MULTI TASK LEARNING AND INCORPORATING COMMON SENSE KNOWLEDGE FOR QUESTION ANSWERING

BY

DHRUV AGARWAL

THESIS

Submitted in partial fulfillment of the requirements
for the degree of Master of Science in Computer Science
in the Graduate College of the
University of Illinois at Urbana-Champaign, 2019

Urbana, Illinois

Adviser:

Associate Professor Julia Hockenmaier

ABSTRACT

Question Answering (QA) system is an automated approach to retrieve correct responses to the questions asked by human in natural language. Reading comprehension (RC)in contrast to information retrieval, requires integrating information and reasoning about events, entities, and their relations across a full document. Immense progress has been made in the recent years for this task, since the advent of deep learning and use of sequence to sequence models for NLP. This thesis deals with two complex tasks in Question Answering with their own inherent challenges: Multi Task Learning for Narrative Question Answering, which involves developing models to deal with the complexity of the domain of stories, movie scripts and human written answers, and second task is to develop novel ways of incorporating common sense knowledge from external knowledge bases for automated question answering. The models developed for these tasks help to advance research in the area of question answering and highlights some of the shortcomings of the methods proposed in literature.

ACKNOWLEDGMENTS

My inspiration to conduct research in Natural Language processing came after I took up Professor Julia Hockenmaier's course on Deep learning in NLP. It provided me with the knowledge to choose a topic that would excite me and at the same time, novel enough to allow me to make significant contributions to NLP research. All the research performed during this phase has been the result of the papers I read and the discussions we had, which guided me in the right direction and eventually achieve the desired results.

I would also like to thank my lab mates to share the stresses of graduate school. It was great to have spent some time with Siddharth Satpathy, Anjali Narayan Chen, and Prashant Jayannavar. I learnt a great deal about the current research in NLP from our weekly reading group discussions. I would like to especially thank Prashant for guiding me at times when I was struggling to make progress and stay on track. Our thought provoking discussions helped me learn a lot and view problems from a different perspective.

Finally, I would also like to thank my parents and family for their unwavering support which motivated me to keep pushing through graduate school and achieve success.

TABLE OF CONTENTS

CHAPTER 1: INTRODUCTION

Question Answering (QA) systems have emerged as powerful platforms for automatically answering questions asked by humans in natural language using either a pre-structured database or a collection of natural language documents. Building intelligent agents with machine reading comprehension (MRC) or open-domain question answering (QA) capabilities using real world data is an important goal of artificial intelligence. Progress in developing these capabilities can be of significant consumer value if employed in automated assistants e.g., Cortana, Siri, Alexa, or Google Assistanton mobile devices and smart speakers, such as Amazon Echo. The rising popularity of spoken interfaces makes it more attractive for users to use natural language dialog for question answering and information retrieval from the web as opposed to viewing traditional search result pages on a web browser. All of these scenarios can benefit from fundamental improvements in QA models.

Question Answering systems can be defined by the paradigm each one implements: Information Retrieval QA- Usage of search engines to retrieve answers and then apply filters and ranking on the recovered passage, Natural Language Processing QA: Usage of linguistic intuitions and machine learning methods to extract answers from retrieved snippet, Knowledge Base QA: Find answers from structured data source (a knowledge base) instead of unstructured text. Standard database queries are used in replacement of word-based searches and a hybrid QA system which is the combination of IR QA, NLP QA and KB QA.

Reading Comprehension (RC), or the ability to read text and then answer questions about it, is a challenging task for machines, requiring both understanding of natural language and knowledge about the world. Successful reading comprehension systems should be able to learn good representations from raw text, infer and reason over learned representations, and finally generate a summarized response that is correct in both form and content. This thesis will attempt to explore this challenging task of question answering by developing an NLP QA system and a Knowledge Base QA system and highlights the challenges and shortcomings from each of the models.

1.1 THESIS STATEMENT

In this thesis we explore Question Answering from two perspectives. First , we deal with the complexity associated with the domain of stories and movie scripts by developing models for the Narrative QA Dataset [1], recently released by the researchers at Google DeepMind. This task allows us to explore the different types of complicated reasoning required to answer

questions from text and unearth some of the shortcomings of models proposed in literature for similar challenges.

Second, we deal with a novel task of incorporating common sense knowledge, extracted from external structured databases to augment the reasoning process during automated question answering. We develop models for the McScript dataset [2] recently proposed as a SemEval Task and discover the challenges machine faces in making inferences from text that come pretty intuitively to humans.

1.2 CONTRIBUTIONS

Multi Task Learning For Narrative Question Answering : In Chapter 3 we will explore the novel model proposed by us to perform reasoning over long texts and generate answers that are correct and human understandable. We also present ablation studies of the model and future improvements that we plan to implement.

Incorporating Common Sense Knowledge for Question Answering : In Chapter 4 we explore the proposed model for MCScript dataset and some of the novel methods introduced by us to efficiently incorporate external knowledge from structured databases to augment the model's learning process.

CHAPTER 2: QUESTION ANSWERING

Enabling a computer to understand a document so that it can answer comprehension questions is a central, yet unsolved goal of NLP. A key factor impeding its solution by machine learned systems is the limited availability of human-annotated data. In this chapter we would go over some of the most popular datasets for question answering, the challenges they present to the models and their respective shortcomings. We would also cover some of the best performing models and the key contributions to their high performance.

Generally, the input to a question answering system consists of three parts: The text p containing the answer to the question, the question q and the answer a. To test the model's text understanding ability the problem is presented in two ways, first where the answer is a given span of words in the text and second where the actual answer is a human written answer. The first version of the problem is simpler since it does not depend upon the model's ability to generate coherent and grammatically correct answers. It only focuses on the ability to find the correct subset of words in text. Even though this problem is not representative of the real world problem but it acts as a good proxy for developing models , and is currently the major focus of most models present in literature.

Below we present some of the different question-answer pairs that test the model's natural language understanding abilities in various ways.

Multiple Supporting Facts : These questions require the model to chain multiple supporting facts , potentially amongst a set of other irrelevant facts, to find the answer.

QUESTION: Where was the apple before the kitchen?
EVIDENCE: John picked up the apple. John went to the office. John went to the kitchen. John dropped the apple.
ANSWER: office

Time Reasoning : These questions test the model's ability to understand time expressions.

QUESTION: Where did Julie go after the park?
EVIDENCE: In the afternoon Julie went to the park. Yesterday Julie was at school. Julie went to the cinema this evening.
ANSWER: cinema

Compound Coreference resolution : These tests refer to multiple subjects present in a single sentence and the model's ability to perform coreference resolution efficiently.

QUESTION: Where is Daniel?
EVIDENCE: Daniel and Sandra journeyed to the office. Then they went to the garden. Sandra and John travelled to the kitchen. After that they moved to the hallway.
ANSWER: garden
The above examples represent some of the challenging reasoning capabilities that the models should posses to be able to successfully answer these questions. In the following section we will review some of the datasets that were presented in literature and claim that they cover most of the basic reasoning skills which are possessed by humans.

2.1 CORPORA

The public availability of large datasets has been instrumental in many AI research breakthroughs. For example, ImageNets [3] release of 1.5 million labeled examples with 1000 object categories led to the development of object classification models that perform better than humans on the ImageNet task. Similarly Penn TreeBank for syntactic parsing and large scale speech recognition databases have enabled breakthroughs by deep learning models. Several MRC and QA datasets have also recently emerged. Existing datasets (published before those mentioned here) for RC have one of two shortcomings: (i) those that are high in quality[4] are too small for training modern data-intensive models, while (ii) those that are large [5][6] are semi-synthetic and do not share the same characteristics as explicit reading comprehension questions. In this study we summarize some of the recent datasets which overcome a lot of these shortcomings.

The Stanford Question Answering Dataset (SQuAD) [7]: consists of 107,785 question-answer pairs from 536 articles. A large number of a questions and answers are provided for a set of documents, where the answers are spans of the context document i.e. contiguous sequences of words from the document. Although the answers are not just single word/entity answers, many plausible questions for assessing RC cannot be asked because no document span would contain its answer.

MS MARCO [8]: is a dataset which contains 1,010,916 questions, 8,841,823 companion passages extracted from 3,563,535 web documents, and 182,669 editorially generated answers. This a large scale real world dataset. It comprises of anonymized search queries issued through Bing or Cortana. Corresponding to each question, they provide a set of extracted passages from documents retrieved by Bing in response to the question. The passages and the documents may or may not actually contain the necessary information to answer the question. For each question, crowd-sourced editors are required to generate answers based

on the information contained in the retrieved passages.

Narrative QA [1]: This dataset contains questions created by editors based on summaries of movie scripts and books. The dataset contains about 45,000 question-answer pairs over 1,567 stories, evenly split between books and movie scripts. This dataset offers tasks based on both summaries as well as entire novels/movie scripts.The full version of NarrativeQA requires reading and understanding entire stories (i.e., books and movie scripts). This task as the authors argue is at present intractable for existing neural models out of the box. This is also one of the datasets for which we develop models in this thesis. Fig 2.1 presents an example from this dataset.

Title: Ghostbusters II
Question: How is Oscar related to Dana?
Answer: her son
Summary snippet: ...Peter's former girlfriend Dana Barrett has had a son, Oscar...
Story snippet:
DANA (setting the wheel brakes on the buggy)
Thank you, Frank. I'll get the hang of this eventually.

She continues digging in her purse while Frank leans over the buggy and makes funny faces at the baby, OSCAR, a very cute nine-month old boy.

FRANK (to the baby)
Hiya, Oscar. What do you say, slugger?

FRANK (to Dana)
That's a good-looking kid you got there, Ms. Barrett.

Figure 2.1: Example of question answer pair from Narrative QA and associated summary and story snippets

MCScript Dataset [2]: This dataset presented as part of the SemEval Task 2018, assesses how the inclusion of commonsense knowledge benefits natural language understanding systems. In particular, authors focus on commonsense knowledge about everyday activities, referred to as scripts. Scripts are sequences of events describing stereotypical human activities (also called scenarios), for example baking a cake, taking a bus, etc. Many times context may be absent or may lack sufficient information to resolve the ambiguity. In such cases, authors hypothesize it would be beneficial to include commonsense knowledge about the world in an NLU system. It comprises of 2,119 texts and a total of 13,939 questions. The texts in the data set talk about everyday activities and cover 110 script scenarios of differing complexity. 27.4% of the questions require commonsense inference about everyday activities. Following are some examples from this dataset that highlight its novelty:

TEXT: My backyard was looking a little empty, so I decided I would plant something. I went out and bought tree seeds. I found a spot in my yard that looked like it would get enough sunshine. There, I dug a hole for the seeds. Once that was done, I took my watering can and watered the seeds.

QUESTION/ANSWER: Why was the tree planted in that spot?

1. to get enough sunshine
2. there was no other space

While the above question should be easily answerable from text, the following question answer pair requires common sense knowledge, which can be incorporated into the model in various ways and is explored in further depth as part of this thesis in later sections.

QUESTION/ANSWER: What was used to dig the hole?

1. a shovel
2. their bare hands

This is one of the first few large scale datasets that help tackle the challenge of augmenting model's reasoning capabilities with common sense knowledge and hence is a novel task.

2.2 COMPARISON OF DATASETS

Ideal datasets should posses the following characteristics for model development:

1. Large enough to allow training of large deep learning models.
2. Contain questions which are not about the surface form of the text, but rather about the underlying narrative,which should require the formation of more abstract representations about the events and relations expressed in the course of the document.Answering such questions requires that readers integrate information which may be distributed across several statements throughout the document, and generate a cogent answer on the basis of this integrated information. That is, they test that the reader comprehends language, not just that it can pattern match.
3. Question answer pairs should be more representative of a natural distribution of information need that users may want to satisfy using, say, an intelligent assistant.

Table 2.1 highlights some of the salient features of above mentioned datasets:

Table 2.1: Comparison of Datasets

Dataset	Documents	Questions	Answers
MCTest [4]	660 short stories, grade school level	2640 human generated	multiple choice
CNN/Daily Mail [5]	93K+220K news articles	387K+997K Cloze-form, based on highlights	entities
Childrens Book Test (CBT) [6]	687K of 20 sentence passages from 108 childrens books	Cloze-form, from the 21st sentence	multiple choice
SQuAD [7]	23K paragraphs from 536 Wikipedia articles	108K human generated, based on the paragraphs	spans
MS MARCO [8]	1M passages from 200K+ documents retrieved using the queries	100K search queries	human generated, based on the passages
NarrativeQA [1]	1,572 stories (books, movie scripts) and human generated summaries	46,765 human generated, based on summaries	human generated, based on summaries
MCScript [2]	2,119 texts based on everyday activities	13,939 questions	multiple choice,require common sense knowledge

As we can see from the Table 2.1 that the three datasets SQUAD, MS MARCO and Narrative QA require answers to be more than single word/ entities and offer a more challenging task than simple answer ranking problem. Among, these three datasets SQUAD provides a large number of questions. However, these are from a relatively small number of documents, which are themselves fairly short, thereby limiting the lexical and topical diversity of models trained on this data can cope with. Simple models scoring and/or extracting candidate spans conditioned on the question and superficial signal from the rest of the document do well on both SQUAD and MS MARCO [9].These models will not trivially generalize to problems where the answers are not spans in the document or several discontinuous spans are needed to generate a correct answer. Baselines provided in NarrativeQA for summary based questions have the span prediction models to be the best performing too. However there still remains a huge gap between the human generated answers and predicted spans. Moreover,

fictional stories in Narrative QA have number of advantages as a domain as they are largely self-contained, summaries contain more complex relationships and timelines than news articles or short paragraphs from the web and thus provide a task different in nature. However, real-world text is messy: they may include typos or abbreviations and transcription errors in case of spoken interfaces. Both SQUAD and NarrativeQA, contain high-quality stories or text spans from sources such as Wikipedia. Real-world machine reading comprehension systems should be bench-marked on realistic datasets where they need to be robust to noisy and problematic inputs. MS MARCO alleviates that issue since the questions correspond to actual search queries that users submitted to Bing. Also, none of these datasets specifically assess the commonsense reasoning ability of a model, which is crucial when the relevant context for a question might not be explicitly present in text and requires drawing inferences from external knowledge bases. This limitation is overcome by the MCScript Dataset.

Thus, it can be seen that all the datasets have their own limitations and their still remains scope to develop more datasets that would enable the modelling of real world question answering systems.

2.3 PREVIOUS APPROACHES

End to end neural network architectures have proven to perform best on the reading comprehension datasets presented in the previous section. The general structure of these networks can be broken down into three steps as follows:

Encoding: First, all the words in the corpus are mapped to d- dimensional vectors via an embedding lookup matrix. Next these vectors are usually passed through a bidirectional RNN encoder to obtain contextual embeddings for the words in the paragraph as well as those in the question. The encoder could be GRU or LSTM based, the choice being made usually to reduce the computational complexity of the models since the performance of the two cells is usually comparable.

Attention: In this step, the goal is to compare the question embedding and all the contextual embeddings, and select the pieces of information that are relevant to the question. This is the step where most research has been conducted and brings about the biggest differences in performance.

End-to-end machine comprehension using attention mechanisms can usually be broken down into three distinct groups. The first group uses a dynamic attention mechanism (first proposed for the task of machine translation [10]), in which the attention weights are updated dynamically given the query and the context as well as the previous attention. Herman et al.[5] argue that the dynamic attention model performs better than using a single fixed

query vector to attend on context words on CNN and DailyMail datasets. Chen et al. [11] show that simply using bilinear term for computing the attention weights in the same model drastically improves the accuracy. Their model performed well on several of the datasets shown above.

The second group computes the attention weights once, which are then fed into an output layer for final prediction [12][13]. The third group (considered as variants of Memory Network [14]) repeats computing an attention vector between the query and the context through multiple layers, typically referred to as multi-hop.

Decoding: This layer usually depends upon the task at hand and often involves applying a softmax over the entire vocabulary to convert the outputs to a probability distribution. The tasks usually vary from cloze style one word prediction to span based contiguous prediction of words to the use of RNN based decoders for generating answers. This stage often involves some kind of optimizations such as use of hierarchical softmax layers and beam based decoding to reduce the computation time or obtain high probability sequences, which we will explore in further detail in the sections below.

The following sections explore each of these stages in further detail and will do a deep dive of some of the innovative modelling techniques that have been proposed for each of them to design robust question answering systems.

2.3.1 Embeddings

Words in natural language follow a Zipfian distribution whereby some words are frequent but most are rare. Learning representations for words in the long tail of this distribution requires enormous amounts of data. Representations of rare words trained directly on end tasks are usually poor and fail to generalize well, requiring the use of pre-trained embeddings on external data, or treat all rare words as out-of-vocabulary words with a unique representation. The two most popular methods for inducing word embeddings from text corpora are GloVe [15] and word2vec [16]. These packages also provide off-the shelf (OTS) embeddings trained on large corpora.

Dhingra et al. [17] show that minor choices such as the use of which kind of pretrained word embeddings and techniques for handling out of vocabulary words at test time can lead to substantial differences the final performance of the model. These differences are usually much larger than the gains reported due to architectural improvements.

The typical remedy to the rare word problem is to learn embeddings for some proportion of the head of the distribution, possibly shifted towards the domain-specific vocabulary of the dataset or task at hand, and to treat all other words as out-of-vocabulary (OOV), replacing

them with an unknown word UNK token with a shared embedding. This essentially heuristic solution is inelegant, as words from technical domains, names of people, places, institutions, and so on will lack a specific representation unless sufficient data are available to justify their inclusion in the vocabulary. This forces model designers to rely on overly large vocabularies, which are parametrically expensive, or to employ vocabulary selection strategies.

Badhanau et al. [18] proposes a method for computing embeddings on the fly, which jointly addresses the large vocabulary problem and the paucity of data for learning representations in the long tail of the Zipfian distribution. They train a network to predict the representations of words based on auxiliary data. Such auxiliary data need only satisfy the general requirement that it describe some aspect of the semantics of the word for which a representation is needed. Examples of such data could be dictionary definitions, Wikipedia infoboxes or knowledge bases such as Wordnet. The auxiliary data encoders are trained jointly with the objective, ensuring the preservation of semantic alignment with representations of within-vocabulary words. Quantitative results show that auxiliary data improves performance of models proposed for the SQUAD dataset. Qualitative evaluation indicates that their method allows models to draw and exploit connections defined in auxiliary data, along the lines of synonymy and semantic relatedness.

2.3.2 Attention Mechanisms

As mentioned before attention is a process of extracting the relevant context form the reading comprehension which can be used for answering the given question. We will now explore an architecture from each of the three classes of attention.

Dynamic Attention [11]: Authors in this paper propose a simple modification to the original attention mechanism proposed in [10] for machine translation. They show that simply using bilinear term for computing the attention weights in the same model drastically improves the accuracy on the CNN/Daily Mail datasets. The attention weights for the context words are computed as follows:

$$\alpha_i = \text{softmax}_i \mathbf{q}^T \mathbf{W} p_i \tag{2.1}$$

using which the final context vector becomes,

$$c = \sum_i \alpha_i p_i \tag{2.2}$$

where $\mathbf{W} \epsilon \mathbf{R}^{h \times h}$ represents the bilinear term and allows computing attention between ques-

tion and context words much more flexibly. The authors report an improvement of 7-10% over the original architecture.

Attention Sum Reader [12] : This architecture is designed for cloze style RC tasks where the answer is one of the omitted tokens from the context. They use two bidirectional GRU encoders to encode the document and the question.Then they compute a weight for every word in the document as the dot product of its contextual embedding and the query embedding. This weight might be viewed as an attention over the document d. To form a proper probability distribution over the words in the document, they normalize the weights using the softmax function, as follows:

$$s_i \propto exp(d_i q_i) \tag{2.3}$$

then the probability of a token being an answer is simply the sum of these products,

$$P(w_i|\mathbf{q}, \mathbf{d}) = \sum s_i \tag{2.4}$$

No weighted sum is taken to compute a final document embedding i.e no blending of vector representation. Since the summation of attention in their model inherently favours frequently occurring tokens, authors visualize how the accuracy depends on the frequency of the correct answer in the document.They show that the accuracy significantly drops as the correct answer gets less and less frequent in the document compared to other candidate answers. Also accuracy drops with increasing document length and greater number of candidate answers.

Attention is all you Need [19] : This paper has has not yet been applied for question answering task, however the authors in this paper show that multiple blocks of attention could be stacked together to replace the sequence to sequence models that we have seen above. The major motivation for this paper was that RNN units are sequential in nature and do not allow parallelization across training examples which becomes critical at longer sequence lengths, as memory constraints limit batching across examples.

The basic attention mechanism employed here is a simple dot product between the queries and keys followed by a softmax. Authors argue that dot-product attention is much faster and more space-efficient in practice than additive attention [10], since it can be implemented using highly optimized matrix multiplication code.

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{2.5}$$

Parallelization is achieved by projecting each of the queries, keys and values to smaller

dimension components and then computing the attention for each component in parallel, the results for which are concatenated back to original dimension values. The authors call this Multi-Head Attention.

$$\text{MultiHead}(Q, K, V) = Concat(head_1, ...head_h)W^O \tag{2.6}$$

Attention is used in this architecture in three different ways: To compute the attention between encoder decoder layers where the query comes from the decoder layer and values and keys are the output of the encoder layers. And in the self attention layers of the encoder and decoder where all the queries, keys and values come from the same place.

Self Attention: Authors perform a detailed comparison between the performance of the self attention layers and use of RNN/CNN for encoding and decoding. They argue that self attention can compute long range dependencies between any two position in the input/output in a constant number of operations achieved through their multi-head attention architecture. The same operation is proportional to sequence length when RNN s are used. CNNs are generally more expensive than RNNs. Moreover self attention has the added side benefit for providing more interpretable models.

2.3.3 Decoding

Decoding layer depends on the kind of output desired for the particular task. If the task is span prediction then models usually just try to predict the start and the end indices of the answer span, such as in SQUAD, whereas if the answer needs to be generated from the words in the entire vocabulary then usually an RNN based decoder is employed. Here, we will explore some different decoder architectures proposed in literature to combat some of the issues faced by RNNs.

Pointer Networks [20]: RNN based decoding methods require the size of the output dictionary to be fixed a priori. Because of this constraint we either have to choose a very huge vocabulary to be able to generalize the model performance, which slows down training since softmax needs to be applied over this entire vocabulary at every decoding time step or we have to resort to methods such as hierarchical softmax etc. This paper addresses this limitation by re purposing the attention mechanism of [10] to create pointers to input elements. It is suitable for problems where output vocabulary depends upon the input sequence and could be used to handle the out of vocabulary issue at test time. Using the

attention mechanism in [10] the decoding step becomes,

$$u_j^i = v^T tanh(W_1 e_j + W_2 d_i) \tag{2.7}$$

$$P(w_i|w_1...W_i - 1) = softmax(u^i) \tag{2.8}$$

where softmax normalizes the vector u_i (of length n) to be an output distribution over the dictionary of inputs, and v, W1, and W2 are learnable parameters of the output model. Here, they do not blend the encoder state e_j to propagate extra information to the decoder, but instead, use u_j^i as pointers to the input elements.

Intra-Decoder Attention: A decoder can generate repeated phrases based on its own hidden states, especially when generating long sequences. To prevent that, we can incorporate more information about the previously decoded sequence into the decoder. Looking back at previous decoding steps will allow a model to make more structured predictions and avoid repeating the same information, even if that information was generated many steps away. To achieve this authors in [21] introduce intra-decoder attention. For each decoding step t, their model computes a new decoder context vector c_t^d, for $t > 1$ as follows:

$$e_{tt'} = h_t^T W_{attn} h_{t'} \tag{2.9}$$

$$\alpha_{tt'} = \frac{exp(e_{tt'})}{\sum_{j=1}^{t-1} exp(e_{tj})} \tag{2.10}$$

$$c_t = \sum_{j=1}^{t-1} \alpha_{tj} h_j \tag{2.11}$$

Hybrid learning objective : The most widely used method to train a decoder RNN for sequence generation, called the teacher forcing algorithm , minimizes a maximum-likelihood loss at each decoding step. The maximum-likelihood training objective is the minimization of the following loss,

$$L_{ml} = -\sum_{t=1}^{n} logp(y_t|y_1....t_{t-1}, x) \tag{2.12}$$

However, minimizing L_{ml} does not always produce the best results on discrete evaluation metrics such as ROUGE . There are two main reasons for this discrepancy. The first one, called exposure bias, comes from the fact that the network has knowledge of the ground truth sequence up to the next token during training but does not have such supervision when testing, hence accumulating errors as it predicts the sequence. The second reason is due to the large number of potentially valid output sequences. The ROUGE metrics take

some of this flexibility into account, but the maximum-likelihood objective does not.

One way to remedy this is to learn a policy that maximizes a specific discrete metric instead of minimizing the maximum-likelihood loss, which is made possible with reinforcement learning. In [21], authors use a self-critical policy gradient training algorithm. For this training algorithm, model produces two separate output sequences at each training iteration: y_s, which is obtained by sampling from the $p(y_t^s|y_1^s....y_{t-1}^s)$ probability distribution at each decoding time step, and y_b, the baseline output, obtained by maximizing the output probability distribution at each time step, essentially performing a greedy search. Authors define r(y) as the reward function for an output sequence y, comparing it with the ground truth sequence y^* with the evaluation metric of choice (ROUGE,BLEU etc).

$$L_{rl} = (r(y_b) - r(y^s)) \sum_{t=1}^{n} logp(y_t^s|y_1^s....y_{t-1}^s) \tag{2.13}$$

Minimizing L_{rl} is equivalent to maximizing the conditional likelihood of the sampled sequence y^s if it obtains a higher reward than the baseline y_b, thus increasing the reward expectation of the model.

One potential issue of this reinforcement training objective is that optimizing for a specific discrete metric like ROUGE does not guarantee an increase in quality and readability of the output. It is possible to game such discrete metrics and increase their score without an actual increase in readability or relevance. Therefore, to tackle this issue authors propose a mixed learning objective as follows,

$$L_{mixed} = \gamma L_{rl} + (1 - \gamma) L_{ml} \tag{2.14}$$

Authors hypothesize that L_{ml} can assist policy learning algorithm to generate more natural output sequences. The paper shows that this hybrid learning objective achieves the highest discrete metric scores for a given task as compared to only RL or ML learning objectives. Moreover the RL+ML loss function also leads to more readable and natural output sequences.

2.3.4 Incorporating Common Sense Knowledge

Most of the approaches incorporating common sense knowledge include the models developed for the SemEval Task 2018. These include neural and non-neural approaches. Non-neural approaches such as IUCM [22] applied an unsupervised approach that assigns the correct answer to a question based on text overlap. Text overlap is computed based on the

given passage and text sources of the same topic. Different clustering and topic modeling techniques are used to identify such text sources in MCScript and DeScript. Neural based approaches employ different variants of attention mechanism discussed in the previous section. Some of these models also include ensemble of LSTMs with attention mechanisms and logistic regression model using patterns based on the vocabulary of the training set. There also has been significant experimentation with different embeddings such as Glove, Word2vec, Numberbatch etc.

Another popular approach recently proposed by Bishan et al. [23] tackle an important challenge of incorporating external knowledge relevant to the textual context. In general KBs involve polysemy, such as "Washington" can refer both a person or a place and if the right meaning is not captured, can mislead the model. At each time step, the model retrieves KB concepts that are potentially related to the current word. Then, an attention mechanism is employed to dynamically model their semantic relevance to the reading context. Furthermore, authors introduce a sentinel component in BiLSTMs that allows flexibility in deciding whether to attend to background knowledge or not. This is crucial because in some cases the text context should override the context-independent background knowledge available in general KBs. Experimental results show that their model achieves accuracies that surpass the previous state-of-the-art results for both entity extraction and event extraction on the widely used ACE2005 dataset [24].

CHAPTER 3: NARRATIVE QUESTION ANSWERING

Reading comprehension in contrast to information retrieval, requires integrating information and reasoning about events, entities, and their relations across a full document. The dataset should contain questions that are not superficial and cannot be answered by surface pattern matching. NarrativeQA dataset tackles this challenge by presenting text from stories and movie scripts. In this chapter we will explore some of the intricacies of this dataset and the deep learning based models developed by us to solve the challenge.

3.1 NARRATIVE QUESTION ANSWERING DATASET

The dataset consists of stories, which are books and movie scripts, with human written questions and answers based solely on human-generated abstractive summaries. Books were collected from Project Gutenberg and movie scripts scraped from the web. They matched the stories with plot summaries from Wikipedia using titles and verified the matching with help from human annotators. In this way authors obtained 1,567 stories. This provides with a smaller set of documents, compared to the other datasets mentioned in the previous section, but the documents are long which provides us with good lexical coverage and diversity. Each story is associated with 30 question answer pairs, leading to 45,765 human generated questions and answers in total. The questions are grammatical questions written by human annotators, average 9.8 tokens in length, and are mostly formed as WH-questions. Answers in the dataset are human written, short, averaging 4.73 tokens, but not restricted to spans from the documents. There are 44.05% and 29.57% answers that appear as spans of the summaries and the stories, respectively. Fig 3.1 represents some of the statistics of the questions available in this dataset.

First token	Frequency
What	38.04%
Who	23.37%
Why	9.78%
How	8.85%
Where	7.53%
Which	2.21%
How many/much	1.80%
When	1.67%
In	1.19%
OTHER	5.57%

(a) Frequency of question types in training set

Category	Frequency
Person	30.54%
Description	24.50%
Location	9.73%
Why/reason	9.40%
How/method	8.05%
Event	4.36%
Entity	4.03%
Object	3.36%
Numeric	3.02%
Duration	1.68%
Relation	1.34%

(b) Question categories in training set

Figure 3.1: Summary statistics for NarrativeQA Dataset

In the remainder of this chapter, we describe the data preprocessing methodology we used, the modelling techniques employed and experimental results on the summary-reading task.

3.2 DATA PREPARATION

3.2.1 Entity Anonymization

The provided narratives contain a large number of named entities (such as names of characters or places). Inspired by Hermann et al. [5], we replace such entities with markers, such as @entity12. These markers are permuted during training and testing so that none of their embeddings learn a specific entity's representation. This allows us to build representations for entities from stories that were never seen in training, since they are given a specific identifier (to differentiate them from other entities in the document) from a set of generic identifiers re-used across documents. We first perform coreference resolution on the given text and corresponding question and answer pairs together, using the Stanford CoreNLP parser and then we replace entities with the random entity markers.

In one of our experiments we also include the named entity tag in the marker such as @entity_PERSON_42, however this does not provide any performance improvements and also leads to the addition of more unique combinations of entity marker to the vocabulary, thus we do not proceed further with that approach. Table 3.1 demonstrates an example of this preprocessing step.

Table 3.1: Entity Anonymization

Original Version	Anonymized Version
Mark Hunter (Slater), a high school student in a sleepy suburb of Phoenix, Arizona, starts an FM pirate radio station that broadcasts from the basement of his parents' house. Mark is a loner, an outsider, whose only outlet for his teenage angst and aggression is his unauthorized radio station .	**entity_12** (**entity_10**), a high school student in a sleepy suburb of **entity_4** , **entity_18** , starts an FM pirate radio station that broadcasts from the basement of his parents house . **entity_12** is a loner , an outsider , whose only outlet for his teenage angst and aggression is his unauthorized radio station
Who is Mark Hunter ?	Who is **entity_12** ?
He is a high school student in Phoenix.	He is a high school student in **entity_4** .

Clearly a human reader can answer both queries correctly. However in the anonymised setup the context document is required for answering the query, whereas the original version

could also be answered by someone with the requisite background knowledge. Therefore, following this procedure, the only remaining strategy for answering questions is to do so by exploiting the context presented with each question. Thus performance on NarrativeQA corpora truly measures reading comprehension capability.

3.2.2 Determine Answer Spans

In our multi task learning model , one arm of the model is trained by predicting answer start and end indices of spans in text which correspond closest to the actual answer and the other arm corresponds to the decoder which generates the answer word by word. The reason behind this approach is two fold: First, neural span prediction models are the best performing baselines for the NarrativeQA dataset, but there still remains significant room for improvement which we expect is due to the presence of questions whose answers are not direct spans in the text. Second, we employ the Bidirectional Attention Flow module[9] in our model, which was initially used for SQUAD dataset, a span prediction task. Hence, we speculate that training of this attention module will be benefited if use direct spans from texts rather than human generated answers. We use start and end indices of the span achieving the highest Rouge-L score with respect to the reference answers as labels on the training set. The model is then trained to predict these spans by maximizing the probability of the indices. Rouge-L measures longest matching sequence of words using longest common subsequence (LCS). An advantage of using LCS is that it does not require consecutive matches but in-sequence matches that reflect sentence level word order. Since it automatically includes longest in-sequence common n-grams, you dont need a predefined n-gram length. We also experimented with BLEU-N scores, however ROUGE-L led to the best overall results, hence we employed that as our metric for generating answer labels. Following shows an example of the generated answer label:

QUESTION : Who is entity_12 ?
ACTUAL ANSWER : He is a high school student in entity_4.
ANSWER SPAN : a high school student in a sleepy suburb of entity_4

3.2.3 Miscellaneous Data Preprocessing

In addition to the above steps for data preparation we also used other standard NLP methods for preprocessing the given dataset, before feeding into the model. This includes :

- Tokenization of the summary, question and answer pairs using the Stanford CoreNLP parser.
- Modify the actual answers to include the start and the end symbols. This is important for the RNN based decoder during training, as it needs it to understand when to stop generating the answer and also provides more flexibility in decoding rather than using a fixed length for all answers.
- We limit the our training vocabulary size to around 33k tokens. Words which appear with a frequency of less than 10, are not part of the vocabulary. It is important to keep the size of the vocabulary small, else it slows down the training process for the model since we are required to take a softmax over the entire vocabulary each time the decoder generates the answer.
- We create batches by first sorting the summaries according to their lengths. This reduces the per epoch time by over 50%, since it reduces the amount of padding required to keep all the examples in the batch of the same size, thereby reducing the amount of unrolling that the RNN based encoder needs to perform.
- We filter summaries that are larger than 900 tokens and questions longer than 20 tokens. This is done to avoid out of memory GPU issues and reduce the training time. Moreover, the performance of the LSTM deteriorates for very long summaries and is is another challenge of NarrativeQA dataset.

3.3 MODEL

In this section we provide the details of the proposed model for the summary task of NarrativeQA, results and the ablation study for our model. Fig 3.2 shows the schematic of the model used for this dataset.

Word Embedding Layer: We map each word in the context and question into a vector space using pre trained word embedding. For this model, we use 300 dimensional Glove embedding. Experiments were conducted using Word2Vec emebeddings too, however there was no significant difference in performance. Moreover, we also tried to train our own embedding but noticed that the NarrativeQA dataset was not large enough and hence it soon lead to overfitting, thereby decreasing the performance.

Contextual Embedding Layer: Summary and question word vectors are passed through a bi-directional LSTM layer, before feeding into the attention module. This allows us to

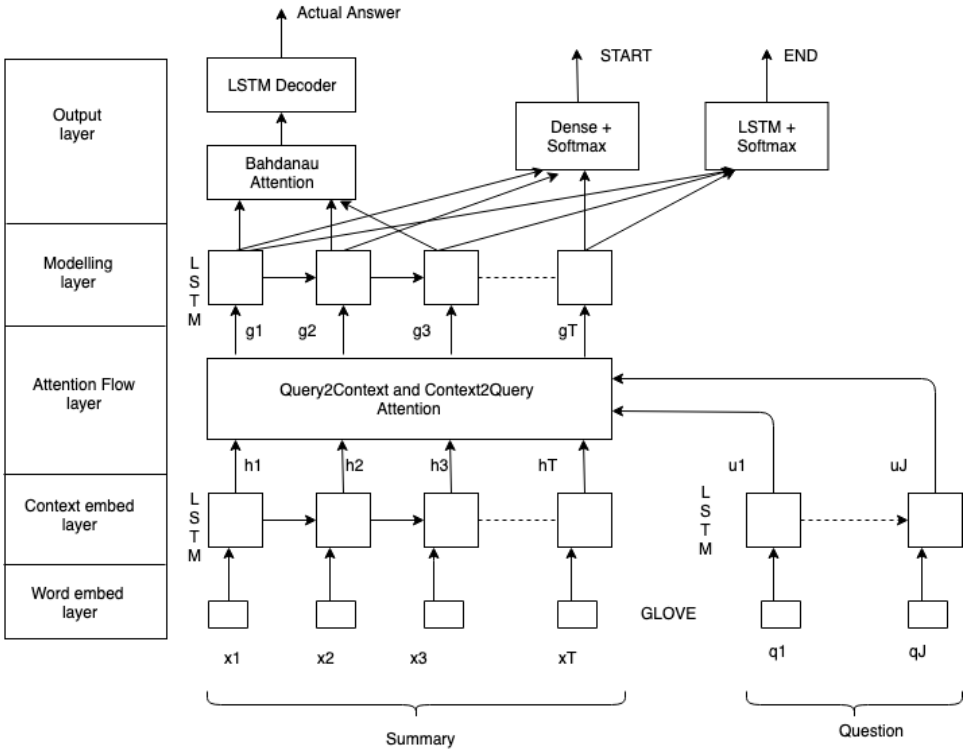

Figure 3.2: Model for Narrative QA dataset

modify the word embedding to capture the context from the surrounding words and encode information from them into these vectors.

Attention Layer: The attention layer for the model is adopted from the Bidirectional Attention Flow Module by Seo et al. [9]. BiDAF attention mechanism offers following improvements to the previously presented attention paradigms. First, their attention layer is not used to summarize the context paragraph into a fixed-size vector. Instead, the attention is computed for every time step, and the attended vector at each time step, along with the representations from previous layers, is allowed to flow through to the subsequent modeling layer. This reduces the information loss caused by early summarization. Second, authors use a memory-less attention mechanism. That is, while they iteratively compute attention through time as in [10], the attention at each time step is a function of only the query and the context paragraph at the current time step and does not directly depend on the attention at the previous time step. The output of the attention layer is fed to a Bi-RNN (modelling layer) which computes the interaction between the embeddings of the attention layer. The authors hypothesize that this leads to division of labor as the attention layer is supposed to compute the interaction between the context and query for a single time step, and relations between time steps is computed by modelling layer. Third, they use attention

mechanisms in both directions, query-to-context and context-to-query, which provide complimentary information to each other, which has been shown to beneficial in visual QA [25]. Moreover, their model never summarizes any information computed at each layer, instead all the computed representations always flow till the last layer.

The model computes attention in two directions : Context to query and query to context, which are derived from a similarity matrix computed based on the contextual **H** and query **Q** embeddings as follows,

$$S_{tj} = \alpha(\mathbf{H}_{:t}, \mathbf{Q}_{:j}) \tag{3.1}$$

where α is a trainable function. The context-query and query-context attentions are computed as follows,

Context-to-Query:

$$\mathbf{a}_t = \text{softmax}(S_{t:}) \tag{3.2}$$

$$\mathbf{U}_{:t} = \sum_j \mathbf{a}_{tj} \mathbf{U}_{:j} \tag{3.3}$$

Query-to-Context:

$$\mathbf{b} = \text{softmax}(max_{col}(S)) \tag{3.4}$$

$$h = \sum_t b_t \mathbf{H}_{:t} \tag{3.5}$$

Finally the contextual embeddings and the attention vectors are combined together to feed query aware representation of each context word into the modelling layer.

Modelling Layer: The output of the modeling layer captures the interaction among the context words conditioned on the query. This is different from the contextual embedding layer, which captures the interaction among context words independent of the query. We use a single layer of bi-directional LSTM, with the output size of d for each direction. Hence we obtain a matrix M , which is passed onto the output layer to predict the answer. Each column vector of M is expected to contain contextual information about the word with respect to the entire context paragraph and the query.

Output Layer: Since this is multi task learning our output layer consists of two parts, for predicting the answer spans and also generating the actual answers.

For answer span prediction, we predict the start and the end indices separately. The start index is derived from the following equation,

$$p^1 = softmax(W_{p^1}^T [G; M]) \tag{3.6}$$

and end index is obtained by passing the output of the modelling layer through another

LSTM before prediction,

$$p^2 = softmax(W_{p^2}^T[G; M^2]) \tag{3.7}$$

For generating the actual answers, we first compute attention over the output of the modelling layer to derive the context vector as in [10],

$$c_i = \sum_{j=1}^{T} \alpha_{ij} h_j \tag{3.8}$$

where α_i is computed as follows,

$$\alpha_{ij} = \frac{exp(e_{ij})}{\sum_{k=1}^{T} exp(e_{ik})} \tag{3.9}$$

$$e_{ij} = \beta(s_{i-1}, h_j) \tag{3.10}$$

At every time step we compute the similarity between the output of the LSTM decoder and the output of the modelling layer. This is used to obtain the context vector which is the weighted representation of the modelling layer's query and context aware word embeddings. The output of the decoder is then passed through a softmax layer over the entire vocabulary to predict the next answer token.

Loss Function: We define two loss functions for both arms of the output layer. For the span prediction output, the loss function is defined as the sum of the negative log probabilities of the true start and end indices by the predicted distributions, averaged over all examples. And for the decoding layer we use the softmax cross entropy loss for the predicted tokens. The model is trained such that for half of the iterations span prediction arm is active and for the rest the decoder arm is active. During validation phase we only use the decoder arm of the output layer to make predictions.

3.3.1 Implementation Details

The hidden state size d of the model is 100. We use the Adam optimizer, with a minibatch size of 20 and an initial learning rate of 0.001, for 20 epochs. A dropout rate of 0.2 is used for all LSTM layers, and the linear transformation before the softmax for the answers. During training, the moving averages of all weights of the model are maintained with the exponential decay rate of 0:999. At test time, the moving averages instead of the raw weights are used. The training process takes roughly 23 hours on a single Titan X GPU.

3.4 RESULTS

We present the results of our model along with the best performing baselines for the NarrativeQA dataset in the following table:

Table 3.2: Results on Summaries

Model	BLEU-1	BLEU-4
Seq2Seq (no context)	15.89	1.26
Attention Sum Reader	23.20	6.39
Span Prediction	33.72	15.53
Our Model	16.77	4.437
Human	44.43	19.65

As can be seen from Table 3.2 we are still not able to beat the strongest baseline , which is a span prediction model using the BIDAF attention module. One reason for this is that our model is too big (8 million trainable parameters) to be trained on the given corpus and we would need to resort to smaller models to be successfully able to train on this dataset and generate grammatically correct answers or pretrain our model on another daatset.

3.4.1 Error Analysis

Below we present some examples from the test set to demonstrate certain flaws of our model and aid in the debugging process.

SUMMARY : *The play begins with three pages disputing over the black cloak usually worn by the actor who delivers the prologue. They draw lots for the cloak, and one of the losers, Anaides, starts telling the audience what happens in the play to come; the others try to suppress him, interrupting him and putting their hands over his mouth. Soon they are fighting over the cloak and criticizing the author and the spectators as well. In the play proper, the goddess Diana, also called Cynthia, has ordained a "solemn revels" in the valley of Gargaphie in Greece. The gods Cupid and Mercury appear, and they too start to argue. Mercury has awakened Echo, who weeps for Narcissus, and states that a drink from Narcissus's spring causes the drinkers to "Grow dotingly enamored of themselves." The courtiers and ladies assembled for the Cynthia's revels all drink from the spring. Asotus, a foolish spendthrift who longs to become a courtier and a master of fashion and manners, also drinks from*

the spring; emboldened by vanity and self-love, he challenges all comers to a competition of "court compliment." The competition is held, in four phases, and the courtiers are beaten. Two symbolic masques are performed within the play for the assembled revelers. At their conclusion, Cynthia (representing Queen Elizabeth) has the dancers unmask and shows that vices have masqueraded as virtues. She sentences them to make reparation and to purify themselves by bathing in the spring at Mount Helicon..

QUESTION: What did the symbolic vices disguise themselves to be?
ANSWER: Virtues
PREDICTED ANSWER : Mount Helicon

QUESTION: Who enters with Mercury?
ANSWER: Cupid
PREDICTED ANSWER : Echo

From the above examples it can be seen that the model's attention is close to the correct answer in text but is not able to generate the correct answer token. We hypothesize that more data is needed to be able to train this model further and improve its accuracy. These kinds of errors are the biggest contributor to the model failures.

QUESTION: What name was Cynthia more famously known by?
ANSWER: The goddess Diana
PREDICTED ANSWER : Cynthia

QUESTION: What is another name for the Goddess Diana?
ANSWER: Cynthia
PREDICTED ANSWER : Cynthia

In the above question answer pairs, both are similar questions but the model answers it correctly only the second time. We suspect that the phrase "famously known" affects the model prediction and it ends up performing shallow pattern matching between the question and answer tokens thereby assigning "Cynthia" higher probability.

QUESTION: What does a drink from Narcissus's spring cause the drinker to do?
ANSWER: Fall in love with themselves
PREDICTED ANSWER : The drinkers to grow dotingly

In the above example, it can be seen that our model does generate a correct answer, however, since these are human written answers, the BLEU score for this example is still very low due to very little overlap. This shows that we might need other metrics to evaluate the performance of our question answering models, which do not rely solely on string overlap.

QUESTION: How many phases does the competition have?
ANSWER: four
PREDICTED ANSWER : four

QUESTION: What challenge does Asotus propose to all comers?
ANSWER: Court Compliment
PREDICTED ANSWER : court compliment competition

The above examples demonstrate the model's ability to successfully tackle some of the challenges of this dataset presented in the previous chapter such as time reasoning, multiple supporting facts etc.

3.4.2 Ablation Study

Table 3.3 presents results with different variants of our model and help us to further understand the effects of different components to the model prediction.

Table 3.3: Ablation Study

Model	BLEU-1	BLEU-4
w/o Span prediction	6.67	0.13
w/o Entity Anonimization	11.73	3.19
Own Word Embedding (w/o Glove)	11.2	2.87
BLEU Score for Answer Span	8.82	2.82
Additional BiLSTM in Modelling layer	12.31	4.43
Increased Dropout	10.13	2.89

It can be observed from the above table that if we remove the span prediction arm from our model, the BLEU scores drop drastically, which confirms our initial intuition that the

BIDAF attention module does require direct supervision from the given text rather than using just the human generated answers as training labels. Addition of another BiLSTM in the modelling layer also does not provide any performance gains and again starts to overfit pretty quickly.

Another, interesting observation is that using ROUGE-L for finding closest answer spans in text performs much better than using BLEU scores. This is because it automatically includes the longest in-sequence common n-grams and we do not need a predefined n-gram length. Also, we observe that the entity anonimization is an important data preprocessing step as it prevents overfitting and allows the model to learn general entity representations. Moreover, we also notice that if we train our own word embeddings, model tends to overfit pretty quickly which deteriorates the performance further.

3.5 CONCLUSION

In this chapter we presented the details of the model developed for the Narrative Question Answering dataset. We observed that this task posits several challenges such as, it deals with the domain of stories,movie scripts which has much more complex timelines and named entities present when compared to news and Wikipedia articles. Moreover, the training labels are human written answers rather than spans from summaries.

We employed a multi task learning approach for this dataset and observed that the models are too big, requiring significant compute resources and susceptible to overfitting since the dataset might not be large enough to allow these models to generalize well. We also observed that certain data preprocessing techniques such as entity anonimization,batch sorting can help improve the performance of the models significantly. However, we still need to improve our models further to perform well on this task, either by pretraining on similar datasets, employing different attention modules or incorporating external knowledge from structured databases. The last avenue is explored further in the context of MCScript dataset and is presented in the following chapter.

CHAPTER 4: MACHINE COMPREHENSION USING COMMON SENSE KNOWLEDGE

Natural language comes with its own complexity and inherent ambiguities. Ambiguities can occur, for example, at the level of word meaning, syntactic structure, or semantic interpretation. Traditionally, Natural Language Understanding (NLU) systems have resolved ambiguities using information from the textual context. However, many times context may be absent or may lack sufficient information to resolve the ambiguity. In such cases, it would be beneficial to include commonsense knowledge about the world in an NLU system. MCScript dataset introduced as part of the SemEval Task 2018 tackles this challenge by providing reading comprehension questions which require common sense knowledge to be answered correctly.

4.1 MCSCRIPT DATASET

This Dataset focuses on commonsense knowledge about everyday activities, referred to as scripts. Scripts are sequences of events describing stereotypical human activities (also called scenarios), for example baking a cake, taking a bus, etc. Factual knowledge is mentioned explicitly in texts from sources such as Wikipedia and news papers. On the contrary, script knowledge is often implicit in the texts as it is assumed to be known to the comprehender. Because of this implicitness, learning script knowledge from texts is very challenging and requires information from external knowledge bases such as ConceptNet etc.

The dataset comprises of 2,119 such texts and a total of 13,939 questions and cover 110 script scenarios of differing complexity. 27.4% require commonsense inference about everyday activities. Fig 4.1 provides a distribution of different type of questions in this dataset.

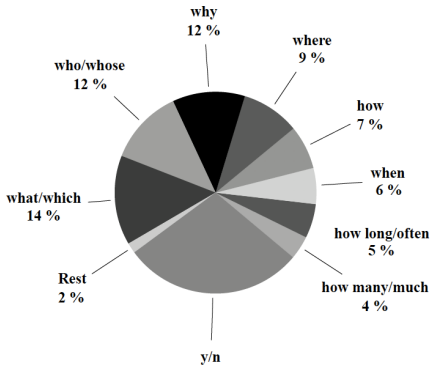

Figure 4.1: Distribution of Question Types in McScript Dataset

In the remainder of this chapter, we describe the data preprocessing methodology we used,the modelling techniques employed and experimental results on the summary-reading task.

4.2 DATA PREPARATION

For data preprocessing, we use spaCy for tokenization, part-of-speech tagging and named entity recognition. To explicitly model commonsense knowledge, relation embeddings based on ConceptNet [26] are used as additional features. ConceptNet is a large-scale graph of general knowledge from both crowd sourced resources and expert-created resources. It consists of over 21 million edges and 8 million nodes. ConceptNet shows state-of-the-art performance on tasks like word analogy and word relatedness. To extract relations from the ConceptNet we extract all the relations between words which are present in our training vocabulary and the weights of the relations is greater or equal to one. Some of the examples of relation and word pair triples extracted from Conceptnet is as follows:
Antonym: absence - presence
Used For: yard - storage
Synonym: youth - young_person

4.2.1 Data Augmentation

The main contribution of this thesis for this task is to explore methods of extracting information from Conceptnet and augmenting the model with it to enhance its accuracy. We experimented with multiple methods and the following procedure provided us with the best results.

- For every word in the question (excluding the stop words) we consider words in the corresponding text for which there exists an edge in the ConceptNet graph. For all these edges, we represent the relations found in the ConceptNet, as text, and append the summary with this text. Following is an example of this augmentation,

  Used For: bath - relaxing
  Text: a bath is used for relaxing.

- We filter out edges which have the relation "Related to" associated with them. This is done , since these edges do not provide any relevant information to the given context and also result in a decrease in the model accuracy. We suspect that this happens because on addition of such edges increases the length of the text considerably, thereby

preventing the model's attention module to focus on the most relevant parts of context for a given question. Some of the examples of such edges are as follows:

Related To: apologize - fault
Text: Apologize is related to fault.
This relation might be relevant but does not provide the model with any explicit knowledge such as a person might apologize after making a fault or why did the person apologize? This is the kind of common sense knowledge we need to provide our model which is absent for these edges.

- To compensate for the removal of the above edges, we make a second hop in the ConceptNet graph, associated with the paragraph word and try to find a node which might be related to the question word. Following is an example of this modification,

  Original text: fix is related to repair
  Modified text: fix is a synonym of repair
  The above modification helps the model understand the specific relation between the two words and make better inferences.

- In addition to the above steps , we also consider pairs of question word and summary bigrams. This is done since a significant percentage of the nodes in ConceptNet are bigrams and help provide relevant information for the given context. Following is an example of these relations.

  Has Last Sub event: bathe - dry_off
  Text: The last thing you do when you bathe is dry off.

- As a variant of the above rules, we also append text for every question separately i.e. we create separate copies of text for each question and append the passage with information that is only relevant for that question. This is done to avoid addition of any irrelevant information for a particular question and should provide further improvement in model accuracy.

4.3 MODEL

In this section we provide the details of the model employed for the McScript dataset, results and error analysis for our model. This model is inspired by the contributions made

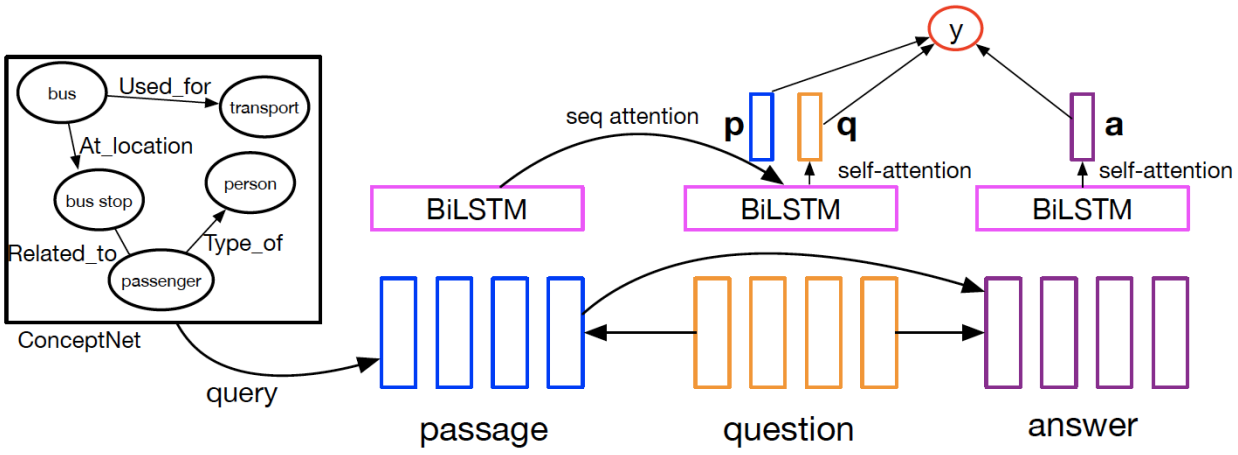

Figure 4.2: Model for McScript dataset

by YuanFundao et al. [27], which was one of the top performing models for this task. Fig 4.2 shows the schematic of the model used for this dataset.

Embedding Layer: We employ 300 dimensional word embeddings to convert the words of text and question answer pairs into vector space. We train our own embeddings for the part of speech tag, named entity and ConceptNet based relational embeddings. The relation is determined by querying ConceptNet and whether there is an edge between the paragraph word and any word in question or answer. If there exist multiple different relations, just randomly choose one.

The POS vocabulary size is 51 and 12 dimensional embeddings are trained for it, NER has a vocabulary of 20 and 8 dimensional embeddings are trained and there are 39 different relation types with an embedding size of 10. The final representation of the paragraph word is a concatenation of these vectors before it is input into the model as follows,

$$w_{P_i} = [E^{Glove}_{P_i}; E^{pos}_{P_i}; E^{ner}_{P_i}; E^{rel}_{P_i}; f_{P_i}] \tag{4.1}$$

where f_{P_i} is a co-occurrence binary feature, which is True if the word in the paragraph is present in the corresponding question or answer. Similar concatenation is also used to obtain the embeddings for the words in question and answer.

Attention Layer: We use word-level attention to model interactions between the given passage, the question and the answer. The attention mechanism is defined as follows,

$$Att_{seq}(\mathbf{u}, \{v_i\}_{i=1}^{n}) = \sum_{i=1}^{n} \alpha_i v_i \tag{4.2}$$

$$\alpha_i = softmax_i(f(\mathbf{W}_1\mathbf{u})^T f(\mathbf{W}_1\mathbf{v}_i)) \tag{4.3}$$

f is non linear activation function set to RelU. Question-aware passage representation$\{w^q_{P_i}\}^P_{i=1}$ can be calculated as:

$$w^q_{P_i} = Att_{seq}(E^{Glove}_{P_i}, \{E^{Glove}_{Q_i}\}^Q_{i=1}) \tag{4.4}$$

Similarly passage aware answer representation $\{w^p_{A_i}\}^a_{i=1}$and question aware answer representation $\{w^q_{A_i}\}^a_{i=1}$ is computed. Three separate BiLSTMs are applied to the concatenation of these vectors to encode the temporal dependency as follows:

$$\mathbf{h}^q = BiLSTM(\{w_{Q_i}\}^Q_{i=1}) \tag{4.5}$$

$$\mathbf{h}^p = BiLSTM(\{[w_{P_i}; w^q_{P_i}]\}^P_{i=1}) \tag{4.6}$$

$$\mathbf{h}^a = BiLSTM(\{[w_{A_i}; w^q_{A_i}; w^p_{A_i}\}^A_{i=1}) \tag{4.7}$$

These are the the vector representations which encode the modified context representations.

Output Layer: Question sequence and answer sequence representation h^q; h^a are summarized into fixed-length vectors with self-attention which is defined as follows:

$$Att_{self}(\{\mathbf{u}_i\}^n_{i=1}) = \sum_{i=1}^{n} \alpha_i u_i \tag{4.8}$$

$$\alpha_i = softmax_i(\mathbf{W}_2^T\mathbf{u}) \tag{4.9}$$

Then we have question representation $\mathbf{q} = Att_{self}(\{\mathbf{h}^q_i\}^Q_{i=1})$, answer representation $\mathbf{a} = Att_{self}(\{\mathbf{h}^a_i\}^A_{i=1})$ and paragraph representation $\mathbf{p} = Att_{seq}(\mathbf{q}, \{\mathbf{h}^p_i\}^P_{i=1})$. The final output is the bilinear interaction between these vectors as follows:

$$y = \sigma(\mathbf{p}^T W_3 a + \mathbf{q}^T W_4 \mathbf{a}) \tag{4.10}$$

We feed a triple of context, question and answer into the model for each of the two answer options for a question and the maximum probability output of the two, is chosen as the final predicted answer for the given question.

4.3.1 Implementation details

The model is implemented in Tensorflow. Models are trained on a single GPU Titan X and each epoch takes about 320 seconds. The dimension of both forward and backward LSTM hidden state is set to 96. Dropout rate is set to 0.4 for both input embeddings and BiLSTM

outputs. For parameter optimization, we use Adam optimizer with an initial learning rate 0.001. The model converges after 70 epochs. Gradients are clipped to have a maximum L2 norm of 10. Minibatch with batch size 32 is used.

4.4 RESULTS

We present the results of the model along with the best performing baselines for the McScript in the following table:

Table 4.1: Results on McScript

Model	CommonSense Knowledge Source	Accuracy
Yuanfudao(Ensemble) [27]	ConceptNet	0.84
Mitre [28]	-	0.82
ELiRF-UPV [29]	ConceptNet	0.75
Our Model	ConceptNet	0.8225
Our Model+ Data Augmentation	ConceptNet	0.82321
Our Model+ Data Augmentation (per question)	ConceptNet	0.81773

As can be seen from Table 4.1 that we are very close to the best performing model on this dataset. The best performing model is an ensemble model which average the output probabilities of 9 models trained with the same dataset and network architecture but different random seeds. The authors also pre train the model on a separate dataset, RACE [30]. Our motivation is not to beat the best ensemble model but to show that our data augmentation technique does help boost the performance of this model, which is indeed the case in the above reported table. However, contrary to our intuition the model performance does not improve when data augmentation is performed for every question separately. We explore this further in the following sections.

4.4.1 Error Analysis

Below we present some examples from the test set to demonstrate how the data augmentation helps the model in certain situations and the areas where it can be refined further. All the added text is in italics.

PASSAGE: I had been having issues with heartburn. It's like every time I had a meal, I was in pain and in agony.Finally, I decided it was time to see the doctor. I called the receptionist at the office to make an appointment. She asked me what my problems were and booked me down for the next week. I came in early for my appointment. They had me update my health insurance card and fill out some paper work. Then it was a short wait before the nurse called my name. She brought me to the back and got my weight and other information. We talked a bit and she led me into the exam room. It was not long before the doctor came in. He checked me out and talked to me about my condition He gave me a prescription , but also told me to stay away from foods that made me hurt.long is not short. *A doctor is supposed to make people feel better. Doctor is related to health. Something you find at the hospital is a doctor.*
QUESTION: Were they in a hospital?
ANSWER: Yes

The above question answer pair is an example where the model answers correctly only after data augmentation, since the information that a doctor is at a hospital is not explicitly present in text and the model is not able to make the correct inference without this information.

PASSAGE: Last weekend, my wife and I had some friends over. After the party ended, we decided that we wanted to watch a movie. I decided to run to the Redbox booth to get a new movie that was out. I got my keys and went to the booth. I paid with my card for the movie that we wanted. I drove back to our house and I got the DVD out of the case. I own a PS4 , so I placed the DVD right into the machine without opening the console. I had to get the controller out of a drawer to play, but I found it easy to find out where to start the movie. We all relaxed on the couch and ate some candy while we watched the movie. We laughed at the funny parts and stayed awake for the whole movie, even when it got a little boring. *Some movies are funny.* ***A watch is a machine. A watch is a easy to carry clock.*** *The last thing you do when you travel is stand in front of your home. The first thing you do when you travel is find out where you are going . If you want to travel then you should decide where to. Play is a type of plan of action.*

In the above passage we observe one of the issues with our approach, which is the addition of text which is not relevant to the desired context. Here, we see that the word 'watch' refers to watching a movie, however the text that gets added is in the context of clock. This

is the cause of one of the biggest failures of this model.

PASSAGE: Yesterday I decided to boil some milk so I could make cottage cheese. The first thing I did was select a pan to boil the milk. I decided to choose the widest pan I had, because the wider pan would let me boil the milk more quickly. It is important to select the right pan, a narrow pan will take longer for the milk to come to a boil and will increase the risk of burning the milk on the bottom of the pan. Once I selected the pan, I poured the milk into the pan , and turned the heat on the stove top to medium high heat. For the next ten minutes, I constantly stirred the milk as it heated, to evenly distribute heat and to reduce the risk of the milk burning in the pan. After ten minutes of heating, the milk came to a boil, so I removed the pan from the heat, ready to use my sterilized milk to make cottage cheese. *You are likely to find a bottle in something to drink. Milk can be poured. Boil causes boil. Heat can cause water to boil. Heat is related to high.*
Other relations in ConceptNet is as follows:
Has Last Sub-event - cook_meal - grab_potholder - The last thing you do when you cook a meal is grab a potholder.
Is A - potholder - pad - potholder is a type of pad.

The above example shows that there is some other relevant evidence available in ConceptNet that could be useful to answer questions for this passage, but could not be extracted because we could not find the required query word/bi-grams in the passage according to our extraction rules.

PASSAGE: Today was the start of Spring, so I decided it was time to get some cleaning done. The first and largest thing to deal with was my apartment flat. I started by picking up any loose items that were not in their proper place. I put as many items into drawers and closets as possible to reduce clutter. I then moved all the furniture so I could clean the floors where the furniture is normally sitting. I vacuumed the exposed floor then moved the furniture back into place with a couple changes in the arrangement for fun. Then I swept the rest of the floor after dusting any surfaces that needed it. Afterwards , I cleaned all the windows, inside and out, and moved to the bathroom. After wiping down all the surfaces in the bathroom with a disinfecting cleaner, I took a look around my flat to assess my work. My apartment was now clean and it felt great! *change is a type of thing. Loosen is a type of change.A house has furniture.A house has a floor.You are likely to find a passage in a house.A house has a bathroom.Day is related to time. Pick is a synonym of clean. Dirty is not clean.* ***A soap is for cleaning. A soap can be used to clean something.***

Something you need to do before you take a bath is soap.
QUESTION: Was a lot of soap used to clean the flat?
ACTUAL ANSWER: No
PREDICTED ANSWER: Yes

In the above example it can be seen, that the model makes a mistake in making a prediction because text related to soap gets appended to the context which was not present earlier and allowed the model to predict the correct answer.

4.5 CONCLUSION

In this chapter we presented the details of the model for the SemEval Task 2018 machine comprehension using commonsense knowledge. We observed that data augmentation techniques using the knowledge extracted from structured databases such as ConceptNet can help improve the performance of the models for this task. We can conclude that this task helps researchers tackle the problem of question answering where the relevant information to resolve ambiguity might not be present in the provided passage. This is especially important since other famous datasets such as SQUAD, MS-MARCO etc. do not asses the model's ability to draw inferences from implicit knowledge which is very intuitive to humans.

However, we also observed that the data augmentation techniques suggested in this section have certain flaws and need to be refined further. Specifically, we need to take into consideration the relevant context before querying these databases and also design more sophisticated data mining techniques to be able to extract all the required information for a given question. Moreover, we also observed that ConceptNet might not always be sufficient to provide the required knowledge for answering questions and we need to explore other large scale knowledge bases to improve the performance of these systems.

CHAPTER 5: CONCLUSION

This thesis focused on developing deep learning based models for the task of question answering. We targeted two challenging problems in this area and introduced models that helped further improve the performance of the systems present in literature. The experiments we have presented demonstrate that these are complex problems and require further development of new datasets, techniques that could help tackle some of the issues we discovered during our research.

In chapter 3 we dealt with narrative question answering, which was based on the domain of stories, movie scripts and required generating answers that were not fixed spans in text. We proposed a multi task approach for this task and observed that the models had become too big to be trained on this dataset. We concluded that we either need to pretrain our model on similar datasets or resort to other techniques to prevent overfiiting.

In chapter 4, we dealt with the unique problem of incorporating common sense knowledge into machine reading comprehension systems. This is important for answering questions where the relevant context might not be explicitly present in the provided text. We developed novel information extraction rules from structured data bases and demonstrated that these improved the performance of the models developed for this task. However, we also observed from our experiments, certain flaws in our approach and the need to develop more robust inference rules. We also concluded that ConceptNet alone might not be sufficient to answer all questions and the requirement to explore other knowledge bases.

Through this thesis it can be seen that significant progress has been made in the recent past to develop elaborate QA systems. Development of huge datasets, synthetic or human generated have been the major supporter for driving the growth of these systems. However, each of these datasets have their own limitations and there still remains further scope for development of new datasets to allow modelling robust systems which are closer to the real world applications. Moreover, there seems to be a recent trend emerging in NLP literature where there is a push to move away from RNN based architecture [19]. Development of such models could be crucial for QA systems as they tackle some of the fundamental limitations of RNN. Also, the use of Reinforcement learning to generate more natural and human readable output sequences brings these systems one step closer to commercial deployment. However, progress on building systems that truly understand language is only possible if our evaluation metrics can distinguish real intelligent behavior from shallow pattern matching. Therefore, we feel that exploring ways to discover the limitations of these models is a promising future research direction and is contingent for the development of new deep learning models.

REFERENCES

[1] T. Kočiskỳ, J. Schwarz, P. Blunsom, C. Dyer, K. M. Hermann, G. Melis, and E. Grefenstette, "The NarrativeQA reading comprehension challenge," *Transactions of the Association of Computational Linguistics*, vol. 6, pp. 317–328, 2018.

[2] S. Ostermann, M. Roth, A. Modi, S. Thater, and M. Pinkal, "SemEval-2018 Task 11: machine comprehension using commonsense knowledge," in *Proceedings of The 12th International Workshop on Semantic Evaluation*, 2018, pp. 747–757.

[3] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Computer Vision and Pattern Recognition, 2009. CVPR 2009. IEEE Conference on*. Ieee, 2009, pp. 248–255.

[4] M. Richardson, C. J. Burges, and E. Renshaw, "Mctest: A challenge dataset for the open-domain machine comprehension of text," in *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, 2013, pp. 193–203.

[5] K. M. Hermann, T. Kocisky, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom, "Teaching machines to read and comprehend," in *Advances in Neural Information Processing Systems*, 2015, pp. 1693–1701.

[6] F. Hill, A. Bordes, S. Chopra, and J. Weston, "The Goldilocks Principle: Reading Children's Books with Explicit Memory Representations," *CoRR*, vol. abs/1511.02301, 2016.

[7] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang, "SQuAD: 100,000+ Questions for Machine Comprehension of Text," in *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, Nov. 2016. [Online]. Available: https://www.aclweb.org/anthology/D16-1264 pp. 2383–2392.

[8] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, and R. a. Majumder, "MS MARCO: A Human Generated MAchine Reading COmprehension Dataset," November 2016. [Online]. Available: https://www.microsoft.com/en-us/research/publication/ms-marco-human-generated-machine-reading-comprehension-dataset/

[9] M. J. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi, "Bidirectional Attention Flow for Machine Comprehension," *CoRR*, vol. abs/1611.01603, 2017.

[10] D. Bahdanau, K. Cho, and Y. Bengio, "Neural Machine Translation by Jointly Learning to Align and Translate," *CoRR*, vol. abs/1409.0473, 2015.

[11] D. Chen, J. Bolton, and C. D. Manning, "A thorough examination of the cnn/daily mail reading comprehension task," *arXiv preprint arXiv:1606.02858*, 2016.

[12] R. Kadlec, M. Schmid, O. Bajgar, and J. Kleindienst, "Text Understanding with the Attention Sum Reader Network," in *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Berlin, Germany: Association for Computational Linguistics, Aug. 2016. [Online]. Available: https://www.aclweb.org/anthology/P16-1086 pp. 908–918.

[13] Y. Cui, Z. Chen, S. Wei, S. Wang, T. Liu, and G. Hu, "Attention-over-Attention Neural Networks for Reading Comprehension," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, July 2017. [Online]. Available: https://www.aclweb.org/anthology/P17-1055 pp. 593–602.

[14] S. Sukhbaatar, J. Weston, R. Fergus et al., "End-to-end memory networks," in *Advances in neural information processing systems*, 2015, pp. 2440–2448.

[15] J. Pennington, R. Socher, and C. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, 2014, pp. 1532–1543.

[16] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Advances in neural information processing systems*, 2013, pp. 3111–3119.

[17] B. Dhingra, H. Liu, R. Salakhutdinov, and W. W. Cohen, "A comparative study of word embeddings for reading comprehension," *arXiv preprint arXiv:1703.00993*, 2017.

[18] D. Bahdanau, T. Bosc, S. Jastrzebski, E. Grefenstette, P. Vincent, and Y. Bengio, "Learning to compute word embeddings on the fly," *arXiv preprint arXiv:1706.00286*, 2017.

[19] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in Neural Information Processing Systems*, 2017, pp. 5998–6008.

[20] O. Vinyals, M. Fortunato, and N. Jaitly, "Pointer networks," in *Advances in Neural Information Processing Systems*, 2015, pp. 2692–2700.

[21] R. Paulus, C. Xiong, and R. Socher, "A deep reinforced model for abstractive summarization," *arXiv preprint arXiv:1705.04304*, 2017.

[22] S. Reznikova and L. Derczynski, "IUCM at SemEval-2018 Task 11: Similar-Topic Texts as a Comprehension Knowledge Source," in *Proceedings of The 12th International Workshop on Semantic Evaluation*, 2018, pp. 1068–1072.

[23] B. Yang and T. Mitchell, "Leveraging Knowledge Bases in LSTMs for Improving Machine Reading," in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Vancouver, Canada: Association for Computational Linguistics, July 2017. [Online]. Available: https://www.aclweb.org/anthology/P17-1132 pp. 1436–1446.

[24] R. Grishman, D. Westbrook, and A. Meyers, "NYUs English ACE 2005 system description," *ACE*, vol. 5, 2005.

[25] J. Lu, J. Yang, D. Batra, and D. Parikh, "Hierarchical question-image co-attention for visual question answering," in *Advances In Neural Information Processing Systems*, 2016, pp. 289–297.

[26] R. Speer, J. Chin, and C. Havasi, "Conceptnet 5.5: An open multilingual graph of general knowledge," in *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

[27] L. Wang, M. Sun, W. Zhao, K. Shen, and J. Liu, "Yuanfudao at SemEval-2018 Task 11: Three-way Attention and Relational Knowledge for Commonsense Machine Comprehension," in *Proceedings of The 12th International Workshop on Semantic Evaluation*. New Orleans, Louisiana: Association for Computational Linguistics, June 2018. [Online]. Available: https://www.aclweb.org/anthology/S18-1120 pp. 758–762.

[28] E. Merkhofer, J. Henderson, D. Bloom, L. Strickhart, and G. Zarrella, "MITRE at SemEval-2018 Task 11: commonsense reasoning without commonsense knowledge," in *Proceedings of the 12th International Workshop on Semantic Evaluation*, 2018, pp. 1078–1082.

[29] J.-Á. González, L.-F. Hurtado, E. Segarra, and F. Pla, "ELiRF-UPV at SemEval-2018 Task 11: Machine Comprehension using Commonsense Knowledge," in *Proceedings of The 12th International Workshop on Semantic Evaluation*, 2018, pp. 1034–1037.

[30] G. Lai, Q. Xie, H. Liu, Y. Yang, and E. Hovy, "RACE: Large-scale ReAding Comprehension Dataset From Examinations," in *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*. Copenhagen, Denmark: Association for Computational Linguistics, Sep. 2017. [Online]. Available: https://www.aclweb.org/anthology/D17-1082 pp. 785–794.