aboard a cruise ship that docked at Rio de Janeiro on Tuesday, the same ship on which 86 passengers previously fell ill, according to the state-run Brazilian news agency, Agencia Brasil.*
>
> *The American tourist died aboard the MS Veendam, owned by cruise operator Holland America. Federal Police told Agencia Brasil that forensic doctors were investigating her death.*
>
> *The ship's doctors told police that the woman was elderly and suffered from* ***diabetes and hypertension****, according to the agency.*
>
> *The other passengers came down with diarrhea prior to her death during an earlier part of the trip, the ship's doctors said.*
>
> *The Veendam left New York 36 days ago for a South America tour.*
>
> **Question**: What did the elderly woman suffered from?
> **Answer**: diabetes and hypertension
> **Question**: Who is the ship's doctor?
> **Answer**: *Nil*

For the first question, the system needs to extract the valid answer *diabetes and hypertension*, whereas in the second question, it should return *Nil* as the answer due to the absence of any valid answer.

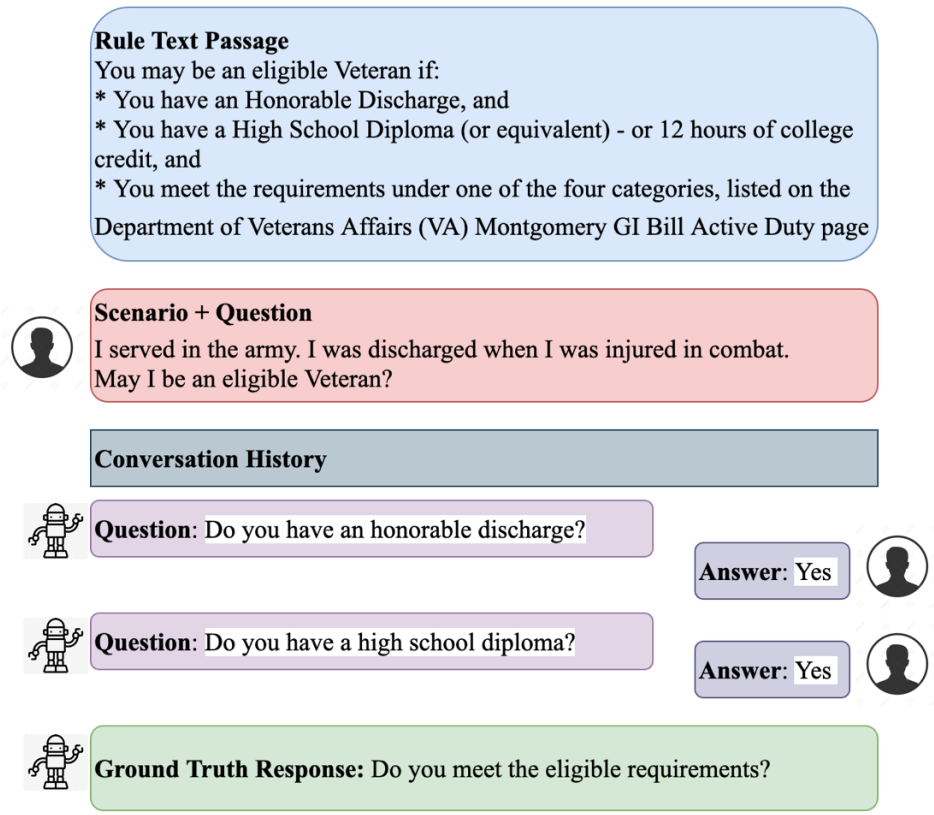


Figure 1.1: A conversational machine reading comprehension-based question answering example. A system is given a rule text as the associated passage for answering the user's question with a specific scenario. During each turn in the conversation history, the system can ask the user a follow-up question to inquire about missing information, and the user answers it with *Yes* or *No*. Finally, the system answers the user's question or ask a clarification question if there is still some missing information.

Finally, we propose a deep neural network-based model for conversational machine reading comprehension-based QA. Humans converse with each other to either seek or test their knowledge about a subject. Depending on the answer, another follow-up question is asked and the second answer builds on what has already been discussed. This incremental aspect makes human conversations succinct. An inability to build and maintain common ground in this way is part of why virtual assistants usually do not seem like competent conversational partners. In conversational machine reading comprehension-based QA, for a given question, a system needs to answer it, or generate a follow-up question if the question is underspecified, by understanding natural language text documents and the previous question-answer pairs in the conversation history. We aim to tackle this task by formulating the problem as a combination of two independent tasks, namely answer classification and follow-up question generation. In this work, we focus on the recently released ShARC dataset (Saeidi et al., 2018). In the ShARC dataset, a system must help users answer underspecified questions by participating in an information-gathering dialog. For example, in Figure 1.1 the system asks a series of questions to help the user decide if he or she is an eligible Veteran. A key challenge in conversational machine reading comprehension-based QA is that the rules by which the decision is made are only provided in natural language text. For every instance, the system must read the rule text passage and the conversation history to determine the answer (*Yes*, *No*, or *Irrelevant*), or generate a follow-up clarification question if it still needs more information to answer the user's question.

1.2 Contributions of the Thesis

There are three major contributions of this thesis, outlined as follows:

- Most of the recently proposed neural network models for QA use recurrent neural networks (RNNs) such as LSTM and GRU for evidence encoding. However, such RNN-based models fail to capture long-range dependencies in practice. They do not possess the capability to aggregate multiple facts distributed across multiple sentences. They are limited to only capturing passage-question interaction. They also do not explicitly focus on identifying the important words in a question which often play a critical role in QA. To overcome these difficulties, we propose a novel end-to-end question-focused multi-factor attention network for machine reading comprehension. Our proposed multi-factor attentive encoding approach is based on tensor transformation to synthesize meaningful evidence across multiple sentences. To implicitly infer the answer type information, we propose a max-attentional question aggregation mechanism that learns to identify the meaningful portions of a question. Our proposed model achieves significant improvements over prior work on three large-scale challenging QA datasets.

- Despite the progress in RC-based QA, current approaches suffer from an impractical assumption that every question has a valid answer in the associated passage. Knowing whether a valid answer exists in a given text passage plays an important role in practical QA systems. We propose a nil-aware answer span extraction framework to return in a single

step an exact answer span to a question or *Nil*, depending on whether a valid answer exists or not. In addition to our previously proposed RC model (mentioned earlier in the first contribution), this framework can be readily integrated with many other recently published neural machine comprehension models. In total, we extend four machine comprehension models with our proposed framework and show that they achieve significantly better results compared to the other pipeline and threshold-based models.

- RC-based QA tasks share the single-turn setting of answering questions with the help of associated text passages, where the questions and their answers are independent of each other. However, humans naturally seek answers via conversation, which carries over context through the conversation flow. Although there is a surge in interest in conversational QA, recently proposed conversational QA systems lack the ability to ask a follow-up clarification question when a given question is underspecified. We propose a conversational QA system that can predict the answer to a question in a conversation, and has the ability to ask a follow-up clarification question if the original question is underspecified. We propose a pipeline approach that consists of an answer prediction model and a follow-up question generation model. The answer prediction model is based on a dual co-attention network while the follow-up question generation model is based on a sequence-to-sequence neural network enhanced with a copying mechanism. Experiments on the challenging ShARC dataset indicates that our proposed approach achieves

competitive performance. In contrast to a recently proposed ShARC-specific model, our proposed approach is generic and can be applied to any conversational question answering task which requires clarification question generation.

1.3 Organization of the Thesis

This thesis is organized as follows. In Chapter 2, we give an overview of related work on question answering. In Chapter 3, we describe the background and our proposed approach for answer span extraction in RC-based QA. In Chapter 4, we describe how several QA models developed for RC can be extended with our unified nil-aware answer extraction framework. In Chapter 5, we present our approach for conversational RC-based QA. Finally, we conclude the thesis in Chapter 6. Our work presented in Chapters 3 and 4 have been published as full papers in AAAI 2018 (Kundu and Ng, 2018a) and EMNLP 2018 (Kundu and Ng, 2018b) respectively.

Chapter 2

Related Work

This chapter presents a brief history of Question Answering (QA) and summarizes the related work on the current question answering systems.

2.1 Early QA Work

Research on QA spanned a history from closed-domain QA (Green et al., 1961; Simmons, 1965; Weizenbaum, 1966; Woods, 1973) to open-domain QA (e.g., Text REtrieval Conference TREC[1] 1999-2007). It attempts to deal with a wide range of question types including fact, list, definition, how, why, hypothetical, semantically-constrained, and cross-lingual questions. The challenges posed by answering different types of questions have been addressed by using a large variety of techniques, such as shallow and deep parsing, keyword extraction, named entity extraction, ontology, question typing, and machine learning of answer patterns appropriate to question forms (Hirschman and

[1]https://trec.nist.gov

Gaizauskas, 2001; Burger et al., 2001; Ng et al., 2001; Harabagiu and Hickl, 2006).

From the 1960s to the 1990s, most QA systems were developed to answer questions in a restricted domain. In the 1960s, BASEBALL answered questions about the US baseball league (Green et al., 1961). The information about every baseball event was stored in a list structure database and the question was also transformed into a list structure. Simmons (1965) surveyed fifteen QA systems including conversational question answerers and front-ends to databases. Most of these systems depended on semantic structure construction and canonical form matching in the databases. Weizenbaum (1966) developed ELIZA as a human-computer conversation system based on keywords and pattern matching. However, ELIZA was not robust enough to answer open-domain questions. From the 1970s, the development of comprehensive theories in computational linguistics led to the development of ambitious projects in text comprehension and QA. During this period, systems usually relied on human experts to hand-craft knowledge in a restricted domain. For instance, LUNAR was introduced which was about lunar science projects (Woods, 1973). LUNAR answered questions about moon rocks and soil gathered by the Apollo 11 mission. Another example was the Unix Consultant (UC), developed by Robert Wilensky at U.C. Berkeley in the 1980s (Wilensky, 1984). The system was able to answer questions related to the Unix operating system. It had a comprehensive hand-crafted domain-specific knowledge base, and it aimed at phrasing the answer to accommodate various types of users. Another noteworthy project was LILOG, a text-understanding system that operated on a German city tourism information domain (Bläsius,

1991).

In subsequent developments, QA systems aimed at making linguistic analysis of the questions to capture the intended requirements in a natural way. For instance, MASQUE (Androutsopoulos et al., 1993) represented natural language questions in a logic representation, and then it translated each logic query into a database query for retrieving the intended information from databases (Androutsopoulos et al., 1995). FAQ FINDER (Burke et al., 1997) matched the questions with a question list compiled in a knowledge base through statistical similarity and semantic similarity. Another QA System, QUARC, developed by Riloff and Thelen (2000), classified questions into different wh-types and derived their expected answer types through the use of lexical and semantic clues. Later, the focus of developing QA systems was shifted toward open domain QA.

2.2 TREC Question Answering

Since 1999, the annual Text REtrieval Conference (TREC) included a QA track which ran until 2007. This was the first large-scale open-domain QA system evaluation. The task of this track is to provide answers to natural language questions instead of only retrieving relevant documents. The answers to the questions are expected to be found from a large corpus of text. This competition had a significant impact on open-domain QA research and QA system architecture (Burger et al., 2001; Ferrucci et al., 2010).

Participants in the TREC-8 QA track (Voorhees, 1999) were given a set of factoid questions and they needed to return a ranked list of (*document*

id, answer string) pairs for each question. Two types of evaluation were performed based on the length of the answer string. An answer string could be up to 50 bytes or 250 bytes. The evaluation was done by human assessors. The assessors were instructed to make a binary decision based on whether the answer is present in the *answer string* or not. Mean reciprocal rank (MRR) was used for evaluation (Voorhees, 1999). Most participants used a three-step system: query formulation, candidate document retrieval from TREC document collection, and candidate answer extraction.

The overall structure of TREC-9 QA track (Voorhees, 2000) was similar to TREC-8, except that the document collection and the question set were larger, and the questions were taken from different sources. Similar to TREC-8, most of the participating teams followed the same three-step architecture to develop the systems. All the participating systems mostly improved their systems by adopting new techniques in these three components. The evaluation was also similar to TREC-8, using the MRR score. Additionally, the competing systems had a more robust question type classifier to tackle the reformulation of the same questions. The best systems in TREC-9 QA track were able to achieve 58% and 76% MRR scores for 50-byte and 250-byte answer length categories respectively.

The TREC 2001 QA track (Voorhees, 2001) was focused on three different tasks: the main task, list task, and context task. The main task was similar to the QA task in 2001, except that the answer length was limited to 50 bytes only. Additionally, the answers to some questions could not be found in the document collection (in which case NIL was considered as the correct answer). Most of the participating systems followed the same three-step ar-

chitecture consisting of query reformulation, document retrieval, and answer candidate extraction. Soubbotin (2001) obtained the best result in the main task. First, multiple overlapping answer-string candidates were created by selecting documents containing the query terms. Then, the system searched predefined patterns for different textual expressions. Finally, they ranked the answer candidates based on the presence of certain predefined patterns. In addition to the main task, a context task was added where the systems were required to answer a set of related questions about the same context. A list task was also added where the systems were required to provide a list of entities as the answer for a given question. However, most of the participating teams applied the same system designed for the main task without any major modifications.

The main task in TREC 2002 (Voorhees, 2002) was similar to 2001 except that the participating teams were asked to return only one exact answer instead of a ranked list of answers. Answer strings that contained redundant information or missed some portion of the answer were marked as incorrect by the assessors. The context task was excluded in TREC 2002. LCC's QA system submitted by Moldovan et al. (2002) performed best in both the main task and list task, outperforming the second-best system by a large margin. They used a customized parser to parse both the question and the retrieved documents, followed by transforming them into their corresponding logic forms. They used lexical chains to further transform the logic forms to axioms to incorporate semantic information about related concepts. Finally, a logic prover was used to rank the candidate answers guided by these axioms.

In TREC 2003 (Voorhees, 2003), the main task was divided into three

subtasks: factoid, list, and definition. To properly judge the answer quality of a definition question, a more rigorous evaluation criterion was introduced. In addition to the main task, a passage task was also introduced in this year to measure the participating systems' ability to retrieve short document extracts. Harabagiu et al. (2003) performed best in factoid and list questions. They improved their system by including a named entity recognizer to determine question types. After obtaining a set of potential candidate answers, they used a logic prover to remove incorrect candidate answers through abductive reasoning. BNN's QA system (Xu et al., 2003) achieved the best performance in definition questions using *kernel facts*. The QA system submitted from National University of Singapore (Zhang and Lee, 2003) also achieved good performance in all the tasks.

The main task of the TREC 2004 QA track (Voorhees, 2004) was similar to 2003, except that different types of questions were grouped together based on a specific target. The target can be viewed as a topic description for a particular group of questions. Hence, the participating systems were also required to deal with references and ellipses in these questions. Evaluation criteria were the same as the previous year. LCC's QA system (Moldovan et al., 2004) achieved the best result in the main task and the list task. The QA system from National University of Singapore (Cui et al., 2004) achieved the best performance in the definition task, and also performed quite well in the other two tasks.

The TREC 2005 QA track (Voorhees and Dang, 2005) consisted of three tasks: the main task, document ranking task, and relationship task. The main task was similar to that in 2004. The document ranking task was introduced

to evaluate the performance of the document retrieval module of the QA systems. In the relationship task, a topic was provided as the context of a question. The participating systems were expected to provide information nuggets as evidence for the answer. LCC's QA system (Harabagiu et al., 2005) performed the best in the main task. Litkowski (2005) performed the best in the relationship task. The QA system from National University of Singapore (Sun et al., 2005) performed quite well in both the main task and the document ranking task.

The TREC 2006 QA track (Dang et al., 2006) consisted of two tracks: the main task and a complex interactive task. The main task was similar to that in 2005. In the complex interactive task, each question consisted of a template part and a narrative part. The template part was a fixed question template with some free named entity slots (also called *facets*) that varied for each question. The narrative part included a few sentences describing the information need. A QA system from LCC (Moldovan et al., 2006) performed the best in both factoid task and list task. The QA system from the University of Edinburgh (Kaisser et al., 2006) was developed based on two complementary approaches and performed quite well on the list task. An automatic QA system from MIT (Katz et al., 2006) achieved impressive performance on the complex interactive QA task.

Similarly, the TREC 2007 QA track (Dang et al., 2007) consisted of a main task and a complex interactive question answering task. In the main task, in addition to news articles, the document collection for answer retrieval was augmented with blogs. As a result, the QA systems were required to deal with noisy blog texts. The complex relation task remained the same.

Moldovan et al. (2007) performed the best in the main task using question type-specific language models. Zhang et al. (2007) performed quite well on the complex interactive QA task by using a heuristic and a machine learning approach.

2.3 Community Question Answering

Community Question Answering (CQA) on web forums began to become popular with the introduction of several social network platforms such as Yahoo! Answers[2], Quora[3], Stack Overflow[4], etc. In community forums, a user can freely ask any questions and expect a variety of answers from other users. It takes effort for a user to read and understand all the answers of different quality. Often, a popular question receives hundreds of answers, and it is difficult for a user to read them all. Hence, it is beneficial to develop automated tools which can go through these CQA sites and automatically provide an answer for a given new question. The first step to automatically answer questions is to retrieve a set of questions that is similar to the user's question. This set of similar questions is then used to extract possible meaningful answers.

Question-question similarity

When a user asks a question in a community QA service, the user typically needs to choose a category label for the question from a predefined hierarchy of categories. Hence, each question in a community QA archive has a

[2]https://answers.yahoo.com/
[3]https://www.quora.com/
[4]https://stackoverflow.com/

category label and questions in community QA services are organized into hierarchies of categories. The questions in the same category or subcategory usually relate to the same general topic. Question-question similarity is typically addressed using different textual similarity measures. Several researchers have proposed methods using *explicit* question topic modeling, such as using question focus (Duan et al., 2008) or using a graph of topic terms (Cao et al., 2008). Researchers have also tackled this task with *implicit* question topic modeling. Cao et al. (2009) proposed a framework that is capable of exploiting classifications of questions in community QA archives for improving question search. They used a language model with smoothing based on the category structures of Yahoo! Answers. Zhang et al. (2014) used an LDA topic language model that matches the questions both at the term level and topic level. Another important type of approach is syntactic structure-based methods. Wang et al. (2009) proposed a retrieval model based on the similarity of syntactic trees. They extensively studied the structural representations of questions to encode lexical, syntactic and semantic features into the model. Notably, their model does not rely on training, and it is shown to be robust against grammatical errors as well. Da San Martino et al. (2016) studied the impact of several features for question ranking in community QA, such as bag-of-words models, syntactic tree kernels, and rank features. Recently, several neural network-based models have also been proposed for question-question similarity detection, e.g., dos Santos et al. (2015) used convolutional neural networks (CNN), Romeo et al. (2016) used long short-term memory (LSTM) networks with neural attention, and Lei et al. (2016) used a combined network of CNN and recurrent neural network (RNN).

Question-answer similarity

Question-answer similarity is a well-researched subtask in the general Question Answering (QA) task. Many methods have been proposed in the past that try to match the syntactic structure of a question to that of the candidate answer. Wang et al. (2007) proposed a probabilistic quasi-synchronous grammar to learn the syntactic transformation from a question to a candidate answer. Heilman and Smith (2010) proposed a tree edit distance (TED)-based algorithm to learn tree transformation in pairs. Wang and Manning (2010) proposed a probabilistic model to learn tree-edit operations on dependency parse trees. Yao et al. (2013) derived features from TED and used a linear chain conditional random field (CRF) to learn associations between questions and candidate answers. Recently, many methods have been proposed which use neural network models for answer sentence selection (Feng et al., 2015; Severyn and Moschitti, 2015; Wang and Nyberg, 2015; Filice et al., 2016; dos Santos et al., 2016; Mohtarami et al., 2016; Yang et al., 2016; Bian et al., 2017). Tan et al. (2015) proposed to use a neural attention mechanism with bidirectional LSTMs to generate better candidate answer representations given a question. Tymoshenko et al. (2016) combined neural networks with syntactic kernels for question-answer similarity modeling.

In question-answer similarity task, it always assumes that there exists a list of candidate answers where each answer is a single sentence. Although this might be useful in community QA, it is not practical for general QA systems, where the answers are not always full sentences, i.e., they are often text fragments.

2.4 Reading Comprehension-based QA

In this section, we provide an overview of reading comprehension (RC)-based QA. In the RC task, a machine reads a story (e.g., a text passage or document), and demonstrates its understanding by answering questions about the story. Since questions can be devised to query any aspect of text comprehension, the ability to answer questions in the reading comprehension scenario requires a high degree of natural language understanding. The history of building machine reading comprehension systems dates back to many years ago. One of the most notable early works is the QUALM system developed by Lehnert (1977). In QUALM, Lehnert (1977) devised a theory of question answering and focused on the importance of the context of the story in responding to questions. Although this early work set a strong vision for natural language understanding, the actual system built at that time was limited to hand-coded scripts, and difficult to generalize to broader domains. Due to the complexity of the problem, research on RC-based QA was mostly neglected in the 1980s and 1990s. In the late 1990s, there was some small revival of interest in RC-based QA. Hirschman et al. (1999) created a reading comprehension dataset, followed by a Workshop on Reading Comprehension Tests as Evaluation for Computer-based Understanding Systems at ANLP/NAACL 2000. The dataset consists of 60 stories for development and 60 stories for testing of 3rd to 6th grade material. This dataset primarily contains *who*, *what*, *when*, *where*, and *why* questions. It only requires systems to return a sentence that contains the right answer. The systems developed at this stage were mostly rule-based bag-of-words approaches with shallow linguistic processing such

as stemming, semantic class identification, and pronoun resolution in the DEEP-READ system (Hirschman et al., 1999), or manually generated rules based on lexical and semantic correspondence (Riloff and Thelen, 2000), or their combinations (Charniak et al., 2000). However, Ng et al. (2000) used a machine learning approach for answering reading comprehension questions and evaluated their approach on the same dataset.

After 2010, there were many efforts to formulate RC-based QA as a supervised learning problem. Researchers collected human-labeled training examples in the form of (passage, question, answer) triples for training statistical models that learn to map a passage and question pair into their corresponding answer. The existing reading comprehension tasks can be divided into four categories depending on the answer type:

- **Multiple choice**: In this category, the correct answer is chosen from a candidate set of hypothesized answers. Typically, accuracy is used for evaluating models in this task.

- **Cloze style**: In this case, the question contains a placeholder, also known as a fill-in-the-blank question. In these tasks, the systems must guess which word or entity completes the question, based on the passage, and the answer is either chosen from a pre-defined set of choices or the full vocabulary. Similar to the previous category, accuracy is used for evaluation.

- **Span extraction**: In this category, the answer must be a single continuous span in the associated passage. For these tasks, typically two evaluation metrics are used: (1) **Exact Match** assigns full credit 1 if

the predicted answer exactly matches the gold answer, and 0 otherwise, and (2) **F1 score** computes the average word overlap between predicted and gold answers. It treats the predicted and gold answers as bags of tokens, and compute their F1.

- **Free-form answer**: The last category allows the answer to be any free-text form, i.e., a word sequence of arbitrary length. In these tasks, typically standard evaluation metrics for natural language generation tasks are used, e.g., BLEU (Papineni et al., 2002), Meteor (Banerjee and Lavie, 2005), and ROUGE (Lin, 2004).

Two notable datasets created after 2010 are MCTest (Richardson et al., 2013) and ProcessBank (Berant et al., 2014). In the MCTest dataset, Richardson et al. (2013) collected 660 fictional stories, with 4 multiple choice questions per story where each question is associated with 4 candidate answers and one of them is correct. ProcessBank is designed to answer binary-choice questions in a passage describing a biological process and requires an understanding of the relations between entities and events in the process. The ProcessBank dataset consists of 585 questions spread over the 200 passages. These datasets have inspired a strand of machine learning models (Sachan et al., 2015; Narasimhan and Barzilay, 2015; Wang et al., 2015). These models were mostly developed on top of a max-margin learning framework using a rich set of hand-engineered linguistic features, including syntactic dependencies, semantic frames, coreference resolution, discourse relations, and word embeddings. Although these models provided modest improvements over rule-based models, they have several limitations. For instance, it is of-

ten very difficult to construct effective hand-engineered features when the evidence is spread over the passage. Further, these models relied heavily on existing linguistic tools such as dependency parsers and semantic role labeling systems. However, these linguistic representation tasks often inject additional errors and off-the-shelf tools are often trained from one single domain and suffer from generalization problems in a practical scenario.

This field witnessed a significant change when Hermann et al. (2015) proposed an automatic way of creating large-scale labeled training data for training machine reading comprehension models. They created two large-scale cloze style datasets, namely CNN and DailyMail. They also proposed a neural network model, namely Attentive Reader, and demonstrated that it outperformed symbolic NLP approaches by a large margin. In their experiments, the Attentive Reader achieved 63.8% accuracy while symbolic NLP systems obtained 50.9% at most on the CNN dataset. Hill et al. (2016) also created a large-scale cloze-style Children's Book Test dataset, designed to measure directly how well models can exploit a wider linguistic context.

Although several models were proposed using the advanced neural network architectures after the introduction of the CNN / DailyMail dataset, Chen et al. (2016) showed that the dataset requires less reasoning than previously thought, and conclude that performance on this dataset is almost saturated. Subsequently, Rajpurkar et al. (2016) released a large and high-quality span extraction-based SQuAD dataset, where the answers are free-form text fragments unlike in the previous datasets. The answers in SQuAD often include non-entities and can be much longer phrases, making the SQuAD dataset more challenging than previous cloze-style datasets. Most of the pre-

viously released datasets are closed-world, i.e., the questions and answers are formulated given the text passages. Hence, the answer spans can often be extracted by simple word and context matching without requiring any deeper understanding. To address these weaknesses, several other datasets were created such as NewsQA (Trischler et al., 2017), TriviaQA (Joshi et al., 2017), SearchQA (Dunn et al., 2017), etc. However, most of these datasets do not consider nil questions, i.e., where no valid answer is present in the associated passage for a given question. Trischler et al. (2017) first came up with a large-scale RC dataset which includes nil questions. More recently, Rajpurkar et al. (2018) augmented the SQuAD dataset by including unanswerable questions.

The reading comprehension field has become one of the most active fields in NLP today and it has further evolved. Following the theme of creating large-scale and more challenging reading comprehension datasets, several other datasets have been created recently from a variety of sources. For instance, Lai et al. (2017) created the RACE dataset. RACE is a multiple-choice RC dataset collected from the English exams for middle-school and high-school Chinese students. All the questions and answer candidates were created by experts. As a result, the dataset is more difficult than many existing RC datasets. Recently, several datasets have been created that require multi-hop reasoning (Welbl et al., 2018; Khashabi et al., 2018; Yang et al., 2018). In multi-hop RC-based QA, a system needs to combine information from multiple sentences or passages in order to arrive at the answer. Nguyen et al. (2016) introduced a large-scale machine reading comprehension dataset (MS MARCO), which encourages systems to generate free-form answers. To encourage progress on deeper comprehension of language, Kočiský

et al. (2018) presented a new dataset (NarrativeQA) and a set of tasks in which the reader must give free-form answers to questions about stories by reading entire books or movie scripts.

2.5 Reading Comprehension Using Deep Neural Networks

In recent years, the QA research community has witnessed some significant progress. Benchmark datasets play an important role in this recent progress in reading comprehension and question answering research. Most of the current top-performing systems on RC-based QA tasks are built on deep end-to-end neural networks. These models usually start from the idea of representing every single token in the associated passage and the question as a dense vector, passing through several modeling or interaction layers, and finally making predictions. All the model parameters can be optimized jointly using the gradient descent algorithm or its variants. In contrast to the feature-engineered classifiers, neural RC models have several advantages:

- Deep neural network-based models do not require labor-intensive, manual engineering of features. Therefore, the models are conceptually simpler.

- In conventional feature-engineered models, features are usually very sparse and do not generalize well. Deep neural network-based models alleviate the sparsity issue by using dense word embeddings. They also generalize well as the features are learned automatically.

- Deep neural network-based models learn all the features on their own. They do not rely on any downstream language processing modules, such as dependency parsing and coreference resolution. This can potentially avoid additional errors due to the intermediate feature extractors, such as a dependency parser.

Since the release of the large-scale cloze-style datasets, several end-to-end neural network models (Hermann et al., 2015; Hill et al., 2016; Kadlec et al., 2016; Sordoni et al., 2016; Dhingra et al., 2017; Kobayashi et al., 2016; Shen et al., 2017a; Cui et al., 2017; Chen et al., 2016) have been proposed. Hermann et al. (2015) first introduced an attention mechanism into machine reading comprehension. Hill et al. (2016) proposed a window-based memory network for the Children's Book Test dataset. Kadlec et al. (2016) proposed the attention sum reader (ASR) which improved the state of the art by a significant margin. ASR first calculates the attention weight for every word in the passage and then chooses the answer which has the maximum sum of attention weights. The attention for every word in the document is calculated by encoding the question and document by separate bi-directional LSTMs (Hochreiter and Schmidhuber, 1997). Sordoni et al. (2016) proposed an alternating attention mechanism that allows a fine-grained exploration of both the question and the passage. Gated attention (GA) reader (Dhingra et al., 2017) extends the AS reader and performs multiple hops over the input passage representation. It uses the same pointer sum attention mechanism as ASR in the output layer to obtain the distribution over candidate answers. Kobayashi et al. (2016) proposed the dynamic entity representation (DER) network, which explicitly reads dynamic meaning representations for enti-

ties by accumulating information around the entities as it keeps reading a passage. Shen et al. (2017a) proposed the ReasoNet model which iteratively infers the answer with a dynamic number of reasoning steps and is trained with reinforcement learning. Cui et al. (2017) proposed a simple but effective attention-over-attention mechanism to capture the interaction between passage and question. Chen et al. (2016) proposed the Stanford attentive reader (SAR) which is primarily based on the idea of ASR. They also explained that the cloze-style task on the CNN/DailyMail dataset is not challenging enough for the recently developed advanced neural network models and hence, advanced models had to be evaluated on more realistic datasets.

Since the release of answer span extraction-based datasets (Rajpurkar et al., 2016; Trischler et al., 2017; Joshi et al., 2017), many end-to-end neural network models were proposed. Models based on the idea of chunking and ranking include Yu et al. (2016) and Lee et al. (2016). Multiple models (Wang and Jiang, 2017; Yang et al., 2017; Trischler et al., 2017; Xiong et al., 2017; Seo et al., 2017; Wang et al., 2016a; Weissenborn et al., 2017; Wang et al., 2017; Pan et al., 2017; Shen et al., 2017b; Chen et al., 2017) were proposed which are based on the idea of a pointer network (Vinyals et al., 2015). Wang and Jiang (2017) used a Match-LSTM to encode the question and passage together and a boundary model determined the beginning and ending boundary of an answer. Yang et al. (2017) used a fine-grained gating mechanism to capture the correlation between a passage and a question. Trischler et al. (2017) reimplemented Match-LSTM for the NewsQA dataset and proposed a faster version of it. Xiong et al. (2017) used a co-attentive encoder followed by a dynamic decoder to iteratively estimate the boundary pointers.

Seo et al. (2017) proposed a bi-directional attention flow approach to capture the interactions between passages and questions. Wang et al. (2016a) proposed a multi-perspective context matching approach that explicitly matches the contextual embeddings of the passage with the question from multiple perspectives. Weissenborn et al. (2017) proposed a simple context matching-based neural encoder and incorporated word overlap and term frequency features to estimate the start and end pointers. Wang et al. (2017) proposed a gated self-matching approach which encodes the passage and question together using a self-matching attention mechanism. Pan et al. (2017) proposed a memory network-based multi-layer embedding model. Shen et al. (2017b) proposed a reasoning network that produces the answer after multiple iterations of the comprehension process. Chen et al. (2017) proposed a multi-layer recurrent neural network model for answer span extraction. However, most of these models primarily rely on simple context matching without the ability to capture long-range dependency.

Moreover, most of the recent work always assumes that a valid answer can always be found in the associated text passage for a given question. However, in practice, there may not exist any valid answer in the associated passage for a question (referred to as nil questions). Although SQuAD (Rajpurkar et al., 2016) became very popular and served as a good test set to develop advanced end-to-end neural network architectures, it does not include any nil questions. In SQuAD, questions and answers are formulated given text passages. Hence, a valid answer can always be found in the associated passage for every question created. In practical QA, it is critical to decide whether or not a passage contains a valid answer for a given question. Subsequently,

Trischler et al. (2017) proposed a more challenging and realistic dataset, NewsQA, where the questions were formed using CNN article summaries without accessing the original full texts. As such, some questions have no valid answers in the associated passages. However, prior models for NewsQA excluded nil questions during evaluation. We focus on developing QA systems that extract an answer for a question if and only if the associated passage contains a valid answer. Otherwise, they are expected to return *Nil* as the answer. Very recently, Rajpurkar et al. (2018) augmented the SQuAD dataset with unanswerable questions which is also known as SQuAD 2.0.

2.6 Multi-turn Conversational Reading Comprehension-based QA

So far, we have discussed single-turn QA, where a machine provides an answer for a single question. In this section, we provide an overview of multi-turn conversational QA.

Conversational question answering is directly related to dialog. Building conversational agents or dialog systems to converse with humans in natural language is one of the major objectives of natural language understanding. dialog systems can be broadly classified into two categories: task-oriented, and chit-chat dialog agents. Task-oriented dialog systems are designed for one particular task and set up to have short conversations (e.g., IT help desk, booking a flight or making a restaurant reservation). In contrast, chit-chat dialog systems do not have a specific goal and aim for extended, casual con-

versations. Usually, the conversations are longer for these systems. Answering questions is also a core task of dialog systems since one of the most common needs for humans to interact with dialog agents is to seek information and ask questions of various topics. QA-based dialog techniques have been developed extensively in many automated virtual personal assistant systems, either based on structured knowledge bases or unstructured text collections. Dialog systems in recent research papers are mostly built on top of deep neural networks.

In recent years, several datasets and neural models have been developed to improve dialog systems (Young et al., 2013; Shawar and Atwell, 2007). Many data-driven machine learning methods have been shown to be effective for tasks relevant for dialog such as dialog policy learning (Young et al., 2013), dialog state tracking (Henderson et al., 2013; Williams et al., 2013; Kim et al., 2016), and natural language generation (Sordoni et al., 2015; Li et al., 2016a; Bordes et al., 2017). Most of the recent dialog systems are either not goal-oriented (e.g., simple chit-chat bots), or domain-specific if they are goal-oriented (e.g., IT help desk, booking a flight or making a restaurant reservation). While the dialog systems, such as chit-chat bots, are developed to generate any free-form text, we particularly focus on clarification question generation in a more complex conversational QA setting.

Recently, there has been a surge of interest in conversational QA. Saha et al. (2018) released a Complex Sequential Question Answering (CSQA) dataset which combines the two tasks of answering factoid questions by inference over a knowledge graph and learning conversations through a series of interrelated QA pairs. In addition to the dataset, Saha et al. (2018) pro-

posed a sequence-to-sequence model, which uses a hierarchical encoder and a key-value memory network for encoding, and then uses a recurrent neural network as the decoder to produce answers. Guo et al. (2018) introduced a dialog memory management module that leverages historical entities, predicates, and action subsequences when generating the logical form for an utterance. Recently, Choi et al. (2018) released a conversational QA dataset, namely question answering in context (QuAC), which is developed over plain text passages and mimics a student-teacher interactive scenario. Elgohary et al. (2018) released the QBLink dataset for sequential question answering in an open-domain setting. QBLink is a sequential question answering dataset collected from Quiz Bowl tournaments, where a sequence contains multiple related questions. These questions are related to the same concept while not focusing on the dialog aspects (e.g., coreference). Zhou et al. (2018) created another chit-chat style dialog dataset based on a single movie-related Wikipedia article, in which two workers are asked to chat about the content. Reddy et al. (2019) released a large-scale CoQA dataset for building conversational QA systems. In contrast to prior datasets, CoQA contains free-form answers. Since the release of CoQA, many systems evaluated on this dataset have appeared on the leaderboard[5]. Zhu et al. (2018) proposed a model that fuses the conversation history into traditional reading comprehension models. It relies on both inter-attention and self-attention to comprehend the conversation context and extract relevant information from the passage. Huang et al. (2019) proposed the FlowQA model that can incorporate intermediate representations generated during the process of answering previous questions,

[5]https://stanfordnlp.github.io/coqa/

through an alternating parallel processing structure. Yeh and Chen (2019) proposed the FlowDelta model that can explicitly capture the information gain through dialog reasoning in order to allow the model to focus on more informative cues.

Although there has been a significant advancement in the area of conversational QA, most of the prior works only focus on answering questions. They do not possess the capability to ask a follow-up clarification question if a question in the conversation is underspecified. Saeidi et al. (2018) created the ShARC dataset from regulatory texts (e.g., traffic rules, tax and visa regulations, etc) for conversational QA. ShARC requires a system to ask a follow-up clarification question if a question is underspecified in a conversation. In this thesis, we focus on the ShARC dataset to develop a conversational QA system that can answer questions and is able to ask clarification questions when the posed questions are underspecified.

2.7 Summary

In summary, we have provided an overview of the related work in different types of QA. We provided a brief history of early QA work. We also presented a detailed overview of TREC QA tasks. We briefly described the related work in community QA. Next, we provided a thorough description of single-turn reading comprehension-based QA tasks and proposed models. We also present the related work in multi-turn conversational QA. In the following chapters, we present our proposed models for single-turn and multi-turn reading comprehension-based QA tasks.

Chapter 3

A Question-Focused Multi-Factor Attention Network for Question Answering

In this chapter, we propose a novel end-to-end question-focused multi-factor attention network for answer extraction. Multi-factor attentive encoding using tensor-based transformation aggregates meaningful facts even when they are located in multiple sentences. To implicitly infer the answer type, we also propose a max-attentional question aggregation mechanism to encode a question vector based on the important words in a question. During prediction, we incorporate sequence-level encoding of the first wh-word and its immediately following word as an additional source of question type infor-

mation. Our proposed model achieves significant improvements over the best prior state-of-the-art results on three large-scale challenging QA datasets, namely NewsQA, TriviaQA, and SearchQA. In order to make it easier for other researchers to replicate our results, we have made the source code of our system publicly available[1].

The rest of this chapter is organized as follows. We start with some preliminary descriptions of some key elements in modern neural network-based NLP models in Section 3.1. Then, we provide the background of this work in Section 3.2. The problem definition is given in Section 3.3. Then, we provide the details of our approach in Section 3.4, followed by visualization of our proposed components in Section 3.5. We present and discuss the results of our experimental evaluation in Section 3.6. In Section 3.7, we provide a brief overview of more recent advances in RC-based QA. Finally, we summarize this chapter in Section 3.8.

3.1 Preliminaries

In this section, we outline a minimal set of elements and the key ideas which form the basis of modern deep neural network-based NLP models.

3.1.1 Word Embedding

Prior to deep learning, it was common to represent a word by a one-hot vector, i.e., each word is represented by a high-dimensional, sparse vector where only one entry of that vector is 1 and the remaining entries are 0s.

[1]https://github.com/nusnlp/amanda

The primary drawback of this one-hot representation is that it does not capture any semantic similarity between words. In deep neural network-based models, each word is typically represented by a low-dimensional, real-valued vector. Low-dimensional word embedding vectors help to capture the semantic similarity between words to the extent that similar words can be encoded as similar vectors in the geometric space.

The word embedding vectors can be effectively learned from large unlabeled text corpora, based on the assumption that words occurring in similar contexts tend to have similar meanings. Representing a word as a vector has a long history and has been finally popularized by recent scalable algorithms and released sets of pre-trained word embeddings such as Word2Vec (Mikolov et al., 2013), GloVe (Pennington et al., 2014), etc. They have become an integral part of modern deep learning-based NLP systems.

3.1.2 Recurrent Neural Network

Another important module is Recurrent Neural Network (RNN) to model sentences, passages, or documents in many NLP tasks. RNNs are a class of neural networks that are suitable for handling sequences of variable length. Specifically, they apply a parameterized function recursively on a sequence of words $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T\}$ to obtain a contextual encoding representation. Mathematically, it can be represented as follows:

$$\mathbf{h}_t \;=\; \mathrm{f}(\mathbf{h}_{t-1}, \mathbf{x}_t, \Phi) \;, \tag{3.1}$$

where $\mathbf{h}_t$ is the hidden vector representation for the tth word and Φ is a set of trainable parameters.

In NLP tasks, typically each word in a sentence, passage, or document is transformed into a real-valued vector through embeddings, i.e., $\mathbf{x}_t \in \mathbb{R}^D$. $\mathbf{h}_t \in \mathbb{R}^H$ is used to capture contextual information. A typical RNN can be represented as:

$$\mathbf{h}_t \;=\; \tanh(\mathbf{h}_{t-1}\mathbf{W}^h \;+\; \mathbf{x}_t\mathbf{W}^x \;+\; \mathbf{b}) \;\;, \tag{3.2}$$

where $\mathbf{W}^h \in \mathbb{R}^{H\times H}$, $\mathbf{W}^x \in \mathbb{R}^{D\times H}$, $\mathbf{b} \in \mathbb{R}^H$ are learnable parameters. For effective optimization and to cope with the challenges of training an RNN (e.g., vanishing gradient issue), several variants have been proposed. Notably, Long-Short Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Unit (GRU) (Chung et al., 2014) are the two most commonly used RNNs recently. Arguably, LSTM is still the most competitive RNN variant for many NLP applications today and also our default choice for the neural models that we will describe in this thesis. Mathematically, LSTMs can be formulated as follows:

$$\begin{aligned}
\mathbf{i}_t &= \sigma(\mathbf{h}_{t-1}\mathbf{W}^{ih} \;+\; \mathbf{x}_t\mathbf{W}^{ix} \;+\; \mathbf{b}^i) \\
\mathbf{f}_t &= \sigma(\mathbf{h}_{t-1}\mathbf{W}^{fh} \;+\; \mathbf{x}_t\mathbf{W}^{fx} \;+\; \mathbf{b}^f) \\
\mathbf{o}_t &= \sigma(\mathbf{h}_{t-1}\mathbf{W}^{oh} \;+\; \mathbf{x}_t\mathbf{W}^{ox} \;+\; \mathbf{b}^o) \\
\mathbf{g}_t &= \tanh(\mathbf{h}_{t-1}\mathbf{W}^{gh} \;+\; \mathbf{x}_t\mathbf{W}^{gx} \;+\; \mathbf{b}^g) \\
\mathbf{c}_t &= \mathbf{f}_t \odot \mathbf{c}_{t-1} \;+\; \mathbf{i}_t \odot \mathbf{g}_t \\
\mathbf{h}_t &= \mathbf{o}_t \odot \tanh(\mathbf{c}_t)
\end{aligned} \tag{3.3}$$

where $\odot$ represents the element-wise multiplication and σ represents the sigmoid function. $\mathbf{W}^{ih}, \mathbf{W}^{fh}, \mathbf{W}^{oh}, \mathbf{W}^{gh} \in \mathbb{R}^{H \times H}$, $\mathbf{W}^{ix}, \mathbf{W}^{fx}, \mathbf{W}^{ox}, \mathbf{W}^{gx} \in \mathbb{R}^{D \times H}$, and $\mathbf{b}^{i}, \mathbf{b}^{f}, \mathbf{b}^{o}, \mathbf{b}^{g} \in \mathbb{R}^{H}$ are learnable parameters.

3.1.3 Attention Mechanism

Another important aspect of modern deep learning-based NLP models is the use of attention mechanisms. Attention was first introduced in the sequence-to-sequence models (Sutskever et al., 2014) for neural machine translation (Bahdanau et al., 2015) and has later been extended to many other NLP tasks. When we use RNNs to encode a sequence of words, typically, the last hidden vector of the sequence is used to predict the label for a particular task. This requires the model to be able to compress all the necessary information into a fixed-length vector, which causes an information bottleneck in improving performance (Cho et al., 2014). In contrast, the attention mechanism looks at the hidden state vectors for all time steps and chooses a subset of these vectors adaptively. A typical attention mechanism can be mathematically illustrated as follows:

$$\alpha_t = \frac{\exp\left(\text{Attn}(\mathbf{h}_t, \mathbf{g}; \Phi_{attn})\right)}{\sum_{t'=1}^{T} \exp\left(\text{Attn}(\mathbf{h}_{t'}, \mathbf{g}; \Phi_{attn})\right)} \tag{3.4}$$

$$\tilde{\mathbf{h}} = \sum_{t=1}^{T} \alpha_t \mathbf{h}_t \quad , \tag{3.5}$$

where α_t is the attention score for the tth word and $\tilde{\mathbf{h}} \in \mathbb{R}^{H}$ is the aggregated representation of the sequence of words. Φ_{attn} is the set of learnable weight parameters. Attn(.) represents the attention scoring function with respect to

a key vector $\mathbf{g} \in \mathbb{R}^H$. It is a parametric function which can be chosen in several ways, such as dot product, bilinear product, or one hidden layer of a feed-forward layer (FFL):

$$\text{Attn}_{\text{DOT}}(\mathbf{h}_t, \mathbf{g}) \;=\; \mathbf{h}_t \mathbf{g}^\top \tag{3.6}$$

$$\text{Attn}_{\text{BILIN}}(\mathbf{h}_t, \mathbf{g}) \;=\; \mathbf{h}_t \mathbf{W} \mathbf{g}^\top \tag{3.7}$$

$$\text{Attn}_{\text{FFL}}(\mathbf{h}_t, \mathbf{g}) \;=\; (\mathbf{h}_t \mathbf{W}^h \;+\; \mathbf{g}\mathbf{W}^g)\; \mathbf{w}^\top \tag{3.8}$$

where $\mathbf{W} \in \mathbb{R}^{H \times H}$, $\mathbf{W}^h \in \mathbb{R}^{H \times H}$, $\mathbf{W}^g \in \mathbb{R}^{H \times H}$, and $\mathbf{w} \in \mathbb{R}^H$ are learnable parameters.

Intuitively, an attention mechanism computes a similarity score for every contextual word representation, followed by applying a softmax function, resulting in a discrete probability distribution over all the words. In this way, α essentially captures which parts of the source sequence are more relevant. $\tilde{\mathbf{h}}$ aggregates over all the time steps with a weighted sum and can be used for final prediction.

3.2 Background

In machine comprehension (MC)-based question answering (QA), a machine is expected to provide an answer for a given question by understanding texts.

Recently, many neural models have been proposed which mostly focus on passage-question interaction to capture the context similarity for extracting a text span as the answer. However, most of the models do not focus on

synthesizing evidence from multiple sentences and fail to perform well on challenging open-world QA tasks such as NewsQA and TriviaQA. Moreover, none of the models explicitly focus on question/answer type information for predicting the answer. In practice, fine-grained understanding of the question/answer type plays an important role in QA.

In this work, we propose an end-to-end question-focused multi-factor attention network for document-based question answering (AMANDA), which learns to aggregate evidence distributed across multiple sentences and identifies the important question words to help extract the answer. Intuitively, AMANDA extracts the answer not only by synthesizing relevant facts from the passage but also by implicitly determining the suitable answer type during prediction. The key contributions of this work reported in this chapter are:

- We propose a multi-factor attentive encoding approach based on tensor transformation that can capture long-range dependency to synthesize meaningful evidence across multiple sentences.

- To subsume fine-grained answer type information, we propose a max-attentional question aggregation mechanism that learns to identify the meaningful portions of a question. We also incorporate sequence-level representations of the first wh-word and its immediately following word in a question as an additional source of question type information.

3.3 Problem Definition

Given a pair of passage and question, an MC system needs to extract a text span from the passage as the answer. We formulate the answer as two pointers in the passage, which represent the beginning and ending tokens of the answer. Let $\mathcal{P}$ be a passage with tokens $(\mathcal{P}_1, \mathcal{P}_2, \ldots, \mathcal{P}_T)$ and $\mathcal{Q}$ be a question with tokens $(\mathcal{Q}_1, \mathcal{Q}_2, \ldots, \mathcal{Q}_U)$, where T and U are the length of the passage and question respectively. To answer the question, a system needs to determine two pointers in the passage, b and e, such that $1 \leq b \leq e \leq T$. The resulting answer tokens will be $(\mathcal{P}_b, \mathcal{P}_{b+1}, \ldots, \mathcal{P}_e)$.

3.4 Network Architecture

The architecture of the proposed question-focused multi-factor attention network is given in Figure 3.1.

3.4.1 Word-level Embedding

Word-level embeddings are formed by two components: pre-trained word embedding vectors from GloVe (Pennington et al., 2014) and convolutional neural network-based (CNN) character embeddings (Kim, 2014). Character embeddings have proven to be very useful for out-of-vocabulary (OOV) words. We use a character-level CNN followed by max-pooling over an entire word to get the embedding vector for each word. Prior to that, a character-based lookup table is used to generate the embedding for every character and the lookup table weights are learned during training. We concatenate these two

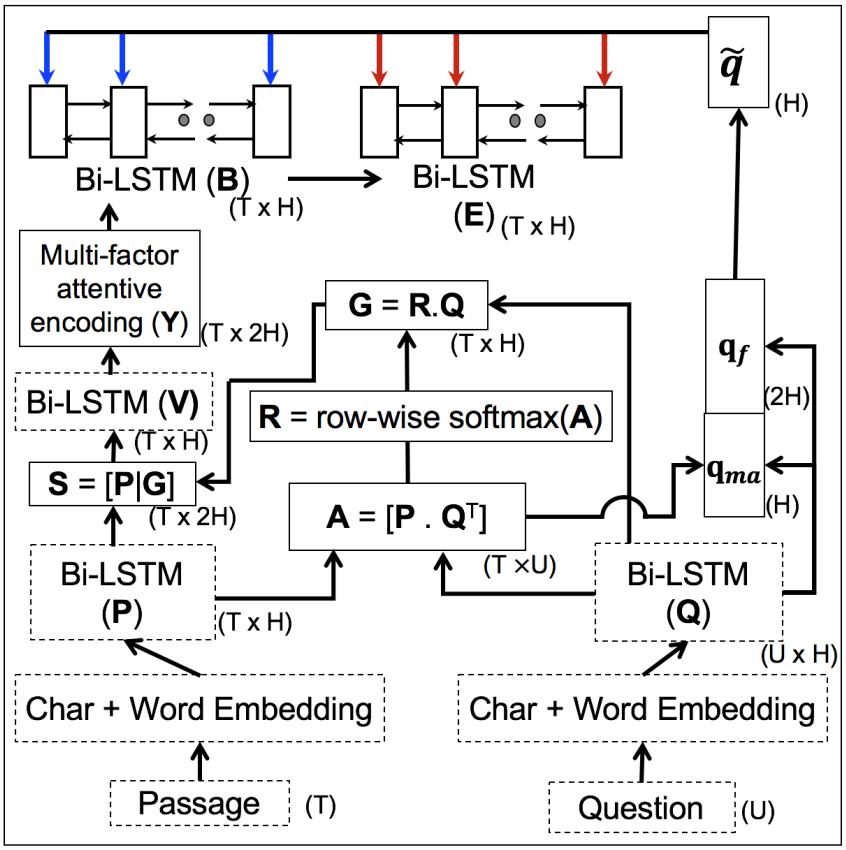


Figure 3.1: Architecture of the proposed model. T, U, and H represent the number of passage tokens, number of question tokens, and the number of hidden units of the Bi-LSTMs, respectively. Hidden unit representations of Bi-LSTMs, **B** and **E**, are shown to illustrate the answer pointers. Blue and red arrows represent the start and end answer pointers respectively.

embedding vectors for every word to generate word-level embeddings.

3.4.2 Sequence-level Encoding

We apply sequence-level encoding to incorporate contextual information. Let $\mathbf{e}_t^p$ and $\mathbf{e}_t^q$ be the tth embedding vectors of the passage and the question respectively. The embedding vectors are fed to a bi-directional LSTM (BiLSTM) (Hochreiter and Schmidhuber, 1997). Considering that the out-

puts of the BiLSTMs are unfolded across time, we represent the outputs as $\mathbf{P} \in \mathbb{R}^{T \times H}$ and $\mathbf{Q} \in \mathbb{R}^{U \times H}$ for passage and question respectively. H is the number of hidden units for the BiLSTMs. At every time step, the hidden unit representation of the BiLSTMs is obtained by concatenating the hidden unit representations of the corresponding forward and backward LSTMs. For the passage, at time step t, the forward and backward LSTM hidden unit representations can be written as:

$$\begin{aligned} \overrightarrow{\mathbf{h}}_t^p &= \overrightarrow{\text{LSTM}}(\overrightarrow{\mathbf{h}}_{t-1}^p,\ \mathbf{e}_t^p) \\ \overleftarrow{\mathbf{h}}_t^p &= \overleftarrow{\text{LSTM}}(\overleftarrow{\mathbf{h}}_{t+1}^p,\ \mathbf{e}_t^p) \end{aligned} \tag{3.9}$$

The tth row of $\mathbf{P}$ is represented as $\mathbf{p}_t = \overrightarrow{\mathbf{h}}_t^p \ || \ \overleftarrow{\mathbf{h}}_t^p$, where $||$ represents the concatenation of two vectors. Similarly, the sequence level encoding for a question is $\mathbf{q}_t = \overrightarrow{\mathbf{h}}_t^q \ || \ \overleftarrow{\mathbf{h}}_t^q$, where $\mathbf{q}_t$ is the tth row of $\mathbf{Q}$.

3.4.3 Cartesian Similarity Layer

The similarity matrix is calculated by taking dot products between all possible combinations of sequence-level encoding vectors for a passage and a question. Note that for calculating the similarity matrix, we do not introduce any additional learnable parameters. The similarity matrix $\mathbf{A} \in \mathbb{R}^{T \times U}$ can be expressed as:

$$\mathbf{A} = \mathbf{P}\ \mathbf{Q}^\top \tag{3.10}$$

Intuitively, $A_{i,j}$ is a measure of the similarity between the sequence-level encoding vectors of the ith passage word and the jth question word.

3.4.4 Question-dependent Passage Encoding

In this step, we jointly encode the passage and question. We apply a row-wise softmax function on the similarity matrix:

$$\mathbf{R} = \text{row-wise softmax}(\mathbf{A}) \tag{3.11}$$

If $\mathbf{r}_t \in \mathbb{R}^U$ is the tth row of $\mathbf{R} \in \mathbb{R}^{T \times U}$, then $\sum_{j=1}^{U} r_{t,j} = 1$. Each row of $\mathbf{R}$ measures how relevant every question word is with respect to a given passage word. Next, an aggregated question vector is computed corresponding to each sequence-level passage word encoding vector. The aggregated question vector $\mathbf{g}_t \in \mathbb{R}^H$ corresponding to the tth passage word is computed as $\mathbf{g}_t = \mathbf{r}_t\mathbf{Q}$. The aggregated question vectors corresponding to all the passage words can be computed as $\mathbf{G} = \mathbf{R}\ \mathbf{Q}$, where $\mathbf{g}_t$ is the tth row of $\mathbf{G} \in \mathbb{R}^{T \times H}$.

The aggregated question vectors corresponding to the passage words are then concatenated with the sequence-level passage word encoding vectors. If the question-dependent passage encoding is denoted as $\mathbf{S} \in \mathbb{R}^{T \times 2H}$ and $\mathbf{s}_t$ is the tth row of $\mathbf{S}$, then $\mathbf{s}_t = \mathbf{c}_t \mid\mid \mathbf{g}_t$, where $\mathbf{c}_t$ is the sequence-level encoding vector of the tth passage word (tth row of $\mathbf{P}$). Then a BiLSTM is applied on $\mathbf{S}$ to obtain $\mathbf{V} \in \mathbb{R}^{T \times H}$, which is later used as input for multi-factor attentive encoding.

3.4.5 Multi-factor Attentive Encoding

Tensor-based neural network approaches have been used in a variety of natural language processing tasks (Pei et al., 2014; Li et al., 2016b). We propose a

multi-factor attentive encoding approach using tensor-based transformation. In practice, recurrent neural networks fail to remember information when the context is long. Our proposed multi-factor attentive encoding approach helps to aggregate meaningful information from a long context with fine-grained inference due to the use of multiple factors while calculating attention.

Let $\mathbf{v}_i \in \mathbb{R}^H$ and $\mathbf{v}_j \in \mathbb{R}^H$ represent the question-dependent passage vectors of the ith and jth word, i.e., the ith and jth row of $\mathbf{V}$. Tensor-based transformation for multi-factor attention is formulated as follows:

$$\mathbf{f}_{i,j}^m = \mathbf{v}_i \ \mathbf{W}_f^{[1:m]} \ \mathbf{v}_j^\top \quad , \tag{3.12}$$

where $\mathbf{W}_f^{[1:m]} \in \mathbb{R}^{H \times m \times H}$ is a 3-way tensor and m is the number of factors. The output of the tensor product $\mathbf{f}_{i,j}^m \in \mathbb{R}^m$ is a vector where each element $f_{i,j,k}^m$ is a result of the bilinear form defined by each tensor slice $\mathbf{W}_f^{[k]} \in \mathbb{R}^{H \times H}$:

$$f_{i,j,k}^m = \mathbf{v}_i \ \mathbf{W}_f^{[k]} \ \mathbf{v}_j^\top = \sum_{a,b} v_{i,a} W_{f_{a,b}}^{[k]} v_{j,b} \tag{3.13}$$

$\forall i, j \in [1, T]$, the multi-factor attention tensor can be given as $\mathbf{F}^{[1:m]} \in \mathbb{R}^{m \times T \times T}$. For every vector $\mathbf{f}_{i,j}^m$ of $\mathbf{F}^{[1:m]}$, we perform a max pooling operation over all the elements to obtain the resulting attention value:

$$F_{i,j} = \max(\mathbf{f}_{i,j}^m) \quad , \tag{3.14}$$

where $F_{i,j}$ represents the element in the ith row and jth column of $\mathbf{F} \in \mathbb{R}^{T \times T}$. Each row of $\mathbf{F}$ measures how relevant every passage word is with respect to a given question-dependent passage encoding of a word. We apply a row-

wise softmax function on $\mathbf{F}$ to normalize the attention weights, obtaining $\tilde{\mathbf{F}} \in \mathbb{R}^{T \times T}$. Next, an aggregated multi-factor attentive encoding vector is computed corresponding to each question-dependent passage word encoding vector. The aggregated vectors corresponding to all the passage words, $\mathbf{M} \in \mathbb{R}^{T \times H}$, can be given as $\mathbf{M} = \tilde{\mathbf{F}}\ \mathbf{V}$. The aggregated multi-factor attentive encoding vectors are concatenated with the question-dependent passage word encoding vectors to obtain $\tilde{\mathbf{M}} \in \mathbb{R}^{T \times 2H}$. To control the impact of $\tilde{\mathbf{M}}$, we apply a feed-forward neural network-based gating method to obtain $\mathbf{Y} \in \mathbb{R}^{T \times 2H}$. If the tth row of $\tilde{\mathbf{M}}$ is $\tilde{\mathbf{m}}_t$, then the tth row of $\mathbf{Y}$ is:

$$\mathbf{y}_t = \tilde{\mathbf{m}}_t \odot \text{sigmoid}(\tilde{\mathbf{m}}_t \mathbf{W}^g + \mathbf{b}^g) \quad , \tag{3.15}$$

where $\odot$ represents element-wise multiplication. $\mathbf{W}^g \in \mathbb{R}^{2H \times 2H}$ and $\mathbf{b}^g \in \mathbb{R}^{2H}$ are the transformation matrix and bias vector respectively.

We use another pair of stacked BiLSTMs on top of $\mathbf{Y}$ to determine the beginning and ending pointers. Let the hidden unit representations of these two BiLSTMs be $\mathbf{B} \in \mathbb{R}^{T \times H}$ and $\mathbf{E} \in \mathbb{R}^{T \times H}$. To incorporate the dependency of the ending pointer on the beginning pointer, the hidden unit representation of $\mathbf{B}$ is used as input to $\mathbf{E}$.

3.4.6 Question-focused Attentional Pointing

Unlike previous approaches, our proposed model does not predict the answer pointers directly from contextual passage encoding or use another decoder for generating the pointers. We formulate a question representation based on two parts:

- max-attentional question aggregation ($\mathbf{q}_{ma}$)
- question type representation ($\mathbf{q}_f$)

$\mathbf{q}_{ma}$ is formulated by using the similarity matrix $\mathbf{A}$ and the sequence-level question encoding $\mathbf{Q}$. We apply a maxcol operation on $\mathbf{A}$ which forms a row vector whose elements are the maximum of the corresponding columns of $\mathbf{A}$. We define $\mathbf{k} \in \mathbb{R}^U$ as the normalized max-attentional weights:

$$\mathbf{k} = \text{softmax}(\text{maxcol}(\mathbf{A})) \tag{3.16}$$

where softmax is used for normalization. The max-attentional question representation $\mathbf{q}_{ma} \in \mathbb{R}^H$ is:

$$\mathbf{q}_{ma} = \mathbf{k}\ \mathbf{Q} \tag{3.17}$$

Intuitively, $\mathbf{q}_{ma}$ aggregates the most relevant parts of the question with respect to all the words in the passage.

$\mathbf{q}_f$ is the vector concatenation of the representations of the first wh-word and its following word from the sequence-level question encoding $\mathbf{Q}$. The set of wh-words we used is {*what, who, how, when, which, where, why*}. If $\mathbf{q}_{t_{wh}}$ and $\mathbf{q}_{t_{wh}+1}$ represent the first wh-word and its following word (i.e., the t_{wh}th and $(t_{wh}+1)$th rows of $\mathbf{Q}$), $\mathbf{q}_f \in \mathbb{R}^{2H}$ is expressed as:

$$\mathbf{q}_f \ = \ \mathbf{q}_{t_{wh}} \ || \ \mathbf{q}_{t_{wh}+1} \tag{3.18}$$

The final question representation $\tilde{\mathbf{q}} \in \mathbb{R}^H$ is expressed as:

$$\tilde{\mathbf{q}} = \tanh((\mathbf{q}_{ma} \ || \ \mathbf{q}_f)\mathbf{W}_q \ + \ \mathbf{b}_q) \tag{3.19}$$

where $\mathbf{W}_q \in \mathbb{R}^{3H \times H}$ and $\mathbf{b}_q \in \mathbb{R}^H$ are the weight matrix and bias vector respectively. If no wh-word is present in a question, we use the first two sequence-level question word representations for calculating $\tilde{\mathbf{q}}$.

We measure the similarity between $\tilde{\mathbf{q}}$ and the contextual encoding vectors in $\mathbf{B}$ and $\mathbf{E}$ to determine the beginning and ending answer pointers. Corresponding similarity vectors $\mathbf{s}_b \in \mathbb{R}^T$ and $\mathbf{s}_e \in \mathbb{R}^T$ are computed as:

$$\mathbf{s}_b = \tilde{\mathbf{q}}\ \mathbf{B}^\top \quad , \quad \mathbf{s}_e = \tilde{\mathbf{q}}\ \mathbf{E}^\top \tag{3.20}$$

The probability distributions for the beginning pointer b and the ending pointer e for a given passage $\mathcal{P}$ and a question $\mathcal{Q}$ can be given as:

$$\begin{aligned} \Pr(b \mid \mathcal{P}, \mathcal{Q}) &= \text{softmax}(\mathbf{s}_b) \\ \Pr(e \mid \mathcal{P}, \mathcal{Q}, b) &= \text{softmax}(\mathbf{s}_e) \end{aligned} \tag{3.21}$$

The joint probability distribution for obtaining the answer a is given as:

$$\Pr(a \mid \mathcal{P}, \mathcal{Q}) = \Pr(b \mid \mathcal{P}, \mathcal{Q})\ \Pr(e \mid \mathcal{P}, \mathcal{Q}, b) \tag{3.22}$$

To train our model, we minimize the cross entropy loss:

$$\text{loss} = -\sum \log\ \Pr(a \mid \mathcal{P}, \mathcal{Q}) \tag{3.23}$$

summing over all training instances. During prediction, we select the locations in the passage for which the product of $\Pr(b)$ and $\Pr(e)$ is maximum, keeping the constraint $1 \leq b \leq e \leq T$.

Passage: ... The family of a Korean-American missionary believed held in North Korea said Tuesday they are working with U.S. officials to get him returned home. Robert Park told relatives before Christmas that he was trying to sneak into the isolated communist state to bring a message of "Christ's love and forgiveness" to North Korean leader Kim ... **Question**: What is the name of the Korean-American missionary? **Reference Answer**: Robert Park

Table 3.1: Example of a (passage, question, answer)

3.5 Visualization

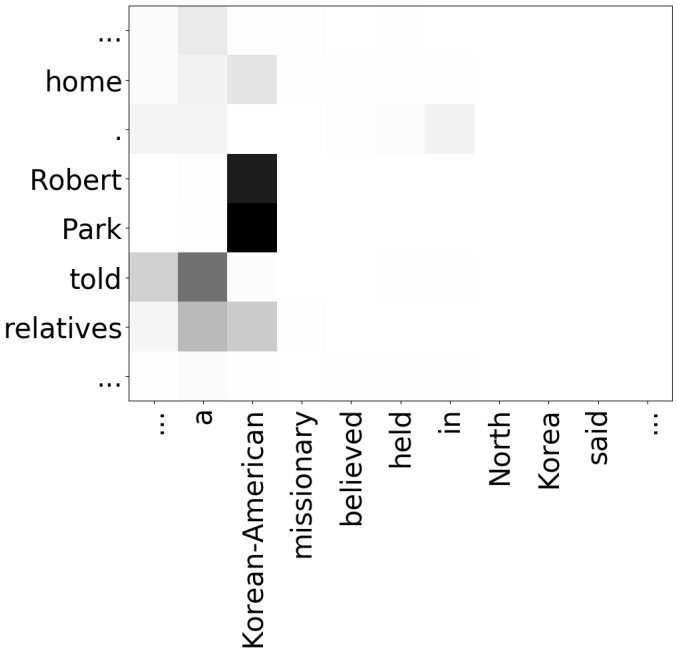


Figure 3.2: Multi-factor attention weights (darker regions signify higher weights).

To understand how the proposed model works, for the example given in Table 3.1, we visualize the normalized multi-factor attention weights $\tilde{\mathbf{F}}$ and the attention weights $\mathbf{k}$ which are used for max-attentional question

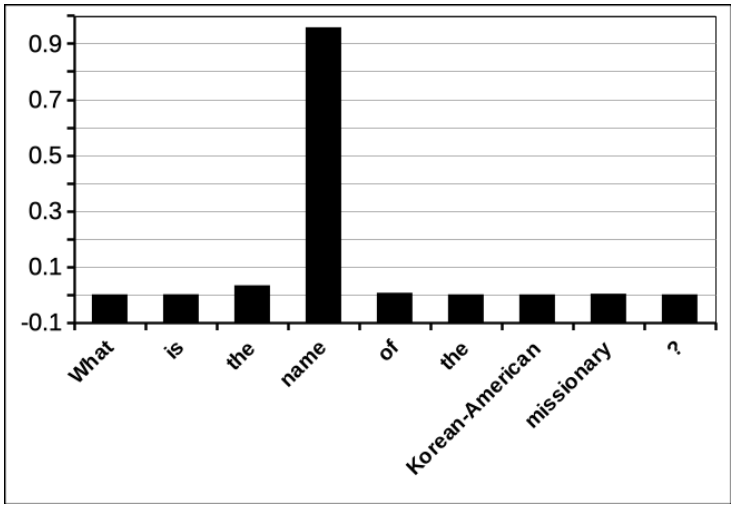


Figure 3.3: Max-attentional weights for question (the origin is set to −0.1 for clarity).

aggregation.

In Figure 3.2, a small portion of $\tilde{\mathbf{F}}$ has been shown, in which the answer words *Robert* and *Park* are both assigned higher weights when paired with the context word *Korean-American*. Due to the use of multi-factor attention, the answer segment pays more attention to the important keyword although it is quite far in the context passage and thus effectively infers the correct answer by capturing the long-range dependency. In Figure 3.3, it is clear that the important question word *name* is getting a higher weight than the other question words. This helps to infer the answer type during prediction, i.e., a person's name in this example.

3.6 Experiments

We evaluated AMANDA on three challenging QA datasets: NewsQA, TriviaQA, and SearchQA. Using the NewsQA development set as a benchmark,

we perform rigorous analysis for better understanding of how our proposed model works.

3.6.1 Datasets

The NewsQA dataset (Trischler et al., 2017) consists of around 100K answerable questions in total. Similar to Trischler et al. (2017) and Weissenborn et al. (2017), we do not consider unanswerable questions in our experiments in this chapter. NewsQA is more challenging compared to the previously released datasets as a significant proportion of questions requires reasoning beyond simple word and context matching. This is due to the fact that the questions in NewsQA were formulated only based on summaries without accessing the main text of the articles. Moreover, NewsQA passages are significantly longer (average length of 616 words) and cover a wider range of topics.

TriviaQA (Joshi et al., 2017) consists of question-answer pairs authored by trivia enthusiasts and independently gathered evidence documents from Wikipedia and Bing Web search. This makes the task more similar to real-life IR-style QA. In total, the dataset consists of over 650K question-answer-evidence triples. Due to the high redundancy in Web search results (around 6 documents per question), each question-answer-evidence triple is treated as a separate sample and evaluation is performed at the document level. However, in Wikipedia, questions are not repeated (each question has 1.8 evidence documents) and evaluation is performed over questions. In addition to distant supervision, TriviaQA also has a verified human-annotated

question-evidence collection. Compared to previous datasets, TriviaQA has more complex compositional questions that require greater multi-sentence reasoning.

SearchQA (Dunn et al., 2017) is also constructed to more closely reflect IR-style QA. They first collected existing question-answer pairs from a Jeopardy archive and augmented them with text snippets retrieved by Google. One difference with TriviaQA is that the evidence passages in SearchQA are Google snippets instead of Wikipedia or Web search documents. This makes reasoning more challenging as the snippets are often very noisy. SearchQA consists of 140,461 question-answer pairs, where each pair has 49.6 snippets on average and each snippet has 37.3 tokens on average.

3.6.2 Experimental Settings

We tokenize the corpora with NLTK[2]. We use the 300-dimension pre-trained word vectors from GloVe (Pennington et al., 2014) and we do not update them during training. The out-of-vocabulary words are initialized with zero vectors. We use 50-dimension character-level embedding vectors. The number of hidden units in all the LSTMs is 150. We use dropout (Srivastava et al., 2014) with probability 0.3 for every learnable layer. For multi-factor attentive encoding, we choose 4 factors (m) based on our experimental findings (refer to Table 3.7). During training, the minibatch size is fixed at 60. We use the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 0.001 and clipnorm of 5. During testing, we enforce the constraint that the ending pointer will always be equal to or greater than the beginning pointer. We use

[2]http://www.nltk.org/

Model	Dev		Test	
	EM	**F1**	**EM**	**F1**
Trischler et al. (2017)				
Match-LSTM	34.4	49.6	34.9	50.0
BARB	36.1	49.6	34.1	48.2
Golub et al. (2017)	-	-	37.1	52.3
Weissenborn et al. (2017)				
Neural BoW Baseline	25.8	37.6	24.1	36.6
FastQA	43.7	56.4	41.9	55.7
FastQAExt	43.7	56.1	42.8	56.1
Weissenborn (2017)	-	-	43.7	56.7
AMANDA	**48.4**	**63.3**	**48.4**	**63.7**

Table 3.2: Results on the NewsQA dataset.

exact match (EM) and F1 scores as the evaluation metrics. We implemented AMANDA in this chapter using Keras[3] and Theano.

3.6.3 Results

Table 3.2 shows that AMANDA outperforms all the state-of-the-art models by a significant margin on the NewsQA dataset. Table 3.3 shows the results on the TriviaQA dataset. In Table 3.3, the model named Classifier based on feature engineering was proposed by Joshi et al. (2017). They also reported the performance of BiDAF (Seo et al., 2017). A memory network-based approach, MEMEN, was recently proposed by Pan et al. (2017). Note that in the Wikipedia domain, we choose the answer which provides the highest maximum joint probability (according to Eq. (3.22)) for any document. Table 3.3 shows that AMANDA achieves state-of-the-art results in both Wikipedia and Web domain on distantly supervised and verified data.

[3]https://keras.io/

Model	Domain	Distant Supervision				Verified			
		Dev		Test		Dev		Test	
		EM	**F1**	**EM**	**F1**	**EM**	**F1**	**EM**	**F1**
‡Random	Wiki	12.72	22.91	12.74	22.35	14.81	23.31	15.41	25.44
‡Classifier		23.42	27.68	22.45	26.52	24.91	29.43	27.23	31.37
‡BiDAF		40.26	45.74	40.32	45.91	47.47	53.70	44.86	50.71
⋆MEMEN		43.16	46.90	-	-	49.28	55.83	-	-
AMANDA		**46.95**	**52.51**	**46.67**	**52.22**	**52.86**	**58.74**	**50.51**	**55.93**
‡Classifier	Web	24.64	29.08	24.00	28.38	27.38	31.91	30.17	34.67
‡BiDAF		41.08	47.40	40.74	47.05	51.38	55.47	49.54	55.80
⋆MEMEN		44.25	48.34	-	-	53.27	57.64	-	-
AMANDA		**46.68**	**53.27**	**46.58**	**53.13**	**60.31**	**64.90**	**55.14**	**62.88**

Table 3.3: Results on the TriviaQA dataset. ‡Joshi et al. (2017), ⋆Pan et al. (2017)

Model	Set	Unigram Accuracy	N-gram F1
Dunn et al. (2017)			
TF-IDF Max	Dev	13.0	-
	Test	12.7	-
ASR	Dev	43.9	24.2
	Test	41.3	22.8
AMANDA	Dev	**48.6**	**57.7**
	Test	**46.8**	**56.6**

Table 3.4: Results on the SearchQA dataset.

Results on the SearchQA dataset are shown in Table 3.4. In addition to a TF-IDF approach, Dunn et al. (2017) modified and reported the performance of attention sum reader (ASR) which was originally proposed by Kadlec et al. (2016). We consider a maximum of 150 words surrounding the answer from the concatenated ranked list of snippets as a passage to more quickly train the model and to reduce the amount of noisy information. During prediction, we choose the first 200 words (about 5 snippets) from the concatenated ranked list of snippets as an evidence passage. These are chosen based on performance on the development set. Based on question patterns, question types are always represented by the first two sequence-level representations of question words. To make the results comparable, we also report accuracy for single-word-answer (unigram) questions and F1 score for multi-word-answer (n-gram) questions. AMANDA outperforms both systems, especially for multi-word-answer questions by a huge margin.

3.6.4 Effectiveness of the Model Components

Table 3.5 shows that AMANDA performs better than any of the ablated models which include the ablation of multi-factor attentive encoding, max-attentional question aggregation ($\mathbf{q}_{ma}$), and question type representation ($\mathbf{q}_f$). We also perform statistical significance test using paired t-test and bootstrap resampling. Performance of AMANDA (both in terms of EM and F1) is significantly better ($p < 0.01$) than the ablated models.

Model	**EM**	**F1**
minus multi factor attn.	46.4	61.2
minus $\mathbf{q}_{ma}$ and $\mathbf{q}_f$	46.2	60.5
minus $\mathbf{q}_{ma}$	46.6	61.3
minus $\mathbf{q}_f$	46.8	61.8
AMANDA	**48.4**	**63.3**

Table 3.5: Ablation of proposed components on the NewsQA development set.

Model	**EM**	**F1**
minus char embedding	47.5	61.4
minus question-dependent passage enc.	32.1	45.0
minus 2nd LSTM during prediction	46.5	61.6
AMANDA	**48.4**	**63.3**

Table 3.6: Ablation of other components on the NewsQA development set

One of the key contributions of this work is multi-factor attentive encoding which aggregates information from the relevant passage words by using a tensor-based attention mechanism. The use of multiple factors helps to fine-tune answer inference by synthesizing information distributed across multiple sentences. The number of factors is the granularity to which the model is allowed to refine the evidence. The effect of multi-factor attentive encoding is

illustrated by the following example taken from the NewsQA development set:

What will allow storage on remote servers?

...The ***iCloud service*** *will now be integrated into the iOS 5 operating system. It will work with apps and allow content to be stored on remote servers instead of the users' iPod, iPhone or other device...*

When multi-factor attentive encoding is ablated, the model could not figure out the cross-sentence co-reference and wrongly predicted the answer as *apps*. On the contrary, with multi-factor attentive encoding, AMANDA could correctly infer the answer as *iCloud service*. We also performed an analysis over the instances in the NewsQA development set where AMANDA could correctly predict the answer but it predicts the answer incorrectly when multi-factor attentive encoding is not considered. We manually analyzed 50 such randomly sampled instances and found that 80% of them indeed required reasoning over multiple facts distributed across sentences.

Another contribution of this work is to include the question focus during prediction. It is performed by adding two components: $\mathbf{q}_{ma}$ (max-attentional question aggregation) and $\mathbf{q}_f$ (question type representation). $\mathbf{q}_{ma}$ and $\mathbf{q}_f$ implicitly infer the answer type during prediction by focusing on the important question words. Impact of the question focus components is illustrated by the following example taken from the NewsQA development set:

who speaks on Holocaust remembrance day?

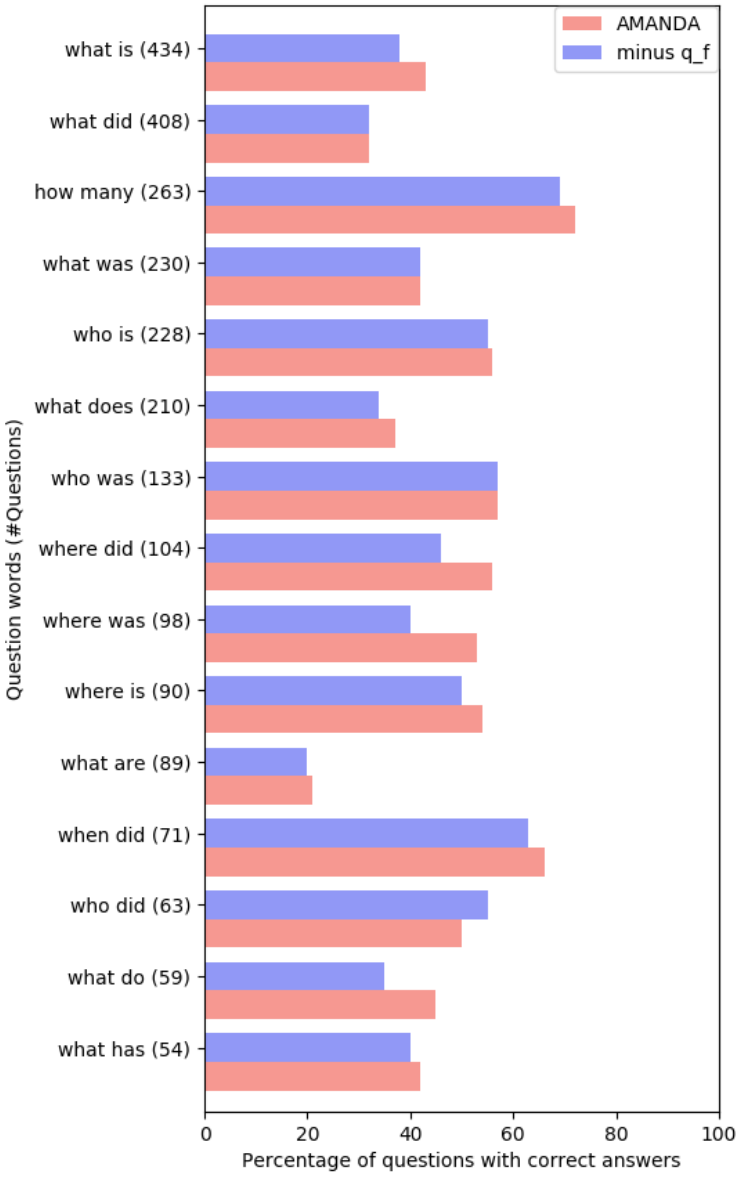


Figure 3.4: Performance analysis of the $\mathbf{q}_f$ component across different question types. Red bars represent the performance of AMANDA and the blue bars represent the performance when $\mathbf{q}_f$ is not considered. AMANDA performs better on the exact match (EM) score across almost all the question types.

*... Israel's vice prime minister **Silvan Shalom** said Tuesday "Israel can never ... people just 65 years ago" ... He was speaking as Israel observes its Holocaust memorial day, remembering the roughly...*

Without the $\mathbf{q}_{ma}$ and $\mathbf{q}_f$ components, the answer was wrongly predicted as *Israel*, whereas with $\mathbf{q}_{ma}$ and $\mathbf{q}_f$, AMANDA could correctly infer the answer type (i.e., a person's name) and predict *Silvan Shalom* as the answer.

We also perform a comparative analysis between AMANDA and *minus* $\mathbf{q}_f$ across the most frequent question types, as depicted in Figure 3.4. We show that AMANDA performs better on almost all the different question types.

Ablation studies of other components such as character embedding, question-dependent passage encoding, and the second LSTM during prediction are given in Table 3.6. When the second LSTM (**E**) is ablated, a feed-forward layer is used instead. Table 3.6 shows that question-dependent passage encoding has the highest impact on performance.

3.6.5 Variation on the number of factors (m) and $\mathbf{q}_{ma}$

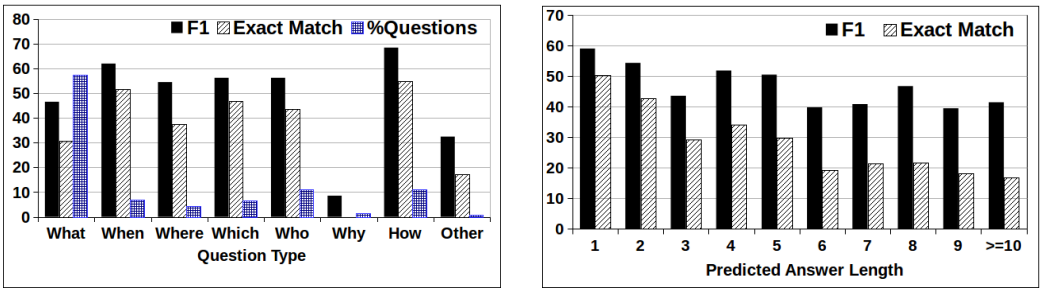
Table 3.7 shows the performance of AMANDA for different values of m. We use 4 factors for all the experiments as it gives the highest F1 score. Note that $m = 1$ is equivalent to standard bilinear attention.

Value of m	**1**	**2**	**3**	**4**	**5**
EM	45.8	47.4	**48.7**	48.4	48.0
F1	61.2	61.9	62.9	**63.3**	62.5

Table 3.7: Variation of m on the NewsQA development set.

Aggregation	EM	F1
Mean	46.6	61.3
Sum	47.9	62.2
Max (AMANDA)	**48.4**	**63.3**

Table 3.8: Variation of question aggregation formulation on the NewsQA development set.



(a) (b)

Figure 3.5: (a) Results for different question types. (b) Results for different predicted answer lengths.

Table 3.8 shows the variation of question aggregation formulation. For mean aggregation, the attentional weight vector $\mathbf{k}$ is formulated by applying column-wise averaging on the similarity matrix $\mathbf{A}$. Intuitively, it is giving equal priority to all the passage words to determine a particular question word attention. Similarly, in the case of sum aggregation, we apply a column-wise sum operation. Table 3.8 shows that the best performance is obtained when $\mathbf{q}_{ma}$ is obtained with a column-wise maximum operation on $\mathbf{A}$. Effectively, it is helping to give higher weights to the more important question words based on the most relevant passage words.

3.6.6 Quantitative Error Analysis

We analyzed the performance of AMANDA across different question types and different predicted answer lengths. Figure 3.5(a) shows that it performs poorly on *why* and *other* questions whose answers are usually longer. Figure 3.5(b) supports this fact as well. When the predicted answer length increases, both F1 and EM start to degrade. The gap between F1 and EM also increases for longer answers. This is because for longer answers, the model is not able to decide the exact boundaries (low EM score) but manages to predict some correct words which partially overlap with the reference answer (relatively higher F1 score).

3.6.7 Qualitative Error Analysis

On the NewsQA development set, AMANDA predicted completely wrong answers on 25.1% of the questions. We randomly picked 50 such questions for analysis. The observed types of errors are given in Table 3.9 with examples. 42% of the errors are due to answer ambiguities, i.e., no unique answer is present. 22% of the errors are due to mismatch between question and context words. 10% of the errors are due to the need for highly complex inference. 6% of the errors occur due to paraphrasing, i.e., the question is posed with different words that do not appear in the passage context. The remaining 20% of the errors are due to insufficient evidence, incorrect tokenization, wrong co-reference resolution, etc.

Answer ambiguity (42%) Q: What happens to the power supply? ... customers possible." The outages were due mostly to power **lines downed** by Saturday's hurricane-force winds, which knocked over trees and utility poles. At ...
Context mismatch (22%) Q: Who was Kandi Burrus's fiance? Kandi Burruss, the newest cast member of the reality show "The Real Housewives of Atlanta" ... fiance, who died ... fiance, 34-year-old **Ashley "A.J." Jewell**, also...
Complex inference (10%) Q: When did the Delta Queen first serve? ... the Delta Queen steamboat, a floating National ... scheduled voyage this week ... The Delta Queen will go ... Supporters of the boat, which has roamed the nation's waterways since **1927** and helped the Navy ...
Paraphrasing issues (6%) Q: What was Ralph Lauren's first job? Ralph Lauren has ... Japan. For four ... than the former **tie salesman** from the Bronx. "Those ties ... Lauren originally named his company Polo because ...

Table 3.9: Examples of different error types and their percentages. Ground truth answers are bold-faced and predicted answers are underlined.

3.7 Further Advances

In this section, we discuss the recent advances in deep neural network-based reading comprehension systems. These systems were mostly proposed after our work was published.

Tay et al. (2018a) proposed a Dilated Compositional Unit (DCU)-based encoder for fast and expressive sequence encoding for RC-based QA. The proposed architecture utilizes DCU within a Bi-Attentive framework for both

multiple choice and span prediction RC tasks. DCU achieved 49.4 Unigram Accuracy and 59.5 N-gram F1 scores on the SearchQA dataset. Lin et al. (2018) employed a passage selector to filter out noisy passages and a passage reader to extract the correct answer from those denoised passages. They reported 58.8 Exact Match and 64.5 F1 scores on the SearchQA dataset. Tay et al. (2018b) proposed a densely connected neural architecture (DECAPROP) for reading comprehension-based QA. DECAPROP densely connects all pairwise layers of the network to model relationships between a passage and a question across all hierarchical levels. Additionally, the dense connectors are learned by using a Bidirectional Attention Connector (BAC). DECAPROP achieved 53.1 Exact Match and 66.3 F1 scores on the NewsQA dataset. It achieved 62.2 Unigram Accuracy and 70.8 N-gram F1 scores on the SearchQA dataset. Clark and Gardner (2018) improved upon the popular BiDAF model (Seo et al., 2017) by sampling multiple passages from the documents during training and using a shared normalization training objective that encourages the model to produce globally correct output. Clark and Gardner (2018) reported a 71.3 F1 score on the web portion of the TriviaQA dataset.

Another important recent advancement is the use of contextualized word embeddings (Peters et al., 2018; Devlin et al., 2019). In traditional word embeddings, each word is mapped to a unique single vector. In contrast, contextualized word embeddings assign each word a vector as a function of the entire input sequence. It has been shown that the contextualized word embeddings can better model complex characteristics of word use (e.g., syntax and semantics) and how these uses vary across linguistic contexts (i.e., polysemy).

The use of contextualized word embeddings has shown consistent improvements across many NLP tasks including QA. Peters et al. (2018) proposed Elmo where the contextualized word embeddings are learned functions of the internal states of a deep bidirectional LSTM-based language model, which is pre-trained on a large text corpus. Elmo embeddings are typically used in conjunction with traditional word type embeddings and character embeddings. More recently, Devlin et al. (2019) proposed that these contextualized word embeddings can not only be used as features of word representations in a task-specific neural architecture, but the deep language models can also be directly fine-tuned with minimal modifications to perform downstream tasks. Devlin et al. (2019) randomly mask out some words at the input layer, stack bidirectional transformer layers, and predict these masked words at the top layer. While BERT neglects dependency between the masked positions and suffers from a pretrain-finetune discrepancy, Yang et al. (2019) proposed XLNet using an autoregressive formulation that enables learning bidirectional contexts by maximizing the expected likelihood over all permutations of the factorization order. XLNet further improves over BERT across many NLP tasks. Typically, the contextualized word embeddings are high dimensional and computationally intensive. Hence, it is challenging to apply them to tasks where the input sequences are very long (e.g., passages in TriviaQA).

3.8 Summary

In this chapter, we proposed an end-to-end neural model for the machine reading comprehension task. Specifically, we proposed a question-focused

multi-factor attention network (AMANDA), which learns to aggregate meaningful evidence from multiple sentences and to focus on the important words in a question for extracting an answer span from the passage with a suitable answer type. AMANDA achieves the best performance on NewsQA, TriviaQA, and SearchQA datasets, outperforming prior models by significant margins. Ablation results show the importance of the proposed components.

In this chapter, we assume that there always exists a valid answer in the associated passage for a given question. In the next chapter, we extend AMANDA and several other RC models for nil-aware answer extraction, where the associated passage might not always contain a valid answer.

Chapter 4

A Nil-Aware Answer Extraction Framework for Question Answering

Prior approaches for reading comprehension (RC) based question answering (QA) suffer from an impractical assumption that every question has a valid answer in the associated passage. A practical QA system must possess the ability to determine whether a valid answer exists in a given text passage. In this chapter, we focus on developing QA systems that can extract an answer for a question if and only if the associated passage contains an answer. If the associated passage does not contain any valid answer, the QA system should correctly return *Nil*. We propose a nil-aware answer span extraction framework that is capable of returning *Nil* or a text span from the associated passage as an answer in a single step. We show that our proposed framework

can be easily integrated with several recently proposed QA models developed for reading comprehension and can be trained in an end-to-end fashion. Our proposed nil-aware answer extraction neural network decomposes pieces of evidence into relevant and irrelevant parts and then combines them to infer the existence of any answer. Experiments on the NewsQA dataset show that the integration of our proposed framework significantly outperforms several strong baseline systems that use pipeline or threshold-based approaches.

The rest of this chapter is organized as follows. Section 4.1 provides the background of this work. Section 4.2 describes our proposed framework in detail and how it can be integrated with several QA models. Section 4.3 provides the descriptions of several baseline systems used for comparison, followed by the experiments and analysis in Section 4.4. In Section 4.5, we provide an overview of the further advances in nil-aware machine reading comprehension-based QA. Finally, we summarize the chapter in Section 4.6.

4.1 Background

In recent years, research on question answering has witnessed substantial progress with rapid advances in neural network architectures. While several neural models for machine reading comprehension have been proposed (Wang and Jiang, 2017; Seo et al., 2017; Yang et al., 2017; Xiong et al., 2017; Weissenborn et al., 2017; Wang et al., 2017; Shen et al., 2017b; Chen et al., 2017; Kundu and Ng, 2018a), none of the models considered nil questions, although it is crucial for a practical QA system to be able to determine whether a text passage contains a valid answer for a question. In this work, we focus on

developing QA systems that extract an answer for a question if and only if the associated passage contains a valid answer. Otherwise, they are expected to return *Nil* as an answer.

We propose a nil-aware answer extraction framework which returns *Nil* or a span of text as answer, when integrated to end-to-end neural RC-based QA models. Our proposed framework is based on evidence decomposition-aggregation where the evidence vectors derived by a higher level encoding layer are first decomposed into relevant and irrelevant components and later aggregated to infer the existence of a valid answer. In addition, we develop several baseline models with pipeline and threshold-based approaches. In a pipeline model, the detection of nil questions is carried out separately before answer span extraction. In a threshold-based model, the answer span extraction model is entirely trained on questions that have valid answers, and *Nil* is returned based on a confidence threshold.

The contributions of this work reported in this chapter are as follows:

- We propose a nil-aware answer span extraction framework to return *Nil* or an exact answer span to a question, in a single step, depending on the existence of a valid answer.

- Our framework can be readily integrated with many recently proposed neural machine comprehension models. In this work, we extended four machine comprehension models, namely BiDAF (Seo et al., 2017), R-Net (Wang et al., 2017), DrQA (Chen et al., 2017), and AMANDA (Kundu and Ng, 2018a), with our proposed framework and show that they achieve significantly better results compared to the corresponding

pipeline and threshold-based models on the NewsQA dataset.

4.2 Proposed Framework

Given a passage and a question, we propose models that can extract an answer if and only if the passage contains a valid answer. When the passage does not contain any valid answer, the models return *Nil* as the answer. Similar to the work in Chapter 3, the valid answer is denoted as two pointers in the passage, representing the start and end tokens of the answer span. We first describe our proposed evidence decomposition-aggregation framework for nil-aware answer extraction. Then, we provide a detailed description of how we extend our previously proposed model AMANDA (Kundu and Ng, 2018a) to NAMANDA[1] (nil-aware AMANDA). We also provide brief descriptions about how we integrate our proposed framework to the other three models, namely BiDAF, DrQA, and R-Net.

4.2.1 Nil-Aware Answer Extraction

Decomposition of lexical semantics over sentences has been successfully used in the past for sentence similarity learning (Wang et al., 2016b). Most of the recently proposed machine reading comprehension models can be generalized based on a common pattern observed in their network architecture. They have a question-passage joint encoding layer (also known as question-aware passage encoding layer) followed by an evidence encoding layer. In this work, we decompose the evidence vectors for each passage word obtained from

[1]Our source code is released at https://github.com/nusnlp/namanda

the evidence encoding layer with respect to question-passage joint encoding vectors to derive semantically relevant and irrelevant components. We decompose the evidence vectors for each passage word, because passage vectors can be partially supported by the corresponding question-passage joint encoding vectors, and based on the level of support, it either increases or decreases the chance of finding a valid answer. When we aggregate the orthogonally decomposed evidence vectors, it combines both the supportive and unsupportive pieces of evidence for a particular passage word. To obtain the most impactful portions, we perform a max-pooling operation over all the aggregated vectors. The resulting vector is denoted as the Nil vector. Typically, all the considered RC-based QA models compute two scores for each passage word to derive the start and end pointers of a valid answer. Additionally, we compute a nil pointer score from the Nil vector for the nil answer. We jointly normalize the start and end pointer scores with the nil pointer score. As the training set[2] contains both nil questions (with no valid answers) and non-nil questions (with valid answers), the model automatically learns when to pool unsupportive (for nil questions) and supportive (for non-nil questions) portions to construct the Nil vector. In this way, the model is able to induce a strong bias towards the nil pointer when there is no answer present due to the dominance of unsupportive components in the nil vector.

The proposed method in Wang et al. (2016b) was developed for sentence similarity learning tasks, such as answer sentence selection. They decompose an answer sentence with respect to a question and vice versa. The decomposed vectors are then aggregated to obtain a single vector which is used

[2]Refers to the training set of the NewsQA dataset

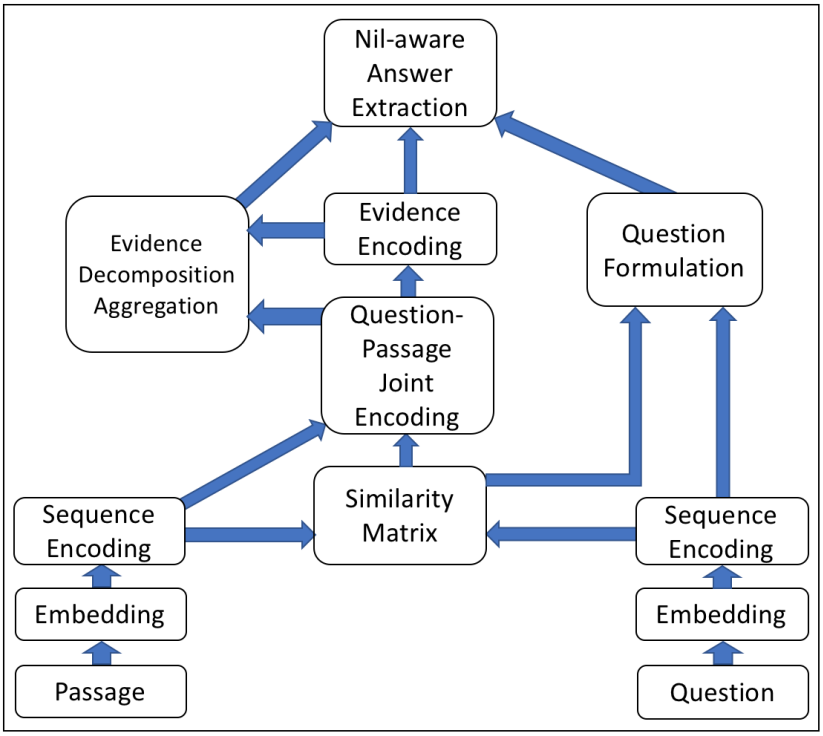


Figure 4.1: Overview of the architecture of Nil-aware AMANDA (NAMANDA). Except *Evidence Decomposition Aggregation* and *Nil-aware Answer Extraction*, the remaining components are the same as AMANDA.

to derive the similarity score. Although our proposed method (developed for the more complex task of answer span extraction) is inspired from the idea of lexical decomposition and composition, one major difference is that we decompose the evidence vectors with respect to question-passage joint encoding vectors. Another important advance is how it is adapted to return nil or a span of text from the passage in a single step.

4.2.2 Nil-Aware AMANDA

The architecture of Nil-aware AMANDA (NAMANDA) is given in Figure 4.1. Initial layers such as word embedding, similarity matrix, question-dependent

passage encoding, and question formulation of AMANDA remains the same as described in Section 3.4. Additionally, we extend AMANDA with an evidence decomposition-aggregation component followed by a nil-aware pointing mechanism. They are discussed as follows.

Evidence Decomposition-Aggregation

First, multi-factor self-attentive encoding is performed to accumulate evidence from the entire passage. The use of multiple factors while calculating self attention helps to obtain meaningful information from a long context with fine-grained inference. If m represents the number of factors, multi-factor attention $\mathbf{F}^{[1:m]} \in \mathbb{R}^{T \times m \times T}$ is formulated as:

$$\mathbf{F}^{[1:m]} \;=\; \mathbf{V}\; \mathbf{W}_f^{[1:m]}\; \mathbf{V}^\top \quad , \tag{4.1}$$

where $\mathbf{W}_f^{[1:m]} \in \mathbb{R}^{H \times m \times H}$ is a 3-way tensor. T denotes the number of tokens in the passage, and $\mathbf{V}$ represents the question-passage joint encoding vectors. Now, to refine the evidence, a max-pooling operation is performed on $\mathbf{F}^{[1:m]}$ over the factor axis resulting in the self-attention matrix $\mathbf{F} \in \mathbb{R}^{T \times T}$. We normalize $\mathbf{F}$ by applying a row-wise softmax function, obtaining $\tilde{\mathbf{F}} \in \mathbb{R}^{T \times T}$. Now the self-attentive encoding $\mathbf{M} \in \mathbb{R}^{T \times H}$ can be given as $\mathbf{M} = \tilde{\mathbf{F}}\; \mathbf{V}$. The self-attentive encoding vectors are then concatenated with the question-dependent passage word encoding vectors and a feed-forward neural network-based gating is applied to control the overall impact, resulting in $\mathbf{Y} \in \mathbb{R}^{T \times 2H}$.

Then we decompose the evidence vector for every passage word with orthogonal decomposition. The primary motivation behind decomposing the

evidence vectors is to focus on both the semantically relevant and irrelevant parts to determine whether a valid answer is present in the associated passage or not. Each row of $\mathbf{Y}$, $\mathbf{y}_t \in \mathbb{R}^{2H}$, is decomposed into its parallel and perpendicular components with respect to the corresponding question-passage joint encoding ($\mathbf{S}$) vector, $\mathbf{s}_t \in \mathbb{R}^{2H}$. The parallel components represent the relevant parts of the accumulated evidence while the orthogonal components represent the irrelevant counterparts. We use an orthogonal decomposition function to decompose the evidence vectors in the geometric space. If the parallel component of $\mathbf{y}_t$ is represented as $\mathbf{y}_t^{=} \in \mathbb{R}^{2H}$ and the perpendicular component is represented as $\mathbf{y}_t^{\perp} \in \mathbb{R}^{2H}$, then:

$$\mathbf{y}_t^{=} \;=\; \frac{\mathbf{y}_t\ \mathbf{s}_t^{\top}}{\mathbf{s}_t\ \mathbf{s}_t^{\top}}\ \mathbf{s}_t \tag{4.2}$$

$$\mathbf{y}_t^{\perp} \;=\; \mathbf{y}_t \;-\; \mathbf{y}_t^{=} \tag{4.3}$$

Similarly, we derive the parallel and orthogonal vectors for all the passage words. We denote parallel components with $\mathbf{Y}^{=} \in \mathbb{R}^{T\times 2H}$ and perpendicular components with $\mathbf{Y}^{\perp} \in \mathbb{R}^{T\times 2H}$.

The aim of the aggregation step is to extract features from both the parallel component matrix $\mathbf{Y}^{=}$ and the perpendicular component matrix $\mathbf{Y}^{\perp}$. In the aggregation step, the parallel and orthogonal components are fed to a linear layer. $\mathbf{Y}^{a} \in \mathbb{R}^{T\times H}$ denotes the output of the linear layer and $\mathbf{y}_t^{a} \in \mathbb{R}^{H}$ is its tth row:

$$\mathbf{y}_t^{a} \;=\; \tanh(\mathbf{y}_t^{=}\mathbf{W}_a \;+\; \mathbf{y}_t^{\perp}\mathbf{W}_a \;+\; \mathbf{b}_a)\ \ , \tag{4.4}$$

where $\mathbf{W}_a \in \mathbb{R}^{2H\times H}$ and $\mathbf{b}_a \in \mathbb{R}^{H}$ are the weight matrix and bias vector

respectively. Then we apply a max-pooling operation over all the words to obtain the *Nil* vector representation denoted as $\hat{\mathbf{n}}$. Now we derive the score for the *Nil* pointer which will be shared for normalizing the beginning and ending pointers later. The *Nil* pointer score is given as:

$$n_s \;=\; \hat{\mathbf{n}}\mathbf{w}_n^\top \quad , \tag{4.5}$$

where $\mathbf{w}_n \in \mathbb{R}^H$ is a learnable weight vector. Through the computation of the *Nil* pointer score, the model learns to assign a higher score to the nil pointer when the perpendicular components are more significant compared to the parallel ones, i.e., when there is no valid answer present in the associated passage. Similarly, when there is a valid answer present, it learns to assign a lower score to the nil pointer.

Nil-Aware Pointing

Two stacked BiLSTMs are used on top of $\mathbf{Y}$ to determine the beginning and ending pointers. Let the hidden unit representations of these two BiLSTMs be $\mathbf{B} \in \mathbb{R}^{T\times H}$ and $\mathbf{E} \in \mathbb{R}^{T\times H}$. We measure the similarity scores between the previously derived question vector $\tilde{\mathbf{q}}$ and the contextual encoding vectors in $\mathbf{B}$ and $\mathbf{E}$. If $\mathbf{s}_b \in \mathbb{R}^T$ and $\mathbf{s}_e \in \mathbb{R}^T$ are the scores for the beginning and ending pointers, then

$$\mathbf{s}_b = \tilde{\mathbf{q}}\;\mathbf{B}^\top \quad , \quad \mathbf{s}_e = \tilde{\mathbf{q}}\;\mathbf{E}^\top \tag{4.6}$$

We prepend the *nil* score n_s to $\mathbf{s}_b$ and $\mathbf{s}_e$ for shared normalization. The

updated scores $\hat{\mathbf{s}}_b \in \mathbb{R}^{T+1}$ and $\hat{\mathbf{s}}_e \in \mathbb{R}^{T+1}$ can be represented as:

$$\hat{\mathbf{s}}_b \;=\; [n_s,\ \mathbf{s}_b] \;\;,\;\; \hat{\mathbf{s}}_e \;=\; [n_s,\ \mathbf{s}_e] \tag{4.7}$$

The beginning and ending pointer probability distributions for a given passage $\mathcal{P}$ and a question $\mathcal{Q}$ is given as:

$$\begin{aligned} \Pr(b \mid \mathcal{P}, \mathcal{Q}) &= \text{softmax}(\hat{\mathbf{s}}_b) \\ \Pr(e \mid \mathcal{P}, \mathcal{Q}) &= \text{softmax}(\hat{\mathbf{s}}_e) \end{aligned} \tag{4.8}$$

The joint probability distribution for answer a is given as:

$$\Pr(a \mid \mathcal{P}, \mathcal{Q}) = \Pr(b \mid \mathcal{P}, \mathcal{Q})\ \Pr(e \mid \mathcal{P}, \mathcal{Q}) \tag{4.9}$$

For training, we minimize the cross entropy loss summing over all training instances. During prediction, we select the locations in the passage for which the product of $\Pr(b)$ and $\Pr(e)$ is maximum, where $1 \leq b \leq e \leq T+1$. If the value of b is 1, we assign the answer as *Nil*.

4.2.3 Nil-Aware DrQA

We extend DrQA (Chen et al., 2017) to NDrQA by integrating our proposed nil-aware answer extraction framework. In DrQA, the embeddings of passage tokens consist of pre-trained word vectors from GloVe, several syntactic features, and passage-question joint embedding (aligned question embedding). The syntactic features include exact match of passage words with question

in surface, lowercase, and lemma form. They also used part-of-speech tags, named entity tags, and term frequency values for each passage word. Subsequently, a stack of BiLSTMs (3 layers) is used for encoding. The outputs of the stacked BiLSTMs are used as evidence vectors to help extract the answer span. We decompose those stacked BiLSTM output vectors with respect to the passage embedding and generate the *nil* pointer score as given in Equations (4.2–4.5). The question vector formulation in DrQA is performed by applying a stack of BiLSTMs on question embeddings and combining the resulting hidden units into one single vector. The question vector $\mathbf{q}_D \in \mathbb{R}^H$ can be represented as:

$$\mathbf{q}_D \;=\; \sum_t \gamma_t \; \mathbf{q}_t \quad , \tag{4.10}$$

where $\mathbf{q}_t \in \mathbb{R}^H$ represents the hidden units of any tth question word. γ_t represents the importance of each question word:

$$\gamma_t \;=\; \frac{\exp(\mathbf{q}_t \; \mathbf{w}^\top)}{\sum_{t'} \exp(\mathbf{q}_{t'} \; \mathbf{w}^\top)} \quad , \tag{4.11}$$

where $\mathbf{w} \in \mathbb{R}^H$ is a weight vector to learn. Chen et al. (2017) used a bi-linear term to compute the scores for start and end pointers of the answer. If the evidence vector encodings are represented as $\mathbf{E}_D \in \mathbb{R}^{T \times H}$, the start and end pointer scores are given as:

$$\mathbf{s}_b^D \;=\; \mathbf{q}_D \; \mathbf{W}_b \; \mathbf{E}_D^\top \quad , \quad \mathbf{s}_e^D \;=\; \mathbf{q}_D \; \mathbf{W}_e \; \mathbf{E}_D^\top \quad , \tag{4.12}$$

where $\mathbf{W}_b \in \mathbb{R}^{H \times H}$ and $\mathbf{W}_e \in \mathbb{R}^{H \times H}$ are trainable matrices. The nil-aware pointing mechanism is the same as discussed previously in Equations (4.7–

4.9).

4.2.4 Nil-Aware R-Net

In R-Net (Wang et al., 2017), after embedding and encoding of the passage and question words, a gated recurrent network is used to obtain the question-passage joint representation. The gated attention-based recurrent neural network contains an additional gate to determine the importance of information in the passage regarding a question. Different from the gates in traditional RNNs such as LSTM or GRU, the additional gate is based on the current passage word and its attention-pooling vector of the question, which focuses on the relation between the question and current passage word. The gate effectively models the phenomenon that only parts of the passage are relevant to the question in reading comprehension-based QA. Subsequently, a self-matching attentive encoding is used to accumulate evidence from the entire passage. In the output layer, an answer recurrent pointer network is used to predict the boundary of an answer span.

To extend R-Net to nil-aware R-Net (NR-Net), we decompose the output vectors of the self-matching layer with respect to the question-passage joint encoding vectors, and then aggregate them to obtain the *nil* pointer score as illustrated in Equations (4.2–4.5). In the output layer, we combine the *nil* pointer score to the beginning and ending pointer unnormalized scores, and jointly normalize them using softmax function as given in Equations (4.7–4.8). The output layer of NR-Net is slightly different from the other considered models in terms of using a recurrent pointer network over using

two simple pointers for computing the start and end pointers of the answer. For further details, we refer the readers to the R-Net paper (Wang et al., 2017).

4.2.5 Nil-Aware BiDAF

In BiDAF (Seo et al., 2017), after embedding and encoding of the passage and question words, an attention flow layer is used to jointly encode the passage and question. The attention flow layer is responsible for linking and fusing information from the passage and the question words. In the attention flow layer, the attention vector at each time step, along with the encodings from previous layers, is allowed to flow through to the subsequent layer. This reduces the information loss caused by early summarization while computing the joint representation of the passage and the question. Then, a modeling layer is used to capture the interaction among the question-aware passage vectors. Seo et al. (2017) used two layers of bi-directional LSTM to construct the modeling layer. The output of the modeling layer serves as evidence to help extract the answer span in the output layer.

To extend the BiDAF model to nil-aware BiDAF (NBiDAF), we decompose the output of the modeling layer with respect to the question-passage joint encoding, and then aggregate them to derive the *nil* pointer score (similar to Equations 4.2–4.5). Similar to the other nil-aware models, we prepend the *nil* pointer score to the start and end pointer unnormalized scores derived in the output layer, and jointly normalize them.

4.3 Baseline Models

For comparison, we propose two types of baseline approaches for nil-aware answer extraction.

4.3.1 Pipeline Approach

Here, two models are used in a pipeline:

- **Nil detector**: Given a pair of passage and question, a nil detector model determines whether a valid answer is present in the passage.
- **Answer span extractor**: If the nil detector model predicts the presence of a valid answer, the answer span extractor then extracts the answer.

For nil detection, we developed a logistic regression (LR) model with manually defined features and four neural models. For the LR model, we extract four different features that capture the similarity between a passage and a question. Let $\mathcal{P}$ be the passage and $\mathcal{Q}$ be the question (consisting of U' tokens excluding stop words). If $f(\mathcal{P},\ \mathcal{Q}_i)$ is the frequency of the ith question word in passage $\mathcal{P}$, then the first feature η_1 is defined as:

$$\eta_1 = \sum_{i=1}^{U'} \log(1 + f(\mathcal{P}, \mathcal{Q}_i)) \tag{4.13}$$

The second feature η_2 is the same as η_1, except that the lemma form is considered for both passage and question tokens instead of the surface form. Additionally, we include word overlap count features in both surface and

lemma forms. Let c_s (c_l) be the number of tokens appearing in both $\mathcal{P}$ and $\mathcal{Q}$ in surface (lemma) form. Features η_3 and η_4 are defined as:

$$\eta_3 = c_s \ / \ U' \ , \ \eta_4 = c_l \ / \ U' \tag{4.14}$$

We also developed several advanced neural network architectures for nil detection. In neural architectures, we do not use any syntactic features. After embedding, we apply sequence-level encoding with either BiLSTM or Convolutional Neural Network (CNN). For CNN, we use equal numbers of unigram, bigram, and trigram filters and the outputs are concatenated to obtain the final encoding. Next, we apply either global max-pooling (MP) or attentive pooling (AP) over all the sequence vectors to obtain an aggregated vector representation. Let the sequence encoding of a passage be $\mathbf{P}^{nd} \in \mathbb{R}^{T \times H}$, and $\mathbf{p}_t^{nd}$ be the tth row of $\mathbf{P}^{nd}$. The aggregated vector $\tilde{\mathbf{p}}_{nd} \in \mathbb{R}^H$ for AP can be obtained as:

$$a_t^{nd} \ \propto \ \exp(\mathbf{p}_t^{nd} \mathbf{w}^\top) \tag{4.15}$$

$$\tilde{\mathbf{p}}_{nd} \ = \ \mathbf{a}^{nd} \mathbf{P}^{nd} \ , \tag{4.16}$$

where $\mathbf{w} \in \mathbb{R}^H$ is a learnable vector. Similarly, we derive the aggregated question vector $\tilde{\mathbf{q}}_{nd}$. For nil detection, we compute the similarity score (s_{nd}) between the aggregated vectors:

$$s_{nd} = \text{sigmoid}(\tilde{\mathbf{p}}_{nd} \ \tilde{\mathbf{q}}_{nd}^\top) \tag{4.17}$$

We use binary cross-entropy for training these models.

We experimented with four state-of-the-art answer span extractor models namely BiDAF (Seo et al., 2017), R-Net (Wang et al., 2017), DrQA (Chen et al., 2017), and AMANDA (Kundu and Ng, 2018a). Note that the answer extraction models are trained entirely on passage-question pairs which always have valid answers.

4.3.2 Threshold-based Approach

Here, we do not use any nil questions to train the neural answer span extraction model. The model is entirely trained on passage-question pairs where every question has a valid answer in the corresponding passage. This approach assumes that when there is a valid answer, the beginning and ending pointers will have lower entropy. This results in a higher maximum joint probability of the beginning and end pointers. In contrast, when an answer is not present in the associated passage, the beginning and ending pointers will have higher entropy, resulting in a lower value of maximum joint probability. We set the maximum joint probability *threshold* based on the best Nil F1 score on the nil questions in the development set (by performing a grid search). Now, for a given test passage and question, we first compute the maximum of all the joint probabilities associated with all the answer spans. Let a_{span} be the answer span with highest joint probability p_{max}. We assign the final *answer* as follows:

$$answer = \begin{cases} Nil, & \text{if } p_{max} \leq threshold \\ a_{span}, & \text{otherwise} \end{cases} \tag{4.18}$$

Dataset		#Passages	#Questions
Train	NewsQA	10,938	92,549
	+Nil Qs		107,673
Dev	NewsQA	638	5,166
	+Nil Qs		5,988
Test	NewsQA	632	5,126
	+Nil Qs		5,971

Table 4.1: Statistics of the NewsQA dataset. #Passages and #Questions denote the number of passages and questions respectively.

4.4 Experiments

In this section, we present the experimental settings, results, followed by an analysis of the models.

4.4.1 Experimental Settings

We use the NewsQA dataset with nil questions (Trischler et al., 2017) in our experiments. Its training, development, and test sets consist of 10,938, 638, and 632 passages respectively and every passage is associated with some questions. In each subset, there are some questions that have no answers in the corresponding associated passages (i.e., the nil questions). The detailed statistics of the dataset are given in Table 4.1.

We compute exact match (EM) and F1 score for questions with valid answers. For questions without any valid answers, we compute Nil precision,

recall, and F1 scores:

$$\text{Nil precision} = \frac{\#\text{Correctly predicted Nil}}{\#\text{predicted Nil}} \tag{4.19}$$

$$\text{Nil recall} = \frac{\#\text{Correctly predicted Nil}}{\#\text{Nil questions}} \tag{4.20}$$

$$\text{Nil F1} = 2 \times \frac{\text{Nil precision} \times \text{Nil recall}}{\text{Nil precision} + \text{Nil recall}} \tag{4.21}$$

To compute the overall EM and F1 scores, we consider *Nil* as correct for the questions which do not have any valid answers. All evaluation scores reported in this work are in %.

All the neural network models are implemented in PyTorch[3]. We use the default hyper-parameters for all the answer span extractor models. We use the open source implementation of DrQA[4]. We use a third-party implementation of R-Net[5] whose performance is very similar to the original scores. We reimplemented BiDAF[6] and AMANDA[7] with PyTorch to easily integrate our proposed nil-aware answer extraction framework and make the training faster.

We integrate the nil-aware answer span extraction framework with each model keeping all the hyper-parameters unchanged. For nil-detection models, we use the same settings as (N)AMANDA. We use 300 hidden units for BiLSTMs and a total of 300 filters for the CNN-based models. We use dropout (Srivastava et al., 2014) with probability 0.3 for every trainable layer. We use

[3]http://pytorch.org

[4]https://github.com/facebookresearch/DrQA

[5]https://github.com/HKUST-KnowComp/MnemonicReader/blob/master/r_net.py

[6]Our implementation gives 3% lower F1 score compared to the reported results in Seo et al. (2017) on the SQuAD development set.

[7]Our reimplementation of AMANDA in PyTorch in this chapter gives 0.5% higher F1 score compared to the reported F1 score in Chapter 3 on the NewsQA test set.

binary cross-entropy loss and the Adam optimizer (Kingma and Ba, 2015) for training the nil-detection models.

4.4.2 Results

Tables 4.2 and 4.3 compare results of the nil-aware answer span extractor models with several pipeline and threshold-based models respectively. We also include the results of four standalone answer span extraction models on the test set without nil questions.

Table 4.2 shows that the end-to-end nil-aware models achieve the highest overall EM and F1 scores compared to all the corresponding pipeline systems. Note that the MP-BiLSTM nil detection model achieves a higher Nil F1 score compared to LR and MP-CNN. This is because BiLSTM is able to capture long-range contextual information to infer the existence of valid answers. Furthermore, AP-based models perform better compared to MP-based models as the attention mechanism used in AP-based models inherently identifies important contextual information. Due to this, the performance gap between AP-CNN and AP-BiLSTM is lower than the performance gap between MP-CNN and MP-BiLSTM. In addition to achieving a higher Nil F1 score than the strong nil detection baseline systems, nil-aware models manage to achieve competitive scores compared to the corresponding standalone answer span extractors on the test set where there are no nil questions. Hence, the nil answer handling components have no adverse effects on answer span extraction capability on the passage-question pairs when there exist valid answers.

Table 4.3 shows that the nil-aware models outperform the corresponding

Answer Extractor	Nil Detector	Test Set (w/o Nil Questions)		Test Set (with Nil Questions)				
		EM	F1	Nil Precision	Nil Recall	Nil F1	Overall EM	Overall F1
BiDAF	-	42.5	57.5	-	-	-	36.5	49.4
	LR	39.6	53.2	33.1	28.9	30.9	38.1	49.7
	MP-BiLSTM	40.1	54.2	52.5	48.3	50.3	41.3	53.4
	MP-CNN	42.3	57.2	73.8	15.0	24.9	38.4	51.2
	AP-BiLSTM	40.5	54.7	55.3	47.5	51.1	41.5	53.7
	AP-CNN	40.1	54.3	50.8	39.9	44.7	40.1	52.3
NBiDAF		40.8	54.7	48.0	59.6	53.2	43.5	55.4
R-Net	-	49.9	64.0	-	-	-	42.8	54.8
	LR	46.4	58.8	33.1	28.9	30.9	43.9	54.6
	MP-BiLSTM	47.3	60.3	52.5	48.3	50.3	47.5	58.6
	MP-CNN	49.7	63.6	73.8	15.0	24.9	44.8	56.7
	AP-BiLSTM	47.6	60.7	55.3	47.5	51.1	47.6	58.8
	AP-CNN	47.2	60.4	50.8	39.9	44.7	46.2	57.5
NR-Net		47.0	60.8	53.6	57.6	55.5	48.5	60.3
DrQA	-	50.0	64.0	-	-	-	42.9	54.8
	LR	46.3	58.8	33.1	28.9	30.9	43.8	54.6
	MP-BiLSTM	47.1	60.2	52.5	48.3	50.3	47.3	58.5
	MP-CNN	49.6	63.5	73.8	15.0	24.9	44.7	56.7
	AP-BiLSTM	47.4	60.6	55.3	47.5	51.1	47.4	58.8
	AP-CNN	47.0	60.2	50.8	39.9	44.7	46.0	57.3
NDrQA		48.5	61.8	53.5	57.2	55.3	49.8	61.1
AMANDA	-	49.2	64.2	-	-	-	42.2	55.1
	LR	45.4	58.8	33.1	28.9	30.9	43.1	54.5
	MP-BiLSTM	46.2	60.2	52.5	48.3	50.3	46.5	58.5
	MP-CNN	48.3	63.3	73.8	15.0	24.9	43.6	56.5
	AP-BiLSTM	46.3	60.6	55.3	47.5	51.1	46.5	58.7
	AP-CNN	45.9	60.0	50.8	39.9	44.7	45.1	57.1
NAMANDA		48.6	62.2	57.1	56.7	56.9	49.7	61.5

Table 4.2: Performance Comparison with pipeline approaches on the NewsQA test set (LR – Logistic Regression, MP – Max-pooling, AP – Attentive Pooling).

Answer Extractor	Nil Answer Handling	Test Set (w/o Nil Questions)		Test Set (with Nil Questions)				
		EM	F1	Nil Precision	Nil Recall	Nil F1	Overall EM	Overall F1
BiDAF	No	42.5	57.5	-	-	-	36.5	49.4
	Yes	37.9	48.3	25.8	60.2	36.1	41.0	50.0
NBiDAF		40.8	54.7	48.0	59.6	53.2	43.5	55.4
R-Net	No	49.9	64.0	-	-	-	42.8	54.8
	Yes	45.3	54.7	25.5	53.6	34.6	46.5	54.5
NR-Net		47.0	60.8	53.6	57.6	55.5	48.5	60.3
DrQA	No	50.0	64.0	-	-	-	42.9	54.8
	Yes	40.8	48.1	23.1	68.0	34.5	44.6	51.0
NDrQA		48.5	61.8	53.5	57.2	55.3	49.8	61.1
AMANDA	No	49.2	64.2	-	-	-	42.2	55.1
	Yes	42.2	51.3	24.0	59.5	34.2	44.6	52.5
NAMANDA		48.6	62.2	57.1	56.7	56.9	49.7	61.5

Table 4.3: Performance comparison with threshold-based approaches on the NewsQA test set.

threshold-based models. Note that all four answer span extraction models, when used in a threshold-based approach for nil detection, produce low Nil precision and relatively higher Nil recall. The low precision significantly degrades performance on the test set without nil questions. These models often return *Nil* since it is critical to find suitable values for the required *threshold*. This is because NewsQA passages are often very long and as a result, probability distributions with higher entropy for answer pointer selection lead to irregular maximum joint probability *threshold* values.

We perform statistical significance tests using paired t-test and bootstrap resampling. Performances of all the nil-aware models (in terms of overall EM and F1) are significantly better ($p < 0.01$) than the corresponding best pipeline models and threshold-based approaches.

4.4.3 Analysis

For better understanding, we present further experiments and analysis of one of the proposed model NAMANDA.

In addition to linear aggregation, we experiment with BiLSTM-based and CNN-based aggregation models. When we use BiLSTM aggregation, Eq. (4.4) is modified to $\mathbf{y}_t^a = \mathbf{h}_t^{=} + \mathbf{h}_t^{\perp}$, where

$$\begin{aligned} \mathbf{h}_t^{=} &= \text{BiLSTM}(\mathbf{y}_t^{=}, \mathbf{h}_{t-1}^{=}, \mathbf{h}_{t+1}^{=}) \\ \mathbf{h}_t^{\perp} &= \text{BiLSTM}(\mathbf{y}_t^{\perp}, \mathbf{h}_{t-1}^{\perp}, \mathbf{h}_{t+1}^{\perp}) \end{aligned} \tag{4.22}$$

We use equal numbers of unigram, bigram, and trigram filters for CNN-based aggregation. Similar to BiLSTM-based aggregation, we add the CNN outputs

Aggregation Model	w/o Nil	with Nil	
	EM (F1)	Nil Prec/ Rec (F1)	Overall EM (F1)
BiLSTM	47.6 (60.9)	56.5/53.5 (55.0)	48.4 (59.9)
CNN	46.2 (60.0)	52.7/54.5 (53.6)	47.3 (59.2)
Linear (NAMANDA)	47.8 (60.5)	51.2/57.2 (54.0)	49.1 (60.0)

Table 4.4: Effect of different aggregation models on the NewsQA dev set.

for $\mathbf{Y}^{=}$ and $\mathbf{Y}^{\perp}$. Table 4.4 shows that linear aggregation achieves the highest overall F1 score despite using the least number of parameters.

Table 4.5 shows the results of NAMANDA on the NewsQA development set when different components are removed such as character embeddings, question-passage joint encoding, and the second LSTM for the answer-ending pointer. When question-passage joint encoding is ablated, self-attentive encoding is formed as well as decomposed with respect to sequence-level passage encoding. When we remove the second LSTM for the answer-ending pointer, a feed-forward network is used instead. It is clear from Table 4.5 that question-passage joint encoding has the highest impact.

Figure 4.2(a) and (b) show the results of NAMANDA on different question (excluding the stop words) and passage lengths respectively on the NewsQA development set. With increasing question length, the Nil F1 score also improves. This is because with more information in a question, it becomes easier to detect whether the associated passage contains a valid answer. Increasing Nil F1 scores also help to improve the overall F1 scores. However, the overall F1 score degrades with the increasing length of the associated passage.

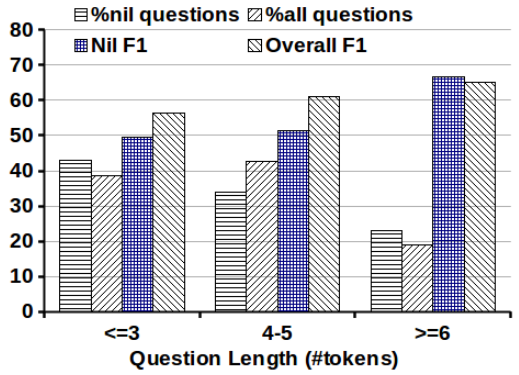
Model	w/o Nil	with Nil	
	EM (F1)	Nil Prec/ Rec (F1)	Overall EM (F1)
– character embeddings	46.3 (59.2)	51.9/54.1 (53.0)	47.4 (58.5)
– q-passage joint encoding	32.5 (43.9)	41.4/58.8 (48.6)	36.1 (45.9)
– second LSTM	47.6 (60.3)	56.7/51.0 (53.7)	48.0 (59.0)
NAMANDA	47.8 (60.5)	51.2/57.2 (54.0)	49.1 (60.0)

Table 4.5: Ablation studies on the NewsQA dev set.

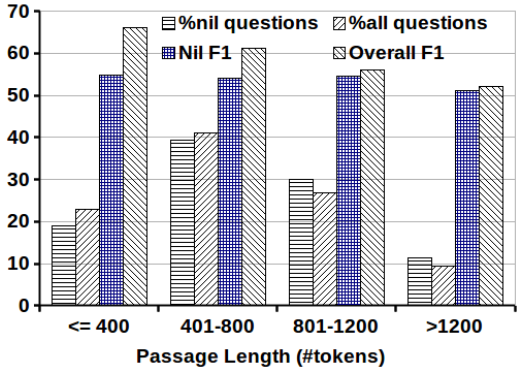
When the associated passage is long, it is difficult for the answer span extractor to extract an answer for a question which has a valid answer, due to the increasing amount of potentially distracting information. The Nil F1 scores remain similar for passages consisting of not more than 1,200 tokens. Beyond that, the Nil F1 score degrades a little as it becomes very challenging to infer the existence of a valid answer accurately with an increasing amount of potentially distracting information present in the passage.

Nil detection is itself a very challenging task. Performances of the nil-aware models are worse than the corresponding answer extractor models on the test set without nil questions as the Nil Precision is lower than 100%. We carried out an experiment to evaluate the performance of NAMANDA on development sets with varying number of nil questions. As the proportion of nil questions in a set increases, NAMANDA outperforms AMANDA by a larger margin on overall scores.

We also conducted a qualitative error analysis to better understand the shortcomings of our proposed model NAMANDA. We performed the error



(a)



(b)

Figure 4.2: Results for different (a) question and (b) passage lengths on NewsQA dev set.

analysis for two cases, and for each case, we randomly picked 50 instances from the NewsQA development set.

- First, we analyzed the instances for which the gold answers were Nil but the model incorrectly predicted text spans as answers. For these instances, the error types are summarized in Table 4.6 with examples. 44% of the errors occurred due to the mismatch of context between the passage and the question. For instance, in the example given in Table 4.6, the gold answer is Nil as there is no mention of Kingsley winning

Oscar. However, the model incorrectly predicted the answer as *Monday 31 May: 0300* as there is a match on *Kingsley* between the question and the passage. Notably, the model predicted the correct answer type to the *when* question, and predicted a date. We found that 28% of the errors are due to impossible condition, i.e., the answer is Nil because there is no valid continuous span in the passage which can answer the question. However, there might be multiple potential answers which do not occur in the passage in a continuous span. 12% of the errors occur due to insufficient question context. The remaining 16% of the errors are due to multiple other reasons such as detection of antonyms, entity swapping, dataset noise (incorrect annotation), etc.

- Second, we analyzed the instances for which there were valid answers, but the model incorrectly predicted the answers as Nil. For these instances, the error types are summarized in Table 4.7 with examples. 36% of the errors occurred when it requires complex reasoning. For instance, in the example given in Table 4.7, it requires a complex coreference resolution, i.e., iPhone 4S is Apple's new phone and it has *a new function called Siri*. 28% of the errors occurred due to insufficient context in the question. For instance, in the example given in Table 4.7, it is not clear from the question whether it is asking about the business goal or design goal of *Ralph Lauren*. We found that 20% of the errors are due to paraphrasing issues. The remaining 16% of the errors are due to the absence of clear answers, ambiguous question content, dataset noise (i.e., the question is actually unanswerable), etc.

4.5 Further Advances

Concurrent to our work, Rajpurkar et al. (2018) released the SQuAD 2.0 dataset (also known as SQuADRUN) by expanding the original SQuAD dataset with unanswerable questions. Since then, there has been much advancement in this area. Several research papers have been published based on the recent introduction of contextual embedding models, such as ELMO (Peters et al., 2018) and BERT (Devlin et al., 2019). For instance, Hu et al. (2019) proposed a read-then-verify system, which utilizes an ELMO-based neural reader to extract candidate answers, produces Nil answer probabilities, and leverages an answer verifier to decide whether the predicted answer is entailed by the input snippets. Zhang et al. (2019a) proposed to incorporate explicit contextual semantics from pre-trained semantic role labeling and introduced an improved language representation model, namely Semantics-aware BERT (SemBERT), which is capable of explicitly absorbing contextual semantics over a BERT backbone. Zhang et al. (2019b) proposed using syntax to guide the text modeling of both passages and questions by incorporating explicit syntactic constraints into attention mechanisms for better, linguistically motivated word representations. They proposed a dual contextual architecture called syntax-guided network (SG-Net), which consists of a BERT context vector and a syntax-guided context vector, to provide more fine-grained representation. Lan et al. (2019) proposed ALBERT which has a self-supervised loss that focuses on modeling inter-sentence coherence and showed that it consistently helps downstream tasks with multi-sentence inputs. Moreover, they also presented parameter-reduction techniques to lower

memory consumption and increase the training speed of BERT.

4.6 Summary

In this chapter, we have focused on nil-aware answer span extraction for RC-based QA. A nil-aware QA system only extracts a span of text from the associated passage as an answer to a given question if and only if the passage contains a valid answer. We have proposed a novel nil-aware answer span extraction framework based on evidence decomposition and aggregation that can be easily integrated into several recently proposed neural answer span extraction models. We have also developed several pipeline and threshold-based models using advanced neural architectures for comparison. Experiments on the NewsQA dataset show that our proposed framework, when integrated to the answer span extraction models, achieves better performance compared to all the corresponding pipeline and threshold-based models.

In Chapters 3 and 4, we have focused on single-turn machine reading comprehension-based QA. In the following chapter, we will focus on a more challenging multi-turn conversational QA task.

Context Mismatch (44%) Q: When did Kingsley win oscar? ... Watch The Screening Room Cannes special on CNN at the following dates and times: Wednesday 27 May: 0730, ..., **Monday 31 May: 0300** (All times GMT) In the most high-profile amalgamation of Indian and western talent yet, Academy Award-winning actor Ben Kingsley stars with Bollywood superstar Amitabh Bachchan in a drama about a pair of maths geniuses. Ben Kingsley who stars in "Teen Patti" is the first Academy Award-winner ever to take a role in a Bollywood movie. ...
Impossible Condition (28%) Q: When is the new date of the shows? ... to change the schedule but in the end we wanted to ensure that all of Michael's fans attending the concerts get the same quality in staging and level of entertainment," said the Ticketmaster e-mail sent to someone who bought tickets for the third show. "In order to deliver a phenomenal and unprecedented show – the first show on the 8th July will take place on **13th July 2009**," according AEG Live, the promoter of the London concerts. "The subsequent shows on 10th July will be moved to 1st March 2010, 12th July will be moved to 3rd March 2010, and the show on the 14th July will be moved to 6th March 2010." The delay is ...
Insufficient Question Context (12%) Q: Exactly what type of problems? ... who later miscarried their unborn child. Dr. Jennifer Arnold and husband Bill Klein, who both have **skeletal dysplasia**, a bone-growth disorder that causes dwarfism, have ...

Table 4.6: Examples of different error types and their percentages. Ground truth answers are Nil, but NAMANDA incorrectly predicted the answers as text spans. The incorrectly predicted answers are bold-faced.

Complex Reasoning (36%) Q: What will Apple 4s have? ... The iPhone 4S may not look any different from its predecessor, but it is Apple's only model with a sort of robot living inside. Apple's new phone, which was announced on Tuesday to be sold in stores on October 14, will have **a new function called Siri**. The program lets people bark commands ...
Insufficient Question Context (28%) Q: What is his goal? ... he's Ralph Lauren, and we're not. Ralph Lauren has his eye on **China and Japan**. For four decades no ... Lauren instinctively caught something that was in the air before any of his competitors had a chance to grab it – the desire, not just to be a success but to look like one before you'd even achieved your goal. What's more, Lauren ...
Paraphrasing Issues (20%) Q: What makes them the leader? ... Brazil jolted the global health community in 1996 when it began guaranteeing **free anti-retroviral treatment to HIV/AIDS patients** ... estimated $2 billion the program has saved Brazil in hospital costs between 1996 and 2004. Brazil's efforts to reverse the tide of the AIDS epidemic have become the object of admiration in the global health community, but the trailblazer is encountering new challenges. When Brazil decided to guarantee free anti-retrovirals, there were 10,000 people being treated ...

Table 4.7: Examples of different error types and their percentages. Ground truth answers are not Nil, i.e., there are valid answers present in the associated passages. However, NAMANDA incorrectly predicted the answers as Nil. The ground truth answers are bold-faced.

Chapter 5

Learning to Ask Clarification Questions in Conversational Question Answering

Recently proposed conversational question answering systems lack the ability to ask a follow-up clarification question when a given question is underspecified. In this chapter, we focus on developing a question answering system that can predict the answer to a question in a conversation, and has the ability to ask a follow-up clarification question if the original question is underspecified. We propose a pipeline approach that consists of an answer prediction model and a follow-up question generation model. The answer prediction model is based on a dual co-attention network while the follow-up question generation model is based on a sequence-to-sequence neural network enhanced with a copying mechanism. We experimented on the recently

released ShARC dataset, and show that our approach achieves performance competitive to the state of the art. The current state-of-the-art model E3 (Zhong and Zettlemoyer, 2019) is very ShARC-specific. An important contribution of this work is that the proposed approach is generic and can be applied to any conversational question answering task which requires clarification question generation.

5.1 Background

Traditional machine reading comprehension-based question answering (QA) tasks share the single-turn setting of answering questions with the help of associated text passages (Rajpurkar et al., 2016; Trischler et al., 2017; Joshi et al., 2017; Dunn et al., 2017; Welbl et al., 2018; Rajpurkar et al., 2018). Usually, the questions are independent of each other in this setting. However, humans naturally seek answers via conversation, which carries over context through the conversation flow.

To mimic the process of natural conversation, several datasets such as QuAC (Choi et al., 2018) and CoQA (Reddy et al., 2019) have been recently released which require a system to infer the answer for a question by understanding a series of question-answer pairs in the conversation history, in addition to the associated text passage. Subsequently, several neural network-based models have been proposed which are able to answer questions in a conversational setting (Huang et al., 2019; Zhu et al., 2018). Although QuAC and CoQA became very popular, they do not contain underspecified questions that require a system to generate clarification questions. However, a

Passage: If you are a female Vietnam Veteran with a child who has a birth defect or you are a child of a female Vietnam Veteran with a birth defect, the child may be eligible for VA-financed health care. **Conversation History**: f_{q_1}: Are you a female Vietnam Veteran? f_{a_1}: Yes f_{q_2}: Does your child have a birth defect? f_{a_2}: Yes **Question**: I registered as a single not knowing my son would have a birth defect. Is my child eligible for VA-financed health care? **Reference Answer**: Yes
Passage: In order to qualify for this benefit program, you must be a Native American/American Indian who has been accepted or enrolled in an accredited degree program, college or university to study in the field of health care, and you or your family member must be enrolled in a federally recognized American Indian tribe or Alaska Native village. **Conversation History**: f_{q_1}: Are you a Native American/American Indian? f_{a_1}: Yes **Question**: I am currently in my third year studying for my Doctorate of Medicine at Cambridge University. Do I qualify for this benefit program? **Reference Answer**: Are you or your family member enrolled in a federally recognized American Indian tribe or Alaska Native village?

Table 5.1: Example instances from the ShARC development set.

practical conversational QA system must possess the ability to ask a clarification question when a question is underspecified. Concurrently, Saeidi et al. (2018) released the ShARC dataset consisting of regulatory texts such as traffic laws, benefit programs, tax and visa regulations, etc. Different from QuAC and CoQA, the ShARC dataset contains underspecified questions that require a system to generate clarification questions.

In this work, we focus on the ShARC dataset. We propose a pipeline system, namely LEIA, for learning to ask clarification questions in conversational question answering. LEIA consists of an answer classification model and a clarification question generation model. Given a passage and a conversation

history, the answer classification model predicts the answer to a question. The answer can be *Yes*, *No*, *Irrelevant*, or *More*. For an underspecified question, we expect the answer classification model to predict *More*, and then the question generator generates a clarification question. Consider the examples given in Table 5.1 (taken from the ShARC development set). For the first example, the answer classification model should predict *Yes*, since the child has a *birth defect* and the mother is a *female Vietnam Veteran*, as given in the conversation history. However, in the second example, the question is underspecified as there is no information provided about *the family* in the question or the conversation history. Hence, the answer classification model should predict *More*, and the clarification question generator is expected to generate a clarification question.

The proposed answer classifier is based on novel dual co-attention networks for jointly encoding the associated passage, the question, and the conversation history. First, we use a co-attention network that captures the interaction between the conversation history and the question to compute a reformulated question representation. Then another co-attention network captures the interaction between the reformulated question and the associated passage to predict the answer. The clarification question generator is based on a sequence-to-sequence neural network model enhanced with a copying mechanism. As shown in the second example in Table 5.1, a clarification question often repeats several words from the associated passage and the conversation history. Hence, the copying mechanism allows the clarification question generator to copy words from the passage and conversation history while generating the words.

Recently, Zhong and Zettlemoyer (2019) proposed an entailment-driven extract and edit (E^3) network. E^3 extracts implicit decision rules from the text, computes whether each rule is entailed by the conversation history, and answers the question. To do so, it needs to identify explicit rule spans from the passage. The rule spans are obtained by using edit distance-based matching between the passage words and the question words in the conversation history. Additionally, E^3 also needs to access the full dialog tree to extract the rule spans which are often not covered by a single instance. This makes the model rather restrictive since there might not be exact matching of words between the passage and the questions in the conversation history. Moreover, in practical settings, a machine would not be able to access the future question-answer pairs in the dialog tree for extracting the rule spans from the passage. In contrast to E^3, our proposed system is generic and does not rely on any task-specific preprocessing, such as explicit rule span extraction.

Overall, the contributions of this work are:

- We developed a *generic* conversational QA system that can *answer* a question, or *ask a clarification question* if the question is underspecified.
- Our proposed system achieves performance competitive to the state of the art on the ShARC dataset. Additionally, it does not have any dataset-specific preprocessing steps in contrast to a recent state-of-the-art ShARC-specific model (Zhong and Zettlemoyer, 2019).
- We show that the proposed pipeline approach performs better than the joint model where the two models are trained end-to-end in a multi-task learning approach.

5.2 Problem Definition

The conversational QA task in ShARC can be formalized on a per-utterance basis. Let $\mathcal{Q}$ be an input question, and $\mathcal{P}$ be an input support rule text, i.e., the associated passage. Let $\mathcal{V}$ be the vocabulary. Furthermore, let $\mathcal{H} = (f_{q_1}, f_{a_1}),\ (f_{q_2},\ f_{a_2}), \ldots,\ (f_{q_n}, f_{a_n})$ be a conversation history where each f_{q_i} is a follow-up question, and each $f_{a_i} \in \{\, \textit{Yes}, \textit{No}\}$ is a follow-up answer. We concatenate the follow-up questions and their corresponding answers with a special separator token. In this way, a conversation history can be given as: $\mathcal{H} = f_{q_1} \mid\mid f_{a_1} \mid\mid f_{q_2} \mid\mid f_{a_2} \mid\mid \ldots \mid\mid f_{q_n} \mid\mid f_{a_n}$ A question is often associated with a scenario which describes the context of the question. For simplicity, we concatenate the scenario and the question together and represent it as the question $\mathcal{Q}$. An input instance can be referred to $\mathcal{X} = (\mathcal{Q}, \mathcal{P}, \mathcal{H})$. Given an input $\mathcal{X}$, the task is to predict an answer $\mathcal{Y} \in \{\, \textit{Yes}, \textit{No}, \textit{Irrelevant}\} \cup \mathcal{V}^{\star}$, that specifies whether the answer is *Yes*, *No*, *Irrelevant*, or a clarification question. *Irrelevant* is the target answer when the associated passage $\mathcal{P}$ is not related to $\mathcal{Q}$, i.e., the question cannot be answered. In our proposed approach, the answer classifier first predicts whether the answer is *Yes*, *No*, *Irrelevant*, or *More*, and if the answer is *More*, the clarification question generator generates a follow-up clarification question.

5.3 Proposed Approach

In this section, we describe our proposed system LEIA, which consists of two components: (1) the Answer Classifier (AC) predicts the answer and (2) the

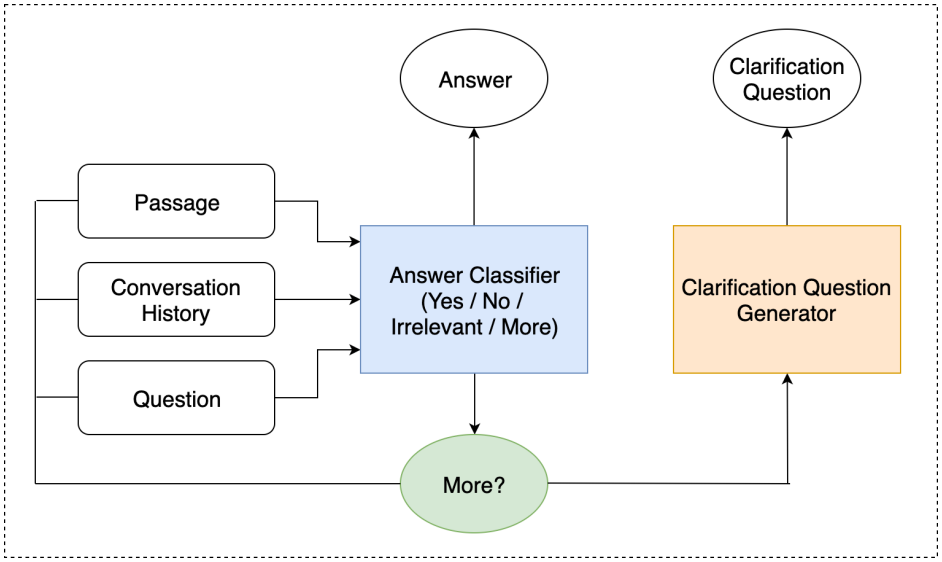


Figure 5.1: Overview of the proposed architecture. Given a passage, conversation history, and a question, the answer classifier predicts the answer. If the predicted answer is *More*, the clarification question generator generates a follow-up clarification question.

Question Generator (QG) generates a clarification question if the question is underspecified, i.e., if the AC predicts *More*. The overview of our proposed system is depicted in Figure 5.1.

5.3.1 Answer Classifier

The answer classifier predicts the answer to a question by reasoning over the associated passage and the conversation history. An overview of the answer classifier is depicted in Figure 5.2.

Embedding and Encoding

We use BERT (Devlin et al., 2019) for contextual word embedding. We take the final transformer layer output for all *wordpieces* (Wu et al., 2016) in a sen-

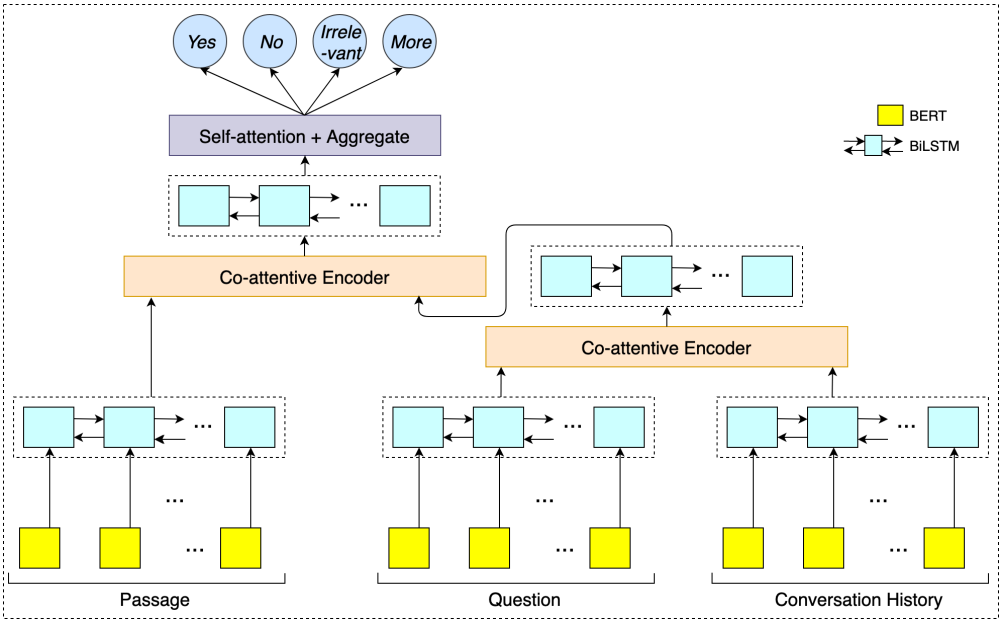


Figure 5.2: Answer classification model architecture.

tence/passage. We do not update BERT's internal weights during training. If a word is tokenized into multiple wordpieces, the output vectors corresponding to the wordpieces are averaged to obtain the embedding of the word.

Next, we use bi-directional LSTMs (BiLSTM) (Hochreiter and Schmidhuber, 1997) to contextually encode the embedding vectors. Suppose the passage consists of T tokens, the question consists of U tokens, and the conversation history consists of V tokens. The BiLSTM outputs are unfolded across time. The outputs for the passage, question, and the conversation history are represented as $\mathbf{P} \in \mathbb{R}^{T \times H}$, $\mathbf{Q} \in \mathbb{R}^{U \times H}$, and $\mathbf{C} \in \mathbb{R}^{V \times H}$, respectively. H represents the number of hidden units. The BiLSTM outputs are obtained by concatenating the forward and backward LSTM outputs.

Dual Co-attentive Encoding

We propose a novel dual co-attentive encoder to compute a combined representation of the passage, question, and conversation history. Co-attention network (Lu et al., 2016) was initially proposed for the visual question answering (VQA) task. In VQA, it was used to jointly reason over image and question attention. Xiong et al. (2017) adapted the co-attention network for machine reading comprehension task (e.g., SQuAD). They used the co-attention network to capture the interaction between the question and the passage for answer span extraction. While Xiong et al. (2017) used a single co-attentive encoder, we propose a dual co-attentive encoder which makes use of two co-attentive encoder modules to capture the interaction between the associated passage, the question, and the conversation history. Moreover, Xiong et al. (2017) used the co-attention network for a simpler machine reading comprehension-based question answering task while we use the dual co-attentive encoder for a more complex conversational question answering task, in which it needs to reason over the conversation history in addition to the associated passage and the question. The proposed dual co-attentive encoder is described as follows.

We start by computing a similarity matrix that captures the similarity between the question words and the words in the conversation history. This similarity matrix, $\mathbf{A}_{q,c} \in \mathbb{R}^{U \times V}$ is given as:

$$\mathbf{A}_{q,c} \;=\; \mathbf{Q}\;\mathbf{W}\;\mathbf{C}^{\top} \quad , \tag{5.1}$$

where $\mathbf{W} \in \mathbb{R}^{H \times H}$ is trainable bi-linear matrix. Intuitively, $A_{q,c}(i,j)$ represents the relevance of the jth word of the conversation history with respect to the ith word of the question. We apply a row-wise softmax function for normalization to produce the attention weights across the conversation history for each word in the question, resulting in $\tilde{\mathbf{A}}_{q,c}$. Next, we compute the summary or weighted representation of the conversation history corresponding to each question word.

$$\mathbf{Q}_c \;=\; \tilde{\mathbf{A}}_{q,c}\,\mathbf{C} \;\in\; \mathbb{R}^{U \times H} \tag{5.2}$$

Next, the initial question encoding vectors in $\mathbf{Q}$ (i.e., each row of $\mathbf{Q}$) and the vectors in $\mathbf{Q}_c$ are concatenated and passed through a BiLSTM, which results in $\tilde{\mathbf{Q}} \in \mathbb{R}^{U \times H}$. $\tilde{\mathbf{Q}}$ essentially captures the interaction between the question and the conversation history by jointly encoding them.

The same process is repeated for the passage encoding and $\tilde{\mathbf{Q}}$. We compute a similarity matrix, $\mathbf{A}_{p,q} \;=\; \mathbf{P}\;\mathbf{W}\;\tilde{\mathbf{Q}}^{\top} \;\in \mathbb{R}^{T \times U}$, which captures the similarity between the passage word encoding vectors and the joint encoding vectors in $\tilde{\mathbf{Q}}$. Next, we normalize each row of $\mathbf{A}_{p,q}$ using the softmax function, resulting in $\tilde{\mathbf{A}}_{p,q} \in \mathbb{R}^{T \times U}$. Now, similar to Equation (5.2), we compute the summary or the weighted representation of the question and the conversation history corresponding to each passage word.

$$\mathbf{P}_{q,c} \;=\; \tilde{\mathbf{A}}_{p,q}\,\tilde{\mathbf{Q}} \;\in \mathbb{R}^{T \times H} \tag{5.3}$$

To jointly encode the associated passage, the question, and the conver-

sation history, we concatenate the encoding vectors of $\mathbf{P}$ and $\mathbf{P}_{q,c}$, and feed them to a BiLSTM which results in $\tilde{\mathbf{P}} \in \mathbb{R}^{T \times H}$.

Self-attentive Encoding and Aggregation

We use a self-attention layer on top of $\tilde{\mathbf{P}}$, which can effectively aggregate evidence from the joint encoding vectors to infer the answer. First, we calculate the self-attention matrix, $\mathbf{A}_s = \tilde{\mathbf{P}}\ \mathbf{W}\ \tilde{\mathbf{P}}^\top \in \mathbb{R}^{T \times T}$, where $\mathbf{W} \in \mathbb{R}^{H \times H}$ is a trainable bi-linear matrix. Next, we apply a row-wise softmax for normalization, resulting in $\tilde{\mathbf{A}}_s$. Now, the self-attentive encoding vectors can be aggregated as $\mathbf{P}_s = \tilde{\mathbf{A}}_s \tilde{\mathbf{P}} \in \mathbb{R}^{T \times H}$. Then, we concatenate the joint encoding vectors in $\tilde{\mathbf{P}}$ with the self-attentive encoding vectors in $\mathbf{P}_s$, followed by a feed-forward layer, which results in $\mathbf{Y} \in \mathbb{R}^{T \times H}$. We aggregate the vectors in $\mathbf{Y}$ for the output layer. If the tth row of $\mathbf{Y}$ is represented as $\mathbf{y}_t \in \mathbb{R}^H$, the aggregated vector $\mathbf{x} \in \mathbb{R}^H$ is:

$$a_t \ \propto \ \exp(\mathbf{y}_t\ \mathbf{w}^\top)\ ; \quad \mathbf{x} \ = \ \mathbf{a}\ \mathbf{Y} \tag{5.4}$$

where $\mathbf{w} \in \mathbb{R}^H$ is trainable vector and the weights in $\mathbf{a}$ sum to one. Note that, the bi-linear matrices used to compute $\mathbf{A}_{q,c}$, $\mathbf{A}_{p,q}$, and $\mathbf{A}_s$ are three different matrices with different learned parameters.

Output Layer

In the output layer, we use a simple feed-forward layer for classification. The number of output units is four for four different classes, i.e., *Yes*, *No*, *Irrelevant*, and *More*. For training, we minimize the cross-entropy loss summing

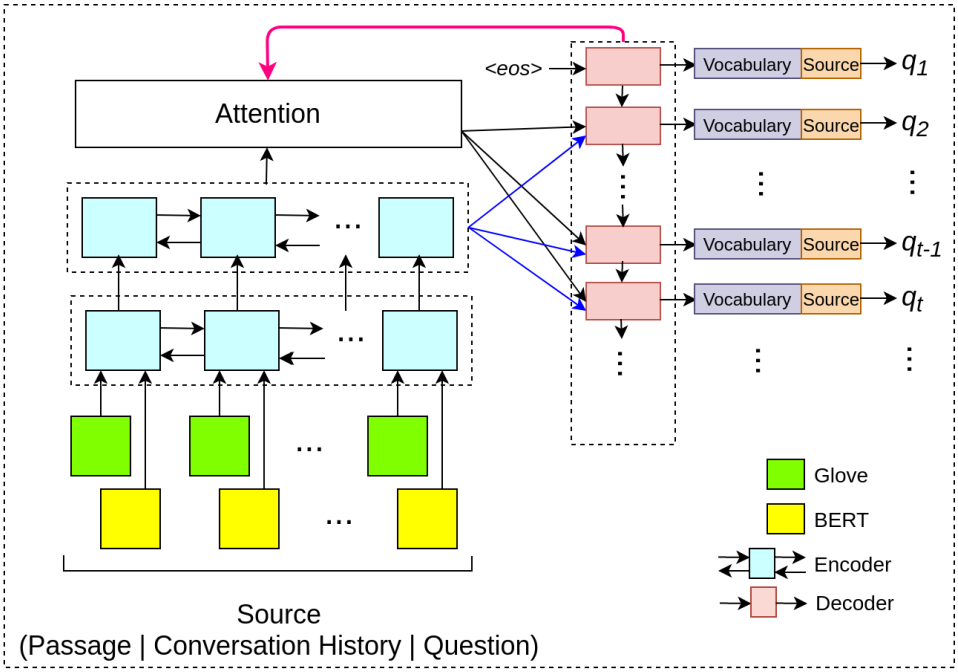


Figure 5.3: Follow-up clarification question generation model architecture.

over all the training instances.

5.3.2 Clarification Question Generator

In this section, we describe the clarification question generation model. The overview of the model is depicted in Figure 5.3. We develop a sequence-to-sequence (Seq2Seq) model enhanced with a copying mechanism for generating the follow-up clarification questions by inferring over the associated passage, the question, and the conversation history. Gu et al. (2016) initially proposed the Seq2Seq model with copy mechanism (CopyNet), and successfully used it for text summarization. The copying mechanism refers to the mechanism that locates a portion of the input text and puts it into the output sequence. While Gu et al. (2016) employed the copy mechanism for summarization, we use it

for follow-up clarification question generation as clarification questions often repeat words from the passage, the question, and the conversation history.

Encoder

We create the source sequence by concatenating the associated passage, the conversation history, and the question with special separator tokens. In the remainder of the thesis, we denote the length of the source sequence as L. We obtain the embedding vectors of the source words from GloVe (Pennington et al., 2014) and BERT (Devlin et al., 2019). We use two layers of stacked BiLSTMs to transform the source embedding vectors into a series of hidden state representations. The BiLSTM outputs are unfolded across time, and are represented as $\mathbf{M} \in \mathbb{R}^{L \times H}$.

Decoder

We use an LSTM to read the encoding vectors in $\mathbf{M}$ and predict the target sequence. The decoder predicts target words based on two perspectives: (1) probabilities over the words in the vocabulary (2) probabilities over the source words. It determines whether the word should be copied from the source text or the decoder LSTM should generate a word from the vocabulary. Similar to Bahdanau et al. (2015), the tth decoder LSTM state $\mathbf{s}_t \in \mathbb{R}^H$ is represented as:

$$\mathbf{s}_t \;=\; f\Big(\boldsymbol{\phi}(q_{t-1}),\; \mathbf{s}_{t-1},\; \mathbf{c}_t\Big) \quad , \tag{5.5}$$

where $\boldsymbol{\phi}(q_{t-1})$ denotes the representation of the previously predicted clarification question word, $\mathbf{s}_{t-1}$ represents the previous decoder state, and $\mathbf{c}_t$

represents the context vector. The context vector $\mathbf{c}_t \in \mathbb{R}^H$ is obtained by an attention mechanism over the encoder hidden state vectors in $\mathbf{M}$. If the ith row of $\mathbf{M}$ is represented as $\mathbf{m}_i \in \mathbb{R}^H$, $\mathbf{c}_t$ is given as:

$$\beta_i \propto \exp(\mathbf{s}_{t-1}\mathbf{W}\mathbf{m}_i^\top); \quad \mathbf{c}_t \;=\; \boldsymbol{\beta}\;\mathbf{M} \quad , \tag{5.6}$$

where $\mathbf{W} \in \mathbb{R}^{H \times H}$ is a trainable matrix and the weights in $\boldsymbol{\beta}$ sum to one. The representation of the previously predicted clarification question word, $\boldsymbol{\phi}(q_{t-1})$, is derived by concatenating its word embedding representation $\mathbf{e}(q_{t-1})$, and its location-specific hidden state representation $\boldsymbol{\psi}(q_{t-1})$ from $\mathbf{M}$, if q_{t-1} is copied from the source sentence. More specifically,

$$\boldsymbol{\phi}(q_{t-1}) \;=\; \mathbf{e}(q_{t-1}) \;||\; \boldsymbol{\psi}(q_{t-1}) \quad , \tag{5.7}$$

where $||$ represents the concatenation operation. As q_{t-1} might appear in multiple locations in the source sequence, $\boldsymbol{\psi}(q_{t-1})$ is obtained by the weighted sum of the hidden states in $\mathbf{M}$ corresponding to q_{t-1} (Gu et al., 2016). If q_{t-1} does not appear in the source sequence, we assign a zero vector to $\boldsymbol{\psi}(q_{t-1})$. Next, we describe the prediction of the clarification question words.

Prediction with Copy Mechanism

We represent the target vocabulary with $\mathcal{V}$, and use UNK for any out-of-vocabulary (OOV) word. Additionally, for copying, we consider another set of words, $\mathcal{S}$, consisting of all the unique words in a particular source sequence. Overall, the instance-specific vocabulary is $\mathcal{V} \cup \text{UNK} \cup \mathcal{S}$. Given the decoder

LSTM state $\mathbf{s}_t$, the probability of generating any clarification question word q_t is given by the mixture of two probabilities:

$$p(q_t|\mathbf{s}_t, q_{t-1}, \mathbf{c}_t, \mathbf{M}) = p_g(q_t|\mathbf{s}_t, q_{t-1}, \mathbf{c}_t, \mathbf{M}) + p_c(q_t|\mathbf{s}_t, q_{t-1}, \mathbf{c}_t, \mathbf{M}) \ , \tag{5.8}$$

where p_g represents the probability for generating q_t from the vocabulary, and p_c represents the probability for copying q_t from the source sequence. The probabilities for generation and copying are given as follows:

$$p_g(q_t|.) = \begin{cases} \frac{1}{Z}\exp(\xi_g(q_t)), & q_t \in \mathcal{V} \\ \frac{1}{Z}\exp(\xi_g(\text{UNK})), & q_t = \text{UNK} \end{cases} \tag{5.9}$$

$$p_c(q_t|.) = \frac{1}{Z}\sum_{i:w_i=q_t} \exp(\xi_c(w_i)), \quad q_t \in \mathcal{S} \tag{5.10}$$

where $\xi_g(.)$ and $\xi_c(.)$ are the scoring functions for generating q_t from the vocabulary $\mathcal{V}$, and copying from the source sequence, respectively. Z is the shared normalization factor for both p_g and p_c.

$$Z = \sum_{w \in \mathcal{V} \cup \text{UNK}} \exp(\xi_g(w)) + \sum_{w \in \mathcal{S}} \exp(\xi_c(w)) \tag{5.11}$$

Due to the shared normalization term Z, the two modes, i.e., generation from the vocabulary, and copying from the source, compete through a softmax function (Gu et al., 2016).

The scoring functions are as follows:

$$\xi_g(q_t = v) = \quad \mathbf{v}\ \mathbf{s}_t^\top, \quad v \in \mathcal{V} \cup \text{UNK} \tag{5.12}$$

$$\xi_c(q_t = v_j) = \quad \tanh(\mathbf{m}_j\ \mathbf{W})\ \mathbf{s}_t^\top, \quad v_j \in \mathcal{S} \tag{5.13}$$

where $\mathbf{v} \in \mathbb{R}^H$ is the embedding vector representation of the word v and $\mathbf{W} \in \mathbb{R}^{H \times H}$ is a trainable matrix. Note that $p_c(q_t) = 0$ if q_t does not appear in the source sequence, and $p_g(q_t) = 0$ when q_t only appears in the source sequence.

5.4 Experiments

We start by describing the experimental settings. Then, we present the experimental results and analysis.

5.4.1 Settings

We experimented with the recently proposed ShARC dataset (Saeidi et al., 2018), which is built over regulatory texts. The ShARC dataset is developed from distinct snippets of rule text (e.g., tax and visa regulations, traffic rules, etc.). Each example has an input question and a dialog tree. At every step in the conversation, there is a follow-up question with *Yes/No* answer. The dataset is comprised of utterances from every tree. Overall, the dataset consists of 32,436 utterances. These utterances are divided into training, development, and test sets targeting a 70%/10%/20% split. Each utterance consists of a rule text (i.e., the associated passage), a series of question-answer pairs

as conversation history, a question, and an answer or a follow-up clarification question if the question is underspecified. The test set is hidden, and we need to submit our source code and trained model for evaluation.

We use Spacy[1] for tokenization. We use the pre-trained BERT (Devlin et al., 2019) model for word embeddings, and the model parameters are kept fixed during training. The number of hidden units in all the LSTMs is 50 ($H = 100$). We use dropout (Srivastava et al., 2014) with probability 0.5 for every learnable layer. The minibatch size is fixed at 10 for the answer classifier and 32 for the clarification question generator. For the clarification question generator, we use 300-dimension pre-trained word vectors from GloVe (Pennington et al., 2014) and we do not update them during training. We use the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 0.001. Following Saeidi et al. (2018), we use micro- and macro-averaged accuracies for evaluating the answer classification models. The clarification question generation models are evaluated using BLEU (Papineni et al., 2002) scores.

5.4.2 Main Results

Table 5.2 compares the performance of LEIA with the recently published conversational QA systems for ShARC[2] just before submitting this thesis, including BiSon of Lawrence et al. (2019). Saeidi et al. (2018) proposed the combined model (CM), which consists of a feature-engineered random for-

[1]https://spacy.io/api/tokenizer

[2]There are a couple of other unpublished systems that appeared on the ShARC leaderboard (https://sharc-data.github.io/leaderboard.html) with better performance than LEIA at the time of submitting this thesis.

Model	Micro Acc.	Macro Acc.	BLEU-1	BLEU-4
CM†	61.9	68.9	54.4	34.4
NMT†	44.8	42.8	34.0	7.8
BERT-QA⋆	63.6	70.8	46.2	36.3
E^{3}⋆	67.6	73.3	54.1	38.7
BiSon	66.9	71.6	58.8	44.3
LEIA	64.9	71.4	55.3	38.9

Table 5.2: Comparison results on the ShARC test set. †(Saeidi et al., 2018) ⋆(Zhong and Zettlemoyer, 2019)

Model	Micro Acc.	Macro Acc.
– BERT	61.0	68.2
– Self-Attn.	65.8	71.8
– Dual Co-attn.	64.8	71.0
– Qs-Hist Co-attn.	67.5	73.4
– Psg-QH Co-attn.	65.6	71.8
LEIA-AC	68.3	74.1

Table 5.3: Ablation studies for the answer classifier on the ShARC development set.

est answer classifier, and a rule-based clarification question generator. Saeidi et al. (2018) also proposed a neural model (NMT) using a multi-task learning framework. It has a classification head and a GRU-based decoder. CM performs better than NMT because CM uses a rule-based follow-up question generation model which allows it to exploit the structure of the problem which NMT does not seem to do effectively. Table 5.2 shows that our proposed system outperforms both CM and NMT by large margins.

Zhong and Zettlemoyer (2019) proposed an entailment-driven extract and edit network (E^{3}) and a baseline BERT-based extractive QA model (BERT-QA). E^{3} is a conversational machine reading model that jointly extracts a set of decision rules from the rule text passage while reasoning about which

rules are entailed by the conversation history and which rules still need to be edited to create follow-up clarification questions. Our proposed system outperforms the BERT-QA model on both the answer classification and the clarification question generation tasks. While the performance of our system is lower than E^3 on the answer classification task, it performs slightly better on the clarification question generation task. In addition, our proposed system is generic and does not have any particular dataset-specific module in contrast to the E^3 that contains modules that are ShARC-specific, such as rule extraction by string-matching, usage of full dialog trees to extract rules from the associated passages, etc.

Lawrence et al. (2019) proposed a sequence generation model that leverages both past and future tokens. They proposed a bi-directional sequence generation process (BiSon) by employing special placeholder tokens in the output sequence. These placeholder tokens are replaced by tokens of the output vocabulary. This effectively allows a transformer encoder to attend to both past and future, not-yet-produced tokens. To determine where to position the placeholder tokens at training time, they explored different stochastic placeholder replacement strategies, based on Bernoulli and Gaussian random variables. This is crucial as the models need to be exposed to a large number of heterogeneous placeholder configurations. They also explored several strategies for iteratively generating a complete output sequence from an initial sequence of placeholders at inference time. Notably, their approach is not restricted to produce the output sequence from left to right. Lawrence et al. (2019) achieved significant improvements over the prior state of the art on the ShARC dataset in the clarification question generation task. BiSon

Model	BLEU-1	BLEU-2	BLEU-3	BLEU-4
– BERT	50.8	40.8	35.9	33.1
– Copy	20.6	9.9	6.1	3.7
– 2nd BiLSTM	53.3	43.5	37.0	31.8
LEIA-QG	59.6	51.7	46.9	43.3

Table 5.4: Ablation studies for the clarification question generator on the instances of ShARC development set that require follow-up clarification question generation.

Passage: In order to qualify for this benefit program, homeowners and renters *must have sustained physical damage* and be *located in a disaster declared county.*
Conversation History: f_{q_1}: Are you *located in a disaster declared county*? f_{a_1}: Yes
Question: My house is *no more damaged than when I moved* into it but it's not like I can really afford to fix it up as I only earn 7,6000 a year from my job as a seamstress. Do I qualify for this loan?
Reference Answer: No; **LEIA Prediction**: No; **w/o Dual Co-attention Prediction**: Yes

Table 5.5: An illustrating example of the dual co-attentive encoder.

outperforms LEIA on both the answer classification task and the clarification question generation task.

5.4.3 Ablation Studies

To better understand how the proposed system works, we perform rigorous experiments by removing different components from the answer classifier and the clarification question generator.

Table 5.3 shows the performance of the LEIA answer classifier (LEIA-AC) when different components are removed. When we do not consider BERT for word embeddings, GloVe (Pennington et al., 2014) vectors are used for word embeddings. When we do not consider self-attentive encoding, the aggre-

gation (Equation (5.4)) step is applied on the joint encoding vectors in $\tilde{\mathbf{P}}$. When the dual co-attentive encoder is removed, we concatenate the contextual encoding representations of all the words in the passage, question, and conversation history. The concatenated representation can be denoted as $\mathbf{D} \in \mathbb{R}^{(T+U+V)\times H}$. The self-attentive encoding and the aggregation step are applied on $\mathbf{D}$ for answer prediction. The dual co-attention helps to validate the rules mentioned in the passage by capturing the interactions among the passage, conversation history, and the question. Consider the example in Table 5.5. While the proposed model could correctly predict the answer by cross-validating the two required rules (given in italics), it fails to infer the answer correctly when we do not consider the dual co-attention encoder. When we do not consider the co-attention between the question and the conversation history (– Qs-Hist Co-attn.), the question encoding vectors in $\mathbf{Q}$ and the conversation history vectors in $\mathbf{C}$ are concatenated and fed to the other co-attentive encoder. Similarly, when the co-attention between the passage encoding and the joint encoding of the question and the conversation history is not considered (– Psg-QH Co-attn.), the passage encoding vectors in $\mathbf{P}$ and the joint encoding vectors in $\tilde{\mathbf{Q}}$ are concatenated. As shown in Table 5.3, LEIA-AC outperforms all the competing models and justifies the importance of the individual components.

In Table 5.4, we show the effectiveness of different components in the LEIA clarification question generator (LEIA-QG). Note that the question generation models are trained and evaluated only on the instances which require clarification question generation. When we do not consider BERT for source word embeddings, only pre-trained GloVe vectors are used. When the

Model	Micro Acc.	Macro Acc.	BLEU-1	BLEU-4
Joint	67.9	74.1	48.3	30.0
Pipeline	68.3	74.1	59.6	45.1

Table 5.6: Comparison with the jointly learned model on the ShARC development set.

copy mechanism is disabled, a standard attention-based sequence-to-sequence model (Bahdanau et al., 2015) is used for clarification question generation. In this case, the probability of generating a clarification question word is the same as the generation probability p_g. When the second BiLSTM is not considered for encoding, the first BiLSTM outputs are directly fed to the decoder. Table 5.4 shows that all the mentioned components in LEIA-QG have a large impact on the final performance. Copy mechanism has the greatest impact because a clarification question often contains words from the associated passage and the conversation history. Disabling the copy mechanism hinders the model to output the words which are only present in the source sequence, and thereby degrades the performance by a large margin.

5.4.4 Joint Task Learning

We also experimented with jointly training the answer classifier and the clarification question generator. In this case, we use a multi-task learning setup. For the instances with answer *Yes*, *No*, or *Irrelevant*, we fix the clarification question generator output as NULL. For the joint model, we use a shared embedding layer for the two tasks. During training, the overall training loss, $\mathcal{L}$ is computed by a weighted sum of the answer classification loss $\mathcal{L}_{ac}$ and

the clarification question generation loss $\mathcal{L}_{qg}$.

$$\mathcal{L} \;=\; (1-\alpha)\;\mathcal{L}_{ac} \;+\; \alpha\;\mathcal{L}_{qg} \quad , \tag{5.14}$$

where α is the weighting factor, and $0 < \alpha < 1$. We found that the joint model performs best at $\alpha = 0.999$. During prediction, if the answer classification module predicts *More*, we take the output from the question generation module.

Table 5.6 shows the performance comparison between the joint model and the pipeline system (LEIA). Although the joint model achieves competitive performance for the answer classification task, it performs poorly on the clarification question generation task. This is primarily because the training data of ShARC is not large enough, making it difficult for the joint model to learn the two tasks together.

As shown in Figure 5.4, we observed that the variations in performance of the joint model across different tasks are not consistent when we vary the value of α. Hence, it requires a very careful tuning of α for effectively learning the two tasks jointly.

5.4.5 Error Analysis

Table 5.7 shows the confusion matrix for the proposed answer classifier model predictions on the ShARC development set. Our proposed model predicted incorrectly for 31.7% instances. We randomly sampled 50 such instances and manually analyze them. We observe that in 16% of the cases, the question partially matched the passage or the conversation history, and the model

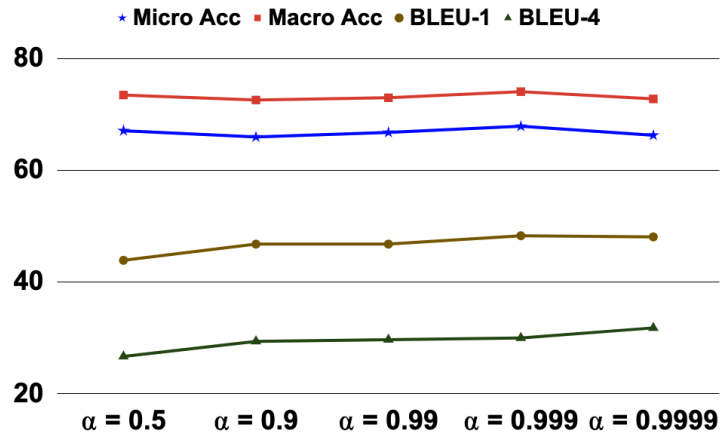


Figure 5.4: Performance of the joint model on the development set for different values of α.

		Gold			
		Yes	No	Irrelevant	More
Predicted	Yes	554	106	0	144
	No	121	511	0	134
	Irrelevant	0	0	136	2
	More	122	85	5	350

Table 5.7: Confusion matrix for the answer classifier predictions on the ShARC development set.

arrived at *Yes* or *No* answer instead of predicting *More*. For 20% of the cases, the rules were too complex to resolve. The errors might be due to the presence of distracting words in the question or the conversation history in 16% of the cases. We observed that the model failed to answer correctly for 12% of the cases when it requires numerical comparison. The remaining errors occurred for various reasons such as when it requires external knowledge, the passage mentions some exception rules, etc.

Both valid

Passage: You may be an eligible Veteran if:

* You have an Honorable Discharge, and

* You have a High School Diploma (or equivalent) - or 12 hours of college credit, and

* You meet the requirements under one of the four categories, listed on the Department of Veterans Affairs (VA) Montgomery GI Bill Active Duty page

Conversation History: N/A

Question: I graduated from high school. May I be an eligible Veteran?

Reference Answer: Do you have an honorable discharge?

LEIA Prediction: Do you have an Honorable Discharge ?

E3 Prediction: do You have an Honorable Discharge ?

Both invalid

Passage: ## What youll need

Youll need accurate information about your:

* savings

* income, including your partners (from payslips, for example)

* existing benefits and pensions (including anyone living with you)

* outgoings (such as rent, mortgage, childcare payments)

Conversation History:

f_{q_1}: Is this information about your savings? f_{a_1}: Yes

Question: The information is about my savings. I do not have info about my existing benefits. Do I need information about this?

Reference Answer: Do you have accurate info about your income?

LEIA Prediction: Are you accurate information about your savings ?

E3 Prediction: do you income, including your partners (from payslips, for example) ?

LEIA valid, E3 invalid

Passage: VA can also reimburse expenses for covered travel to and from your doctor or health care facility within a reasonable commuting area. Preauthorization is required for travel outside of the commuting area.

Conversation History: N/A

Question: I left the country a year ago, but I didn't get any preauthorization. I'm living in a government owned area. Can the VA reimburse my travel to and from the doctor?

Reference Answer: Is it in a reasonable commuting area?

LEIA Prediction: Is it in a reasonable commuting area ?

E3 Prediction: are you travel outside of the commuting area ?

LEIA invalid, E3 valid

Passage: ## Items that qualify for the zero rate

You may be able to apply zero VAT when you sell the following to an eligible charity:

* equipment for making talking books and newspapers

* lifeboats and associated equipment, including fuel

* medicine or ingredients for medicine

* resuscitation training models

Conversation History:

f_{q_1}: Q: Is it equipment for making talking books and newspapers? f_{a_1}: No

Question: I am selling rowboats, along with lots of other stuff. Can I apply zero VAT to this item?
Reference Answer: Are you selling medicine or ingredients for medicine?
LEIA Prediction: Is it resuscitation training models ?
E3 Prediction: is it medicine or ingredients for medicine ?

Table 5.8: Examples illustrating the quality of generated clarification questions by LEIA and E3.

We also manually analyzed 50 randomly chosen instances from the development set to compare the generated question quality between LEIA and E3. We found that both generated perfectly valid questions for 58% of the cases. Both generated incorrect questions for 16% of the cases. For 18% of the cases, our system generated valid questions but E3 could not. For the remaining 8% of the cases, E3 generated the questions correctly but our system could not. Table 5.8 presents some example instances for each of these cases.

We also perform an error analysis of the proposed system to understand the impact of conversation history. Figure 5.5 shows the performance of LEIA on the ShARC development set instances with different numbers of turns in the conversation history. The results are obtained for different disjoint sets of the ShARC development set, corresponding to the number of question-answer pairs in the conversation history. When the number of history turns is zero, the instances do not contain any conversation history. Overall, the proposed system performs better in both answer classification and clarification question generation when there are more turns in the conversation history. This is

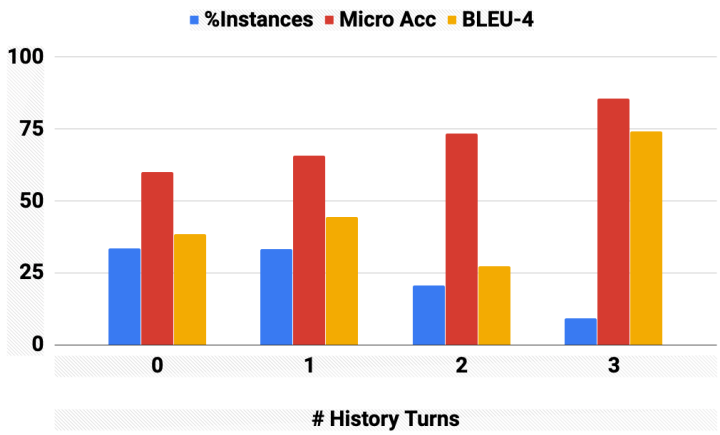


Figure 5.5: Performance of LEIA on the development set instances with different numbers of conversation history turns.

because when there exist more turns in the conversation, the model has more context to effectively capture the interaction between the current question and the conversation history, and hence performs better on both tasks.

5.5 Summary

In summary, we have proposed a system, named LEIA, for conversational question answering. Given an associated passage and a conversation history, it either answers a question or generates a clarification question if the posed question is underspecified. LEIA consists of an answer classifier that is based on a dual co-attention network, and a clarification question generator which is based on a sequence-to-sequence neural network with a copy mechanism. Experiments on the ShARC dataset shows the effectiveness of the proposed system. We have also performed detailed ablation studies to show the impact of different components used to build the system. Additionally, we have also shown that the pipeline-based approach performs better than the joint model

in which the answer classifier and the question generator are trained together in an end-to-end fashion.

Chapter 6

Conclusions

In this thesis, we provide a brief overview of early works in question answering, and how it evolved in recent years. We have also elucidated how we contributed to the development of machine reading comprehension-based question answering.

We have addressed issues related to the current answer span extraction approaches for machine reading comprehension (RC) and have proposed solutions to these problems. We have introduced a question-focused multi-factor attention network (AMANDA) that can be trained in an end-to-end fashion. While the proposed multi-factor attentive encoding method accumulates relevant facts distributed across multiple sentences, question-focused attention pointing helps to learn the important question words and thereby implicitly infer the answer type during the extraction of the answer span. We have shown that this system significantly outperforms prior approaches.

Additionally, we have addressed a more complex nil-aware answer span

extraction task. Current answer span extraction approaches for RC always make an impractical assumption that there always exists a valid answer in the associated passage for a given question. We developed a unified nil-aware answer span extraction framework that can be integrated easily with our previously proposed model AMANDA and several other recently proposed answer span extraction models developed for RC (including BiDAF, DrQA, and R-Net). We have also shown that the combined models can be trained in an end-to-end fashion. Experiments have shown that our proposed framework, when integrated with other answer extraction models, yields significantly better performance compared to the corresponding pipeline and threshold-based models.

Finally, we proposed a conversational QA system that can answer questions in a conversation. It is also able to generate follow-up clarification questions if the questions are underspecified. The proposed system is based on a pipeline approach, which consists of an answer classification model, followed by a clarification question generator model. The answer classifier first predicts the answer, and based on whether the predicted answer suggests asking a clarification question, the clarification question generator model generates a question. The answer classifier model is based on a dual co-attentive encoder, and the clarification question generator is based on a sequence-to-sequence neural network enhanced with a copy mechanism. Experiments have shown that our proposed system achieves competitive performance.

Although it is really exciting how this field has progressed in the past few years, there is still a long way to go towards genuine human-level reading comprehension. There are still many existing challenges and open questions

that we will need to address in the future. Some potential future works are outlined below:

1. One of the major issues of the current machine reading comprehension models is that they fail to answer questions that require long answers, such as *how* and *why* questions. In the future, we will have to focus on developing models that can truly understand what is being discussed, rather than just picking a short span of text as an answer.

2. Another interesting future work is how we can leverage machine reading comprehension models in a more practical question answering setting. In most of the machine reading comprehension-based QA tasks, a question is associated with a predefined document. It would be interesting to see how such models can help when it needs to reason over multiple documents that are obtained from information retrieval engines.

3. Although there has been significant progress in the QA field, we are still far from having a QA system that can explain the reasoning behind giving a particular answer to a question. An explainable QA system can provide transparency to the underlying computational model and can help develop interfaces to enable the end-users to access and validate the interpretation and allow information feedback.

4. Although there is a surge of interest in conversational machine reading comprehension-based QA in the last few years, the tasks are far away from real-world settings. We hope to witness more research on improving the interaction ability of the conversational QA systems, making

the tasks closer to real-world scenarios.

Bibliography

Ion Androutsopoulos, Graeme Ritchie, and Peter Thanisch. MASQUE/SQL – an efficient and portable natural language query interface for relational databases. In *Proceedings of IEA/AIE*, 1993.

Ion Androutsopoulos, Graeme Ritchie, and Peter Thanisch. Natural language interfaces to databases – an introduction. *Natural Language Engineering*, 1:29 – 81, 1995.

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

Satanjeev Banerjee and Alon Lavie. METEOR: An automatic metric for MT evaluation with improved correlation with human judgments. In *Proceedings of the ACL Workshop on Intrinsic and Extrinsic Evaluation Measures for Machine Translation and/or Summarization*, 2005.

Jonathan Berant, Vivek Srikumar, Pei-chun Chen, Brad Huang, Christopher D. Manning, Abby Vander Linden, Brittany Harding, and Peter Clark. Modeling biological processes for reading comprehension. In *Proceedings of EMNLP*, 2014.

Weijie Bian, Si Li, Zhao Yang, Guang Chen, and Zhiqing Lin. A compare-aggregate model with dynamic-clip attention for answer selection. In *Proceedings of CIKM*, 2017.

Karl H. Bläsius. Knowledge based control of the LILOG inference engine: Kinds of metaknowledge. *Text Understanding in LILOG: Integrating Computational Linguistics and Artificial Intelligence Final Report on the IBM Germany LILOG–Project*, 546:428–438, 1991.

Antoine Bordes, Y-Lan Boureau, and Jason Weston. Learning end-to-end goal-oriented dialog. In *Proceedings of ICLR*, 2017.

John Burger, Claire Cardie, Vinay Chaudhri, Robert Gaizauskas, Sanda Harabagiu, David Israel, Christian Jacquemin, Chin-Yew Lin, Steve Maiorano, George Miller, Dan Moldovan, Bill Ogden, John Prager, Ellen Riloff, Amit Singhal, Rohini Shrihari, Tomek Strzalkowski, Ellen Voorhees, and Ralph Weischedel. Issues, tasks and program structures to roadmap research in question & answering (Q&A). Technical report, NIST, 2001.

Robin D. Burke, Kristian J. Hammond, Vladimir Kulyukin, Steven L. Lytinen, Noriko Tomuro, and Scott Schoenberg. Question answering from frequently asked question files: Experiences with the FAQ FINDER system. *AI Magazine*, 18(2):57 – 66, 1997.

Xin Cao, Gao Cong, Bin Cui, Christian Sndergaard Jensen, and Ce Zhang. The use of categorization information in language models for question retrieval. In *Proceedings of CIKM*, 2009.

Yunbo Cao, Huizhong Duan, Chin-Yew Lin, Yong Yu, and Hsiao-Wuen Hon. Recommending questions using the MDL-based tree cut model. In *Proceedings of WWW*, 2008.

Eugene Charniak, Yasemin Altun, Rodrigo de Salvo Braz, Benjamin Garrett, Margaret Kosmala, Tomer Moscovich, Lixin Pang, Changhee Pyo, Ye Sun, Wei Wy, Zhongfa Yang, Shawn Zeller, and Lisa Zorn. Reading comprehension programs in a statistical-language-processing class. In *Proceedings of the 2000 ANLP/NAACL Workshop on Reading Comprehension Tests As Evaluation for Computer-based Language Understanding Sytems - Volume 6*, 2000.

Danqi Chen, Jason Bolton, and Christopher D. Manning. A thorough examination of the CNN / Daily Mail reading comprehension task. In *Proceedings of ACL*, 2016.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading Wikipedia to answer open-domain questions. In *Proceedings of ACL*, 2017.

Kyunghyun Cho, Bart van Merriënboer, Dzmitry Bahdanau, and Yoshua Bengio. On the properties of neural machine translation: Encoder–decoder approaches. In *Proceedings of Eighth Workshop on Syntax, Semantics and Structure in Statistical Translation*, 2014.

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen-tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. QuAC : Question answering in context. In *Proceedings of EMNLP*, 2018.

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

Christopher Clark and Matt Gardner. Simple and effective multi-paragraph reading comprehension. In *Proceedings of ACL*, 2018.

Hang Cui, Keya Li, Renxu Sun, Tat-Seng Chua, and Min-Yen Kan. National University of Singapore at the TREC 13 question answering main task. In *Proceedings of TREC*, 2004.

Yiming Cui, Zhipeng Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. Attention-over-attention neural networks for reading comprehension. In *Proceedings of ACL*, 2017.

Giovanni Da San Martino, Alberto Barron Cedeno, Salvatore Romeo, Antonio Uva, and Alessandro Moschitti. Learning to re-rank questions in community question answering using advanced features. In *Proceedings of CIKM*, 2016.

Hoa Trang Dang, Jimmy Lin, and Diane Kelly. Overview of the TREC 2006 question answering track. In *Proceedings of TREC*, 2006.

Hoa Trang Dang, Diane Kelly, and Jimmy Lin. Overview of the TREC 2007 question answering track. In *Proceedings of TREC*, 2007.

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

Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William Cohen, and Ruslan Salakhutdinov. Gated-attention readers for text comprehension. In *Proceedings of ACL*, 2017.

Cicero dos Santos, Luciano Barbosa, Dasha Bogdanova, and Bianca Zadrozny. Learning hybrid representations to retrieve semantically equivalent questions. In *Proceedings of ACL-IJCNLP*, 2015.

Cícero Nogueira dos Santos, Ming Tan, Bing Xiang, and Bowen Zhou. Attentive pooling networks. *arXiv preprint arXiv:1602.03609*, 2016.

Huizhong Duan, Yunbo Cao, Chin-Yew Lin, and Yong Yu. Searching questions by identifying question topic and question focus. In *Proceedings of ACL*, 2008.

Matthew Dunn, Levent Sagun, Mike Higgins, V. Ugur Güney, Volkan Cirik, and Kyunghyun Cho. SearchQA: A new Q&A dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179*, 2017.

Ahmed Elgohary, Chen Zhao, and Jordan Boyd-Graber. A dataset and baselines for sequential open-domain question answering. In *Proceedings of EMNLP*, 2018.

Minwei Feng, Bing Xiang, Michael R. Glass, Lidan Wang, and Bowen Zhou. Applying deep learning to answer selection: A study and an open task. In *Proceedings of ASRU*, 2015.

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. Building Watson: An overview of the DeepQA project. *AI Magazine*, 31(3):59–79, 2010.

Simone Filice, Danilo Croce, Alessandro Moschitti, and Roberto Basili. KeLP at SemEval-2016 Task 3: Learning semantic relations between questions and answers. In *Proceedings of the Workshop on SemEval*, 2016.

David Golub, Po-Sen Huang, Xiaodong He, and Li Deng. Two-stage synthesis networks for transfer learning in machine comprehension. *arXiv preprint arXiv:1706.09789*, 2017.

Bert F. Green, Jr., Alice K. Wolf, Carol Chomsky, and Kenneth Laughery. Baseball: An automatic question-answerer. In *IRE-AIEE-ACM (Western)*, 1961.

Jiatao Gu, Zhengdong Lu, Hang Li, and Victor O.K. Li. Incorporating copying mechanism in sequence-to-sequence learning. In *Proceedings of ACL*, 2016.

Daya Guo, Duyu Tang, Nan Duan, Ming Zhou, and Jian Yin. Dialog-to-action: Conversational question answering over a large-scale knowledge base. In *Advances in NIPS*, 2018.

Sanda Harabagiu and Andrew Hickl. Methods for using textual entailment in open-domain question answering. In *Proceedings of COLING and ACL*, 2006.

Sanda M. Harabagiu, Dan I. Moldovan, Christine Clark, Mitchell Bowden, John Williams, and Jeremy Bensley. Answer mining by combining extraction techniques with abductive reasoning. In *Proceedings of TREC*, 2003.

Sanda M. Harabagiu, Dan I. Moldovan, Christine Clark, Mitchell Bowden, Andrew Hickl, and Patrick Wang. Employing two question answering systems in TREC 2005. In *Proceedings of TREC*, 2005.

Michael Heilman and Noah A. Smith. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Proceedings of NAACL*, 2010.

Matthew Henderson, Blaise Thomson, and Steve Young. Deep neural network approach for the dialog state tracking challenge. In *Proceedings of the SIGDIAL*, 2013.

Karl Moritz Hermann, Tomás Kociský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. Teaching machines to read and comprehend. In *Advances in NIPS*, 2015.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. The goldilocks principle: Reading children's books with explicit memory representations. In *Proceedings of ICLR*, 2016.

Lynette Hirschman and Rob Gaizauskas. Natural language question answering: The view from here. *JNLE*, 7(4):275–300, 2001.

Lynette Hirschman, Marc Light, Eric Breck, and John D. Burger. Deep Read: A reading comprehension system. In *Proceedings of ACL*, 1999.

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

Minghao Hu, Furu Wei, Yuxing Peng, Zhen Huang, Nan Yang, and Dongsheng Li. Read + Verify: Machine reading comprehension with unanswerable questions. In *Proceedings of AAAI*, 2019.

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

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. TriviaQA: A large scale distantly supervised challenge dataset for reading comprehension. In *Proceedings of ACL*, 2017.

Rudolf Kadlec, Martin Schmid, Ondrej Bajgar, and Jan Kleindienst. Text understanding with the attention sum reader network. In *Proceedings of ACL*, 2016.

Michael Kaisser, Silke Scheible, and Bonnie Webber. Experiments at the University of Edinburgh for the TREC 2006 QA track. In *Proceedings of TREC*, 2006.

Boris Katz, Gregory Marton, Sue Felshin, Daniel Loreto, Ben Lu, Federico Mora, Ozlem Uzuner, Michael McGraw-Herdeg, Natalie Cheung, Yuan Luo, Alexey Radul, Yuan Shen, and Gabriel Zaccak. Question answering experiments and resources. In *Proceedings of TREC*, 2006.

Daniel Khashabi, Snigdha Chaturvedi, Michael Roth, Shyam Upadhyay, and Dan Roth. Looking beyond the surface: A challenge set for reading comprehension over multiple sentences. In *Proceedings of NAACL*, 2018.

Seokhwan Kim, Luis Fernando D'Haro, Rafael E. Banchs, Jason Williams, and Matthew Henderson. The fourth dialog state tracking challenge. In *Proceedings of IWSDS*, 2016.

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

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

Sosuke Kobayashi, Ran Tian, Naoaki Okazaki, and Kentaro Inui. Dynamic entity representation with max-pooling improves machine reading. In *Proceedings of NAACL*, 2016.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. The NarrativeQA reading comprehension challenge. *TACL*, 6:317–328, 2018.

Souvik Kundu and Hwee Tou Ng. A question-focused multi-factor attention network for question answering. In *Proceedings of AAAI*, 2018a.

Souvik Kundu and Hwee Tou Ng. A nil-aware answer extraction framework for question answering. In *Proceedings of EMNLP*, 2018b.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard H. Hovy. RACE: large-scale reading comprehension dataset from examinations. In *Proceedings of EMNLP*, 2017.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. *arXiv preprint arXiv:1909.11942*, 2019.

Carolin Lawrence, Bhushan Kotnis, and Mathias Niepert. Attending to future tokens for bidirectional sequence generation. In *Proceedings of EMNLP*, 2019.

Kenton Lee, Tom Kwiatkowski, Ankur Parikh, and Dipanjan Das. Learning recurrent span representations for extractive question answering. *arXiv preprint arXiv:1611.01436*, 2016.

Wendy Grace Lehnert. *The process of question answering*. PhD thesis, Yale University, 1977.

Tao Lei, Hrishikesh Joshi, Regina Barzilay, Tommi S. Jaakkola, Kateryna Tymoshenko, Alessandro Moschitti, and Lluis Marquez. Semi-supervised question retrieval with gated convolutions. In *Proceedings of NAACL*, 2016.

Jiwei Li, Will Monroe, Alan Ritter, Dan Jurafsky, Michel Galley, and Jianfeng Gao. Deep reinforcement learning for dialogue generation. In *Proceedings of EMNLP*, 2016a.

Qi Li, Tianshi Li, and Baobao Chang. Discourse parsing with attention-based hierarchical neural networks. In *Proceedings of EMNLP*, 2016b.

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

Yankai Lin, Haozhe Ji, Zhiyuan Liu, and Maosong Sun. Denoising distantly supervised open-domain question answering. In *Proceedings of ACL*, 2018.

Kenneth C. Litkowski. Exploring document content with XML to answer questions. In *Proceedings of TREC*, 2005.

Jiasen Lu, Jianwei Yang, Dhruv Batra, and Devi Parikh. Hierarchical question-image co-attention for visual question answering. In *Advances in NIPS*, 2016.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In *Advances in NIPS*, 2013.

Mitra Mohtarami, Yonatan Belinkov, Wei-Ning Hsu, Yu Zhang, Tao Lei, Kfir Bar, Scott Cyphers, and Jim Glass. SLS at SemEval-2016 Task 3: Neural-based approaches for ranking in community question answering. In *Proceedings of the Workshop on SemEval*, 2016.

Dan Moldovan, Sanda Harabagiu, Roxana Girju, Paul Morarescu, Finley Lacatusu, Adrian Novischi, Adriana Badulescu, and Orest Bolohan. LCC tools for question answering. In *Proceedings of TREC*, 2002.

Dan Moldovan, A Harabagiu, Christine Clark, Mitchell Bowden, John Lehmann, and John Williams. Experiments and analysis of LCC's two QA systems over TREC 2004. In *In Proceedings of TREC*, 2004.

Dan Moldovan, Mitchell Bowden, and Marta Tatu. A temporally-enhanced poweranswer in TREC 2006. In *Proceedings of TREC*, 2006.

Dan Moldovan, Christine Clark, and Mitchell Bowden. Lymba's PowerAnswer 4 in TREC 2007. In *Proceedings of TREC*, 2007.

Karthik Narasimhan and Regina Barzilay. Machine comprehension with discourse relations. In *Proceedings of ACL*, 2015.

Hwee Tou Ng, Leong Hwee Teo, and Jennifer Lai Pheng Kwan. A machine learning approach to answering questions for reading comprehension tests. In *Proceedings of the Joint SIGDAT Conference on EMNLP and Very Large Corpora*, 2000.

Hwee Tou Ng, Jennifer Lai Pheng Kwan, and Yiyuan Xia. Question answering using a large text database: A machine learning approach. In *Proceedings of EMNLP*, 2001.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. MS MARCO: A human generated machine reading comprehension dataset. In *Advances in NIPS*, 2016.

Boyuan Pan, Hao Li, Zhou Zhao, Bin Cao, Deng Cai, and Xiaofei He. MEMEN: Multi-layer embedding with memory networks for machine comprehension. *arXiv preprint arXiv:1707.09098*, 2017.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of ACL*, 2002.

Wenzhe Pei, Tao Ge, and Baobao Chang. Max-margin tensor neural network for Chinese word segmentation. In *Proceedings of ACL*, 2014.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. GloVe: Global vectors for word representation. In *Proceedings of EMNLP*, 2014.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *Proceedings of NAACL*, 2018.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of EMNLP*, 2016.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of ACL*, 2018.

Siva Reddy, Danqi Chen, and Christopher D. Manning. CoQA: A conversational question answering challenge. *TACL*, 2019.

Matthew Richardson, Christopher J. C. Burges, and Erin Renshaw. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of EMNLP*, 2013.

Ellen Riloff and Michael Thelen. A rule-based question answering system for reading comprehension tests. In *ANLP-NAACL 2000 Workshop: Reading Comprehension Tests as Evaluation for Computer-Based Language Understanding Systems*, 2000.

Salvatore Romeo, Giovanni Da San Martino, Alberto Barron-Cedeno, Alessandro Moschitti, Yonatan Be-linkov, Wei-Ning Hsu, Yu Zhang, Mitra Mohtarami, and James Glass. Neural attention for learning to rank questions in community question answering. In *Proceedings of COLING*, 2016.

Mrinmaya Sachan, Kumar Avinava Dubey, Eric P. Xing, and Matthew Richardson. Learning answer-entailing structures for machine comprehension. In *Proceedings of ACL*, 2015.

Marzieh Saeidi, Max Bartolo, Patrick Lewis, Sameer Singh, Tim Rocktäschel, Mike Sheldon, Guillaume Bouchard, and Sebastian Riedel. Interpretation of natural language rules in conversational machine reading. In *Proceedings of EMNLP*, 2018.

Amrita Saha, Vardaan Pahuja, Mitesh M. Khapra, Karthik Sankaranarayanan, and Sarath Chandar. Complex sequential question answering: Towards learning to converse over linked question answer pairs with a knowledge graph. In *Proceedings of AAAI*, 2018.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. In *Proceedings of ICLR*, 2017.

Aliaksei Severyn and Alessandro Moschitti. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of SIGIR*, 2015.

Bayan Abu Shawar and Eric Atwell. Different measurements metrics to evaluate a chatbot system. In *Proceedings of the Workshop on Bridging the Gap: Academic and Industrial Research in Dialog Technologies*, NAACL-HLT-Dialog '07. Association for Computational Linguistics, 2007.

Gehui Shen, Yunlun Yang, and Zhi-Hong Deng. Inter-weighted alignment network for sentence pair modeling. In *Proceedings of EMNLP*, 2017a.

Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. ReasoNet: Learning to stop reading in machine comprehension. In *Proceedings of KDD*, 2017b.

Robert F. Simmons. Answering English questions by computer: A survey. *Communications of the ACM*, 8(1):53–70, 1965.

Alessandro Sordoni, Michel Galley, Michael Auli, Chris Brockett, Yangfeng Ji, Margaret Mitchell, Jian-Yun Nie, Jianfeng Gao, and Bill Dolan. A neural network approach to context-sensitive generation of conversational responses. In *Proceedings of NAACL*, 2015.

Alessandro Sordoni, Phillip Bachman, and Yoshua Bengio. Iterative alternating neural attention for machine reading. *arXiv preprint arXiv:1606.02245*, 2016.

Martin M. Soubbotin. Patterns of potential answer expressions as clues to the right answers. In *Proceedings of TREC*, 2001.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *JMLR*, 15(1):1929–1958, 2014.

Renxu Sun, Jing Jiang, Yee Fan Tan, Hang Cui, Tat-seng Chua, and Min-yen Kan. Using syntactic and semantic relation analysis in question answering. In *Proceedings of TREC*, 2005.

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

Ming Tan, Bing Xiang, and Bowen Zhou. LSTM-based deep learning models for non-factoid answer selection. *arXiv preprint arXiv:1511.04108*, 2015.

Yi Tay, Anh Tuan Luu, and Siu Cheung Hui. Multi-granular sequence encoding via dilated compositional units for reading comprehension. In *Proceedings of EMNLP*, 2018a.

Yi Tay, Anh Tuan Luu, Siu Cheung Hui, and Jian Su. Densely connected attention propagation for reading comprehension. In *Proceedings of NeurIPS*, 2018b.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP, ACL*, 2017.

Kateryna Tymoshenko, Daniele Bonadiman, and Alessandro Moschitti. Learning to rank non-factoid answers: Comment selection in web forums. In *Proceedings of CIKM*, 2016.

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

Ellen M. Voorhees. The TREC-8 question answering track report. In *Proceedings of TREC*, 1999.

Ellen M. Voorhees. Overview of the TREC-9 question answering track. In *Proceedings of TREC*, 2000.

Ellen M. Voorhees. Overview of the TREC 2001 question answering track. In *Proceedings of TREC*, 2001.

Ellen M. Voorhees. Overview of the TREC 2002 question answering track. In *Proceedings of TREC*, 2002.

Ellen M. Voorhees. Overview of the TREC 2003 question answering track. In *Proceedings of TREC*, 2003.

Ellen M. Voorhees. Overview of the TREC 2004 question answering track. In *Proceedings of TREC*, 2004.

Ellen M. Voorhees and Hoa Trang Dang. Overview of the TREC 2005 question answering track. In *Proceedings of TREC*, 2005.

Di Wang and Eric Nyberg. A long short-term memory model for answer sentence selection in question answering. In *Proceedings of ACL-IJCNLP*, 2015.

Hai Wang, Mohit Bansal, Kevin Gimpel, and David McAllester. Machine comprehension with syntax, frames, and semantics. In *Proceedings of ACL*, 2015.

Kai Wang, Zhaoyan Ming, and Tat-Seng Chua. A syntactic tree matching approach to finding similar questions in community-based QA services. In *Proceedings of SIGIR*, 2009.

Mengqiu Wang and Christopher D. Manning. Probabilistic tree-edit models with structured latent variables for textual entailment and question answering. In *Proceedings of COLING*, 2010.

Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. What is the Jeopardy model? A quasisynchronous grammar for QA. In *Proceedings of EMNLP-CoNLL*, 2007.

Shuohang Wang and Jing Jiang. Machine comprehension using Match-LSTM and answer pointer. In *Proceedings of ICLR*, 2017.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of ACL*, 2017.

Zhiguo Wang, Haitao Mi, Wael Hamza, and Radu Florian. Multi-perspective context matching for machine comprehension. *arXiv preprint arXiv:1612.04211*, 2016a.

Zhiguo Wang, Haitao Mi, and Abraham Ittycheriah. Sentence similarity learning by lexical decomposition and composition. In *Proceedings of COLING*, 2016b.

Dirk Weissenborn. Reading twice for natural language understanding. *arXiv preprint arXiv:1706.02596*, 2017.

Dirk Weissenborn, Georg Wiese, and Laura Seiffe. Making neural QA as simple as possible but not simpler. In *Proceedings of CoNLL*, 2017.

Joseph Weizenbaum. ELIZA - a computer program for the study of natural language communication between man and machine. *Communications of the ACM*, 9(1):36–45, 1966.

Johannes Welbl, Pontus Stenetorp, and Sebastian Riedel. Constructing datasets for multi-hop reading comprehension across documents. *TACL*, 2018.

Robert Wilensky. Talking to UNIX in English: An overview of an on-line UNIX consultant. *AI Magazine*, 5:29 – 39, 1984.

Jason Williams, Antoine Raux, Deepak Ramachadran, and Alan Black. The dialog state tracking challenge. In *Proceedings of SIGDIAL*, 2013.

William A. Woods. Progress in natural language understanding: An application to lunar geology. In *Proceedings of National Computer Conference and Exposition*, 1973.

Yonghui Wu, Mike Schuster, Zhifeng Chen, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

Caiming Xiong, Victor Zhong, and Richard Socher. Dynamic coattention networks for question answering. In *Proceedings of ICLR*, 2017.

Jinxi Xu, Ana Licuanan, and Ralph Weischedel. TREC2003 QA at BBN: Answering definitional questions. In *Proceedings of TREC*, 2003.

Liu Yang, Qingyao Ai, Jiafeng Guo, and W. Bruce Croft. aNMM: Ranking short answer texts with attention-based neural matching model. In *Proceedings of CIKM*, 2016.

Zhilin Yang, Bhuwan Dhingra, Ye Yuan, Junjie Hu, William W. Cohen, and Ruslan Salakhutdinov. Words or characters? Fine-grained gating for reading comprehension. In *Proceedings of ICLR*, 2017.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of EMNLP*, 2018.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. XLNet: Generalized autoregressive pretraining for language understanding. In *Proceedings of NeurIPS*, 2019.

Xuchen Yao, Benjamin Van Durme, Chris Callison-Burch, and Peter Clark. Answer extraction as sequence tagging with tree edit distance. In *Proceedings of NAACL*, 2013.

Yi-Ting Yeh and Yun-Nung Chen. FlowDelta: Modeling flow information gain in reasoning for conversational machine comprehension. In *1st Workshop NLP for Conversational AI, ACL*, 2019.

Steve Young, Milica Gai, Blaise Thomson, and Jason D. Williams. POMDP-based statistical spoken dialog systems: A review. *Proceedings of the IEEE*, 101(5), 2013.

Yang Yu, Wei Zhang, Kazi Hasan, Mo Yu, Bing Xiang, and Bowen Zhou. End-to-end answer chunk extraction and ranking for reading comprehension. *arXiv preprint arXiv:1610.09996*, 2016.

Chen Zhang, Matthew Gerber, Tyler Baldwin, Steve Emelander, Joyce Y. Chai, and Rong Jin. Michigan State University at the 2007 TREC ciQA evaluation. In *Proceedings of TREC*, 2007.

Dell Zhang and Wee Sun Lee. A language modeling approach to passage question answering. In *Proceedings of TREC*, 2003.

Kai Zhang, Wei Wu, Haocheng Wu, Zhoujun Li, and Ming Zhou. Question retrieval with high quality answers in community question answering. In *Proceedings of CIKM*, 2014.

Zhuosheng Zhang, Yuwei Wu, Hai Zhao, Zuchao Li, Shuailiang Zhang, Xi Zhou, and Xiang Zhou. Semantics-aware BERT for language understanding. *arXiv preprint arXiv:1909.02209*, 2019a.

Zhuosheng Zhang, Yuwei Wu, Junru Zhou, Sufeng Duan, Hai Zhao, and Rui Wang. SG-Net: Syntax-guided machine reading comprehension. *arXiv preprint arXiv:1908.05147*, 2019b.

Victor Zhong and Luke Zettlemoyer. E3: Entailment-driven extracting and editing for conversational machine reading. In *Proceedings of ACL*, 2019.

Kangyan Zhou, Shrimai Prabhumoye, and Alan W. Black. A dataset for document grounded conversations. In *Proceedings of EMNLP*, 2018.

Chenguang Zhu, Michael Zeng, and Xuedong Huang. SDNet: Contextualized attention-based deep network for conversational question answering. *arXiv preprint arXiv:1812.03593*, 2018.