Deep Learning for Reading and Understanding Language

Tomáš Kočiský

Linacre College

University of Oxford

A thesis submitted for the degree of

Doctor of Philosophy

Michaelmas 2017

Acknowledgements

I am deeply grateful to my supervisor Phil Blunsom for his advice and encouragement throughout my studies and all the insightful discussions without which this thesis would have never been accomplished.

I want to also thank my colleagues at the NLP group at Oxford where I spent first years of my studies. Lastly, I want to thank all my colleagues at the language team at DeepMind in London for all the fruitful collaborations and discussions over the last several years. I would especially like to thank Karl Moritz Hermann, Gábor Melis, Chris Dyer, and Ed Grefenstette for our many collaborations.

Abstract

This thesis presents novel tasks and deep learning methods for machine reading comprehension and question answering with the goal of achieving natural language understanding.

First, we consider a semantic parsing task where the model understands sentences and translates them into a logical form or instructions. We present a novel semi-supervised sequential autoencoder that considers language as a discrete sequential latent variable and semantic parses as the observations. This model allows us to leverage synthetically generated unpaired logical forms, and thereby alleviate the lack of supervised training data. We show the semi-supervised model outperforms a supervised model when trained with the additional generated data.

Second, reading comprehension requires integrating information and reasoning about events, entities, and their relations across a full document. Question answering is conventionally used to assess reading comprehension ability, in both artificial agents and children learning to read. We propose a new, challenging, supervised reading comprehension task. We gather a large-scale dataset of news stories from the CNN and Daily Mail websites with Cloze-style questions created from the highlights. This dataset allows for the first time training deep learning models for reading comprehension. We also introduce novel attention-based models for this task and present qualitative analysis of the attention mechanism. Finally, following the recent advances in reading comprehension in both models and task design, we further propose a new task for understanding complex narratives, NarrativeQA, consisting of full texts of books and movie scripts. We collect human written questions and answers based on high-level plot summaries. This task is designed to encourage development of models for language understanding; it is designed so that successfully answering their questions requires understanding the underlying narrative rather than relying on shallow pattern matching or salience. We show that although humans solve the tasks easily, standard reading comprehension models struggle on the tasks presented here.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis presents tasks and models for machine reading comprehension of text, tasks that are substantially diagnostic for learning natural language understanding at the human level.

Natural language understanding is hard to characterize using algorithms or rules. It is a set of behaviours that humans exhibit while performing tasks such as reading and comprehending, or transforming sentences into queries or logical forms. We will thus try to learn to understand natural language by learning to mimic this behaviour, and in particular, in the setting of learning reading comprehension and semantic parsing. One of the challenges towards this is getting enough data, which we address by semi-supervised learning or scalably gathering data for new types of tasks.

Reading comprehension is the ability to understand text; it is a complex task that people do every day with ease. Think about reading a book—comprehension of a book requires, informally, understanding a long body of text, with intricate sequences of events with complex characters, reasoning about various aspects of the narrative, like character motivations, inferring the event and place spatial layout, and abstracting the most salient parts to be able to reason about and reproduce what has been read. This ability clearly requires understanding and reasoning far beyond surface forms of individual sentences.

The field of natural language processing (NLP) is concerned with processing natural language in various forms in order to understand or generate language, which would, for example, enable interactions between humans and machines. Moreover, as most knowledge and discoveries are recorded using a natural language it is important that we are able to process, understand and learn from it efficiently, at scale. Standard practice in NLP is to approach language through simpler components thought to be required for the processing

such as part-of-speech tagging, syntactic parsing, morphological analysis, or relation extraction. We will tackle all these interconnected tasks at once with the goal of learning to understand language.

Reading comprehension can be viewed as an ultimate test of natural language understanding, as it requires all the sub-components of language understanding often studied separately in NLP. Moreover, we can use tasks that require reading comprehension for learning to understand language, by mimicking this process. However, these tasks too require careful design and imaginative dataset construction to avoid a setup where doing well without language understanding, as humans display, is possible.

Human level of text understanding is the ultimate goal as such level of understanding is often associated with, or viewed as a result of, intelligence. Turing (1950) has introduced an empirical test for such ability, now called the Turing Test. The aim for an interrogator is to distinguish a human from a machine, and the machine needs to fool the interrogator that it is human (Jurafsky and Martin, 2009). It has been shown (Weizenbaum, 1966) that simple, template-based programs without any knowledge, understanding, or intelligence can sometime fool the interrogator that they are human. This exemplifies how both the task design and the approach to it affect what we can learn, and what we need to learn to perform well, and in particular, often we can get good performance on a task with only shallow language understanding, contrary to our goals. Both reading comprehension and semantic parsing likely require this level of understanding, but they are also intrinsically useful.

Tasks and processing in NLP require language understanding to various extent to be useful in practice. For some tasks lexicalized models do well (e.g. part-of-speech tagging), in others, processing at level of phrases or sentences achieves useful results (e.g. machine translation). To achieve language understanding we need to focus on tasks where such shortcuts are not possible, where fine understanding of larger and more complex texts is required to do well, and at the same time, tasks that are easy and natural for humans.

1.1 Aims

This thesis aims to learn natural language understanding in two settings using recent deep learning techniques. First, we will use recent deep learning techniques for the task of transducing natural language to logical forms (i.e. semantic parsing), despite its challenges

in terms of only small datasets available and large entity space. We will explore a semi-supervised setup that transfers knowledge from synthetically generated logical forms to help the task.

Second, this thesis tackles the challenging problem of machine reading comprehension through question answering about unstructured text. This is a task that ideally requires language understanding and where shallow processing will have difficulty. However, the task design and scalable data gathering are crucial, perhaps more so than model design.

We will apply deep learning methods to the problem of reading comprehension, including the successful attention based models. We will design new tasks and collect data based on news articles and their highlights. Subsequently, based on the research that followed our first deep learning attempt at reading comprehension, we review the progress and suggest a refined and a more complex task together with a data gathering method that will provide us a new test for natural language understanding.

1.2 Contributions

Contributions of this thesis are as follows.

- We propose a semi-supervised deep learning method for learning from small datasets for semantic parsing. We propose a novel model based on an autoencoder architecture that considers language as a discrete sequential latent variable and allows us to leverage synthetic logical forms to alleviate lack of supervised training data.
- We further propose a differentiable alternative to the intractable marginalization to train this architecture.
- We show our model works on three different semantic parsing tasks, those tasks have in common that the target sequences can be easily generated from a grammar, which in our case is a machine readable query or sequence of instructions, and the source sequence is natural language. We show that this semi-supervised model outperforms the supervised model when trained on additional generated data as well as on subsets of the existing data.
- To teach machines reading comprehension, we design a new supervised task with news articles and automatically generated Cloze-style questions (i.e. fill-in-a-word

sentences) based on human written highlights of the articles. We gather a dataset that is for the first time large enough for development and training deep learning models. We design this task to avoid exploitation of the task structure, either by models or researchers, through entity anonymization (see Section 4.2.1).

- We present baselines and novel attention-based neural models for this task and qualitatively analyse the model's attention weights. We demonstrate that the attention-based models are superior to former methods on this language understanding task.
- We review the large body of subsequent work inspired by this work, as well as the novel models designed for it, subsequent related tasks and datasets for reading comprehension.
- We further propose a much more challenging task and dataset, for both training and evaluation. Instead of relatively short news stories that require a lot of real world knowledge not contained therein, we propose a new very challenging task and dataset for complex narrative comprehension, NarrativeQA, based on full books and movie scripts. We introduce this task to improve on shortcomings of previous tasks' design that were able to leverage simple pattern matching and paraphrase to answer most questions of the reading comprehension test.
- We present a new method for data collection where annotators create questions and answers from high-level plot summaries, instead of the full text we want to understand.
- We establish baselines and benchmarks based on recent state-of-the-art reading comprehension models. We find that previous approaches struggle with this new challenge.
- We investigate the challenges of the proposed task and provide a qualitative analysis of NarrativeQA.

1.3 Thesis Structure

This thesis is comprised of five main chapters with two chapters reviewing background and other work. Chapters 3, 4, and 6 are based on published work.

Below follows an overview of each chapter.

Chapter 2: Background

This chapter presents background for the following chapters, mainly on language understanding, question answering, reading comprehension, and deep learning methods. We review more recent work on reading comprehension in Chapter 5.

Chapter 3: Semi-Supervised Semantic Parsing

This chapter discusses our work on semi-supervised semantic parsing that employs sequential autoencoders and treats language as a latent variable. The work presented in this chapter is based on the following publication:

> Tomáš Kočiský, Gábor Melis, Edward Grefenstette, Chris Dyer, Wang Ling, Phil Blunsom, Karl Moritz Hermann. 2016 Semantic Parsing with Semi-Supervised Sequential Autoencoder. In *Proceedings of EMNLP*.

Chapter 4: Deep Learning for Reading Comprehension

This chapter present a novel dataset and new attention-based neural models for reading comprehension on news articles. We further present analysis of the dataset and visualization of the attention mechanism learned. The work presented in this chapter is based on the following publication:

> Karl Moritz Hermann, Tomáš Kočiský, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, Phil Blunsom. 2015 Teaching Machines to Read and Comprehend. In *Proceedings of NIPS*.

Chapter 5: Rise of Reading Comprehension

In this chapter we review subsequent work leveraging the dataset introduced in Chapter 4, and we also review other related datasets introduced afterwards.

Chapter 6: Complex Narrative Understanding

This chapter introduces a novel, challenging reading comprehension task on narratives, NarrativeQA. We construct dataset of books and movie scripts, together with human-written questions and answers that avoids some of the shortcoming of previous tasks. The work presented in this chapter is based on the following publication:

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, Edward Grefenstette. 2018 The NarrativeQA Reading Comprehension Challenge. In *Transactions of the Association for Computational Linguistics (TACL)*.

Chapter 7: Conclusions

The final chapter of this thesis summarizes the conclusions of this work and suggest possible future directions.

Chapter 2

Background on Question Answering, Reading Comprehension, and Deep Learning

Chapter Abstract

This chapter briefly reviews the background for question answering, question answering on structured knowledge bases and unstructured text. Subsequently, we review relevant topics in deep learning, types of neural networks, recurrent cells, and the attention mechanism.

Natural language understanding involves several abilities to manipulate natural language, such as understanding semantics and syntax of text, reasoning, or text production. Such abilities, or subset of them, of language understanding can be demonstrated to various extent by doing tasks such as question answering, reading comprehension, named entity recognition, part-of-speech tagging, sentiment analysis, or parsing. To do well on some of the tasks requires the ability to do well on many of the other tasks. In this thesis we focus on two of such tasks: semantic parsing and reading comprehension.

In the following sections, we introduce question answering, including semantic parsing (Section 2.1.1), reading comprehension (Section 2.1.2), and recent deep learning methods that allow us to tackle this challenge.

2.1 Question Answering and Reading Comprehension

Question answering (QA) has a long history (Kolomiyets and Moens, 2011). First question answering systems, from 1960s, have focused on closed domain question answering over a database of facts, such as baseball games (Green et al., 1961), data about rock samples (Woods et al., 1972). Later systems interactively asked the user to refine the question, still in closed domains such as medical concepts, or interacting with blocks in the game of SHRDLU (Winograd, 1972a). Today, research often focuses on answering questions based on unstructured text.

The systems considered vary greatly in the types of questions they consider, the supporting context, and the answer we expect from them. For examples of datasets, see Table 5.2.

Questions: We consider a user who is seeking some information. This information need is expressed as a natural language phrase or a sentence. The question can be seeking a fact of several types, such as a number, date, person, place, or relation. The question can be more complex and seek a narrative, like an explanation, a reason, a summary. The question could also aim to retrieve an item, like a document or an image. For all these categories, we could seek one answer or a list of them.

Context: The early QA systems used a structured knowledge base, or a database, of factual information. This was often obtained by extracting from unstructured text or tables, or created by hand. As a context, we could also have a corpus of documents, a single document (CNN/Daily Mail dataset, Chapter 4, e.g.), or part of it. Furthermore, we could have information provided in other modalities than text, such as images, video, GPS coordinates, sensor information, or any combination of these.

Answer: The type of answer we seek affects the complexity of the task and of the evaluation. We could broadly categorize the types of answers into extractive and generated. We could be seeking a document or an image, or we could be seeking a single fact, a paragraph/sentence or a span of a few words (Rajpurkar et al., 2016, e.g.), or we could be selecting from a set of candidate answers (Richardson et al., 2013a; Hill et al., 2016, e.g.). A more complex task is to generate a natural language answer like a short statements or an explanation. Furthermore, we could expect the system to pose a question instead of an

answer to interactively refine the original query.

Reading and comprehending text is an ability people can develop. This corresponds to the setup where we want to answer questions based on unstructured free-form text and it is how people are tested at this ability.

2.1.1 Question Answering on Knowledge Bases

Knowledge bases, such as Freebase or other domain specific databases, come with a schema, a set of relations and entities. They contain factual information, often extracted from unstructured text. For example, the correctness of the facts is assumed to correlate with how often the fact occurs.

The task here becomes transforming the natural language question into a database query or a logical form that can be evaluated using this database to retrieve the answer. The early domain-specific systems (such as for baseball facts, data about rocks, or medical concepts, as discussed above) used such knowledge bases as context for answering.

The task of transforming a natural language query into a logical form is referred to as *semantic parsing*. For examples of sentences and queries see Table 3.1.

Semantic parsing is a well-studied problem with numerous approaches including inductive logic programming (Zelle and Mooney, 1996), string-to-tree (Galley et al., 2004) and string-to-graph (Jones et al., 2012) transducers, grammar induction (Kwiatkowski et al., 2011; Artzi and Zettlemoyer, 2013; Reddy et al., 2014) or machine translation (Wong and Mooney, 2006; Andreas et al., 2013).

Other approaches, for example, focus on learning from question-answer pairs and avoid the need for annotated logical forms (Berant et al., 2013; Bordes et al., 2015, i.a.).

The tasks often come with only small training sets which makes the problem especially challenging for deep learning methods. The success of deep learning methods for QA on knowledge bases is relatively recent (Dong and Lapata, 2016; Jia and Liang, 2016; Bordes et al., 2015). We explore a new approach in Chapter 3.

2.1.2 Question Answering on Unstructured Text

One of the ways to approach question answering from unstructured text is by considering extractive answers at the level of passages, sentences, or phrases, which can be tackled by

standard information retrieval methods. This is employed most often for factoid questions.

The second thread of research is on understanding complex stories or narratives such as short stories or books. Learning to understand books through other modelling objectives than question answering has become an important subproblem in NLP. These include high level plot understanding through clustering of novels (Frermann and Szarvas, 2017), summarization of movie scripts (Gorinski and Lapata, 2015) or finding who committed a crime in a multi-modal crime drama (Frermann et al., 2017), to more fine grained processing by inducing character types (Bamman et al., 2014b; Bamman et al., 2014a), understanding relationships between characters (Iyyer et al., 2016; Chaturvedi et al., 2017), or understanding plans, goals, and narrative structure in terms of abstract narratives (Schank and Abelson, 1977; Wilensky, 1978; Black and Wilensky, 1979; Chambers and Jurafsky, 2009).

Hirschman et al. (1999) and Richardson et al. (2013a) were the first to introduce question answering tasks on free-form stories. The datasets are relatively small (60 development and 60 test, and 500 stories, respectively). MCTest by Richardson et al. (2013a) is crowdsourced and therefore the data gathering is more scalable. The stories are fictional and thus largely self-contained. They propose simple baselines and also apply a model for recognizing textual entailment (RTE) to try to solve this multiple choice question answering task.

For examples of question answering on unstructured text see Figure 6.1, Figure 6.5, or Table 4.2.

In Chapter 4 we will introduce the first large-scale task for reading comprehension (tested by question answering) that for the first time allows training deep learning models (see Section 2.2) for this task. We review subsequent work on this dataset in Chapter 5, subsequently introduced tasks/datasets in Sections 5.2 and 6.2, and propose a further task for complex-narrative reading comprehension in Section 6.3.

2.1.3 Evaluation

There are several metrics commonly used, depending on the answer type and the model.

Accuracy is commonly employed for factoid question answering, paragraph or sentence extraction, or in multiple-choice question answering, including those with fill-in-a-word questions. Alternatively, if permitted by the model, we could use a ranking loss such as

the mean reciprocal rank (MRR), which is the mean over examples of $1/r$, where $r \in \{1, 2, \ldots\}$ is the rank of the correct answer among candidates.

For semantic parsing, where we predict logical forms, we can execute the queries and compare answers using the F_1 metric, or as we do in Chapter 3, we can consider the accuracy of exactly predicting the query.

For span prediction (e.g., Rajpurkar et al. (2016)), we could use mean F1 score, where for each candidate answer and reference answer are treated as bags of words for calculation of the F1 score.

Evaluation of generated answers is more complex. There is currently no best and standard automatic metric for this. Sentence comparison metrics from the machine translation or summarization literature, such as, Bleu-1, Bleu-4 (Papineni et al., 2002), Meteor (Denkowski and Lavie, 2011), or Rouge-L (Lin, 2004) are used. It is unclear whether these metrics are suitable for answers which are often short phrases, rather than sentences.

Burges (2013) suggests a definition and a test of machine comprehension where a machine comprehends a passage when, for a question that most native speakers of that language can answer, it can produce a string that would be judged by those speakers as an answer to that question. This would suggest that the best evaluation would be human evaluation of answer appropriateness for a given question, and that the evaluation might be as hard as answering the questions.

2.2 Deep Learning

Recent popularity and success of artificial neural networks, complex differentiable functions optimized by gradient-based methods, makes them an obvious candidate for an end-to-end approach to learning to understand natural language from unstructured text.

For an overview of deep learning in neural networks see work by Schmidhuber (2014).

We only broadly review types of neural networks based on their structure, recurrent cells, and the attention mechanism, where the latter two both have features to address information propagation and optimization challenges in recurrent neural networks.

2.2.1 Types of Neural Networks

Broadly, there are three types of structures of neural networks when considering sequential input, such as vectors representing words: feed-forward, recurrent, and recursive.

Feed-forward neural networks are the simplest. There is no good way to handle variable length, unless the network in convolutional.

Recurrent neural networks (RNNs, Elman (1990)) apply the same transformation (i.e., cell), with the same weights, for each element of the input sequence in turn, producing an output vector and a recurrent state vector(s). RNNs are the standard, simplest and somewhat effective way to process moderate-length unstructured text. Long sequences pose a challenges for propagating information and for optimization, some of which are addressed by cleverly designed RNN cells (see section below).

Recursive neural networks (Goller and Küchler, 1996) process the input sequence along a recursive structure, for example a parse tree of the sentence, applying transformations at each node, conditioned on the children and possibly annotations of the recursive structure.

2.2.2 Recurrent Cells

One of the most important advances for RNNs was the long short-term memory (LSTM) cell (Hochreiter and Schmidhuber, 1997). RNNs suffer from exploding and vanishing gradients due to non-linearities which hinder learning. LSTMs have a recurrent state which, in part (see definition of $c(t)$), contains only a linear transformation through time (i.e., along the sequence). Alternatives to LSTM are, for example, the gated recurrent unit (GRU) by Cho et al. (2014), or automatically designed cell architectures (Zoph and Le, 2016).

Below we formally present the LSTM, and then a second variant of LSTM with skip connections and peephole connections that can be used with multi-layer, deep RNNs. We use the Deep LSTM cell in Chapter 4 and the simpler LSTM cell in the rest of the thesis.

LSTM Cell LSTM (Hochreiter and Schmidhuber, 1997) is one of the most commonly used RNN cells. It contains eight matrix multiplies (which can be optimized to less) and a number of element-wise transformations. The recurrent state consists of a memory cell c and a hidden state h, which are then updated using input, output, and forget gates. Notice, that $c(t)$ is a linear transformation of $c(t-1)$, which helps with exploding and vanishing

gradients.

$$
\begin{aligned}
i(t) &= \sigma\left(W_{xi}x(t) + W_{hi}h(t-1) + b_i\right) \\
f(t) &= \sigma\left(W_{xf}x(t) + W_{hf}h(t-1) + b_f\right) \\
o(t) &= \sigma\left(W_{xo}x(t) + W_{ho}h(t-1) + b_o\right) \\
c(t) &= f(t)c(t-1) + i(t)\tanh(W_{xc}x(t) + W_{hc}h(t-1) + b_c) \\
h(t) &= o(t)\tanh(c(t))
\end{aligned}
$$

where $h(t), c(t)$ are the hidden and cell states, at time t; i, f, o are the input, forget, and output gates respectively; σ is the sigmoid element-wise non-linearity; and b are the biases.

Deep LSTM Cell For multi-layer RNNs, we can augment the LSTM cell with skip connections, that "skip" layers, and so aid optimizations. We used the following cell in Section 4.3.2. The cell below, compared to the LSTM above, also contains peephole connections (Gers et al., 2003), which is the dependence on $c(t-1,k)$ of the three gates. With skip connection from each input $x(t)$ to every hidden layer, and from every hidden layer to the output $y(t)$:

$$
\begin{aligned}
x'(t,k) &= x(t)||y'(t,k-1), \qquad y(t) = y'(t,1)||\ldots||y'(t,K) \\
i(t,k) &= \sigma\left(W_{kxi}x'(t,k) + W_{khi}h(t-1,k) + W_{kci}c(t-1,k) + b_{ki}\right) \\
f(t,k) &= \sigma\left(W_{kxf}x(t) + W_{khf}h(t-1,k) + W_{kcf}c(t-1,k) + b_{kf}\right) \\
c(t,k) &= f(t,k)c(t-1,k) + i(t,k)\tanh\left(W_{kxc}x'(t,k) + W_{khc}h(t-1,k) + b_{kc}\right) \\
o(t,k) &= \sigma\left(W_{kxo}x'(t,k) + W_{kho}h(t-1,k) + W_{kco}c(t,k) + b_{ko}\right) \\
h(t,k) &= o(t,k)\tanh\left(c(t,k)\right) \\
y'(t,k) &= W_{ky}h(t,k) + b_{ky}
\end{aligned}
$$

where $||$ indicates vector concatenation $h(t,k)$ is the hidden state for layer k at time t, and i, f, o are the input, forget, and output gates respectively, σ is the sigmoind element-wise non-linearity, and b are the biases.

2.2.3 Attention

Another important advance that enabled using RNNs for longer sequences is the attention mechanism introduced by Bahdanau et al. (2015). The mechanism allows to "choose" from a longer sequence of vectors (e.g., output vectors of an encoder RNN). This is done by taking a weighted average of the sequence vectors, weighted by normalized attention weights. This access to the full sequence, intuitively, aids propagation of information in longer sequences. In other words, it can help to avoid a bottleneck of using the last recurrent state of an RNN to represent the entire variable-length and possibly long sequence.

Given a sequence of vectors $v_1, \ldots, v_n$ (e.g., output/hidden states of an encoder RNN) and a query vector q (e.g., a state of the decoder RNN, and possibly additional conditioning vectors), we compute the attention weights m, normalized attention weights s, and the attended value r as follows:

$$
\begin{aligned}
m_i &= a(v_i, q) \\
s_i &= \frac{\exp(m_i)}{\sum_{j=1}^{n} \exp(m_j)} \\
r &= \sum_{i=1}^{n} s_i v_i
\end{aligned}
$$

This particular attention method computes the attention weights m_i based on the values we want to retrieve, an alternative method can treat the sequence to attend as key-value pairs, compute the attention weights based on the query and the keys, and attend (i.e., average) over the values.

The attention function a can be any feed-forward neural network. For example, an MLP with two layers and a $\tanh$ non-linearity (as in Section 4.3.2) or the dot product (as in Section 6.4).

Attention can be incorporated into a model in several ways, for example, attending over the source sentence given a decoder state in an encoder-decoder model (Bahdanau et al., 2015), attending over previous tokens given the current token in a sequence encoder (Cheng et al., 2016), or self-attending to the entire sequence (Vaswani et al., 2017).

Chapter 3

Semi-Supervised Semantic Parsing

Chapter Abstract

We present a novel semi-supervised approach for sequence transduction and apply it to semantic parsing. The unsupervised component is based on a generative model in which latent sentences generate the unpaired logical forms. We apply this method to a number of semantic parsing tasks focusing on domains with limited access to labelled training data and extend those datasets with synthetically generated logical forms.

3.1 Introduction

Neural approaches, in particular attention-based sequence-to-sequence models, have shown great promise and obtained state-of-the-art performance for sequence transduction tasks including machine translation (Bahdanau et al., 2015), syntactic constituency parsing (Vinyals et al., 2015b), and semantic role labelling (Zhou and Xu, 2015). A key requirement for effectively training such models is an abundance of supervised data.

In this chapter we focus on learning mappings from input sequences x to output sequences y (see examples in Table 3.1) in domains where the latter are easily obtained, but annotation in the form of (x, y) pairs is sparse or expensive to produce, and propose a novel architecture that accommodates semi-supervised training on sequence transduction

The work presented in this chapter was originally presented in Kočiský et al. (2016).

Dataset	Example
GEO	what are the high points of states surrounding mississippi answer(high_point_1(state(next_to_2(stateid('mississippi'))))))
NLMAPS	Where are kindergartens in Hamburg? query(area(keyval('name','Hamburg')),nwr(keyval('amenity','kindergarten')), qtype(latlong))
SAIL	turn right at the bench into the yellow tiled hall $(1, 6, 90)$ FORWARD - FORWARD - RIGHT - STOP $(3, 6, 180)$

Table 3.1: Examples of natural language x and logical form y from the three corpora and tasks used in this chapter. Note that the SAIL corpus requires additional information in order to map from the instruction to the action sequence.

tasks. To this end, we augment the transduction objective ($x \mapsto y$) with an autoencoding objective where the input sequence is treated as a latent variable ($y \mapsto x \mapsto y$), enabling training from both labelled pairs and unpaired output sequences. This is common in situations where we encode natural language into a logical form governed by some grammar or database.

While such an autoencoder could in principle be constructed by stacking two sequence transducers, modelling the latent variable as a series of discrete symbols drawn from multinomial distributions creates serious computational challenges, as it requires marginalizing over the space of latent sequences Σ_x^*, all the sequences over the set Σ_x (e.g. words). To avoid this intractable marginalization, we introduce a novel differentiable alternative for draws from a softmax which can be used with the reparameterization trick of Kingma and Welling (2014). Rather than drawing a discrete symbol in Σ_x from a softmax, we draw a distribution over symbols from a logistic-normal distribution at each time step. These serve as continuous relaxations of discrete samples, providing a differentiable estimator of the expected reconstruction log likelihood.

We demonstrate the effectiveness of our proposed model on three semantic parsing tasks: the GEOQUERY benchmark (Zelle and Mooney, 1996; Wong and Mooney, 2006), the SAIL maze navigation task (MacMahon et al., 2006) and the Natural Language Querying corpus (Haas and Riezler, 2016) on OpenStreetMap. We chose these datasets because they are diverse in terms of logical forms, and they are small and so could benefit from a semi-supervised training method. As part of our evaluation, we introduce simple mechanisms for generating large amounts of unsupervised training data for two of these tasks.

In most settings, the semi-supervised model outperforms the supervised model, both when trained on additional generated data as well as on subsets of the existing data.

3.2 Related Work

Semantic parsing The tasks in this chapter all broadly belong to the domain of semantic parsing, which describes the process of mapping natural language to a formal representation of its meaning. This is extended in the SAIL navigation task, where the formal representation is a function of both the language instruction and a given environment. For further review, see Section 2.1.1.

While a large number of relevant literature focuses on defining the grammar of the logical forms (Zettlemoyer and Collins, 2005), other models learn purely from aligned pairs of text and logical form (Berant and Liang, 2014), or from more weakly supervised signals such as question-answer pairs together with a database (Liang et al., 2011). Recent work of Jia and Liang (2016) induces a synchronous context-free grammar and generates additional training examples (x, y), which is one way to address data scarcity issues. The semi-supervised setup proposed here offers an alternative solution to this issue.

Discrete autoencoders Very recently there has been some related work on discrete autoencoders for natural language processing (Suster et al., 2016; Marcheggiani and Titov, 2016, *i.a.*) This work presents a first approach to using effectively discretized sequential information as the latent representation without resorting to draconian assumptions (Ammar et al., 2014) to make marginalization tractable. While our model is not exactly marginalizable either, the continuous relaxation makes training far more tractable. A related idea was recently presented in Gülçehre et al. (2015), who use monolingual data to improve machine translation by fusing a sequence-to-sequence model and a language model.

3.3 Model

Our sequential autoencoder is shown in Figure 3.1. At a high level, it can be seen as two sequence-to-sequence models with attention (Bahdanau et al., 2015) chained together. More precisely, the model consists of four LSTMs (Hochreiter and Schmidhuber, 1997), hence the name SEQ4. The first, a bidirectional LSTM, encodes the sequence y; next, an

Input from unsupervised data (y)	**Generated latent representation** (x)
answer smallest city loc_2 state stateid _STATE_	what is the smallest city in the state of _STATE_ </S>
answer city loc_2 state next_to_2 stateid _STATE_	what are the cities in states which border _STATE_ </S>
answer mountain loc_2 countryid _COUNTRY_	what is the lakes in _COUNTRY_ </S>
answer state next_to_2 state all	which states longer states show peak states to </S>

Table 3.2: Positive and negative examples of latent language together with the randomly generated logical form from the unsupervised part of the GEOQUERY training. Note that the natural language (x) does not occur anywhere in the training data in this form.

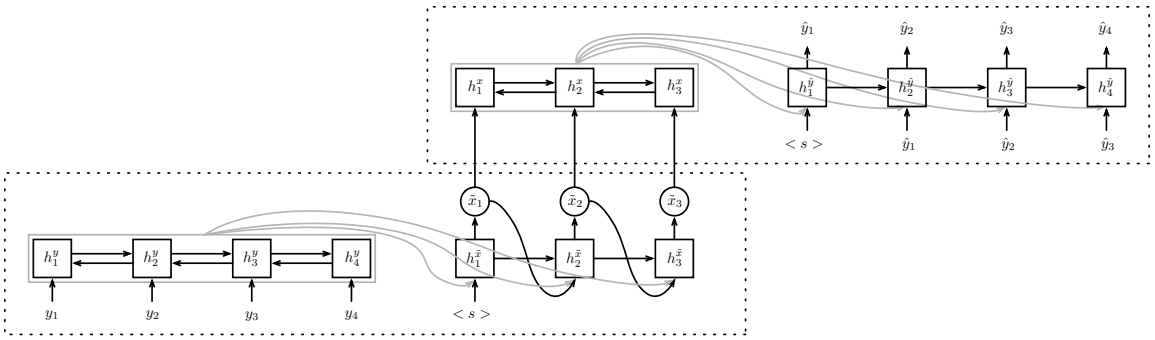


Figure 3.1: SEQ4 model with attention-sequence-to-sequence encoder and decoder. Circle nodes represent random variables, square nodes are steps of the RNN. $< s >$ is the start of sentence symbol.

LSTM with stochastic output, described below, draws a sequence of distributions $\tilde{x}$ over words in vocabulary Σ_x, i.e. each time-step result is a sample which is a distribution. The third LSTM encodes these distributions for the last one to attend over and reconstruct y as $\hat{y}$. We now give the details of these parts.

3.3.1 Encoding y

The first LSTM of the encoder half of the model reads the sequence y, represented as a sequence of one-hot vectors over the vocabulary Σ_y, using a bidirectional RNN into a sequence of vectors $h^y_{1:L_y}$ where L_y is the sequence length of y,

$$h^y_t = \left(f^{\rightarrow}_y(y_t, h^{y,\rightarrow}_{t-1}); f^{\leftarrow}_y(y_t, h^{y,\leftarrow}_{t+1})\right), \tag{3.1}$$

where $f^{\rightarrow}_y$, $f^{\leftarrow}_y$ are non-linear functions applied at each time step to the current token y_t and their recurrent states $h^{y,\rightarrow}_{t-1}$, $h^{y,\leftarrow}_{t+1}$, respectively.

Both the forward and backward functions project the one-hot vector into a dense vector via an embedding matrix, which serves as input to an LSTM.

3.3.2 Predicting a Latent Sequence $\tilde{x}$

Subsequently, we wish to predict x. Predicting a discrete sequence of symbols through draws from multinomial distributions over a vocabulary is not an option, as we would not be able to backpropagate through this discrete choice. Marginalizing over the possible latent strings is intractable, and estimating the gradient through naïve Monte Carlo methods would be a prohibitively high variance process because the number of strings is exponential in the maximum length (which we would have to manually specify) with the vocabulary size as base. To allow backpropagation, we instead predict a sequence of distributions $\tilde{x}$ over the symbols of Σ_x with an RNN attending over $h^y = h^y_{1:L_y}$, which will later serve to reconstruct y:

$$\tilde{x} = q(\tilde{x}|y) = \prod_{t=1}^{L_x} q(\tilde{x}_t|\{\tilde{x}_1, \cdots, \tilde{x}_{t-1}\}, h^y) \tag{3.2}$$

where $q(x|y)$ models the mapping $y \mapsto x$. We define $q(\tilde{x}_t|\{\tilde{x}_1, \cdots, \tilde{x}_{t-1}\}, h^y)$ in the following way.

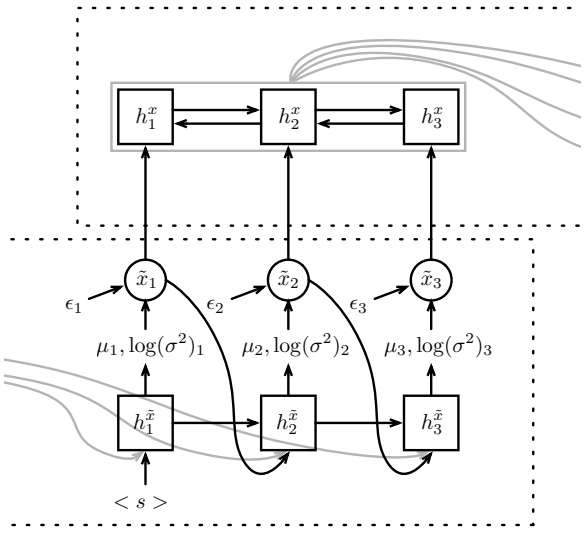


Figure 3.2: Unsupervised case of the SEQ4 model.

Let the vector $\tilde{x}_t$ be a distribution over the vocabulary Σ_x drawn from a logistic-normal distribution[1], the parameters of which, $\mu_t, \log(\sigma^2)_t \in \mathbb{R}^{|\Sigma_x|}$, are predicted by attending by an LSTM attending over the outputs of the encoder (Equation 3.2), where $|\Sigma_x|$ is the size of the vocabulary Σ_x. The use of a logistic normal distribution serves to regularize the model in the semi-supervised learning regime, which is described at the end of this section. Formally, this process, depicted in Figure 3.2, is as follows:

$$h_t^{\tilde{x}} = f_{\tilde{x}}(\tilde{x}_{t-1}, h_{t-1}^{\tilde{x}}, h^y) \tag{3.3}$$

$$\mu_t, \log(\sigma_t^2) = l(h_t^{\tilde{x}}) \tag{3.4}$$

$$\epsilon \sim \mathcal{N}(0, I) \tag{3.5}$$

$$\gamma_t = \mu_t + \sigma_t \epsilon \tag{3.6}$$

$$\tilde{x}_t = \mathrm{softmax}(\gamma_t) \tag{3.7}$$

where the $f_{\tilde{x}}$ function is an LSTM and l a linear transformation to $\mathbb{R}^{2|\Sigma_x|}$. We use the reparameterization trick from Kingma and Welling (2014) to draw from the logistic normal, allowing us to backpropagate through the sampling process. In particular, we reparameterized the distribution in terms of deterministicly predicted, differentiable $\mu_t, \log(\sigma_t^2)$ and a sample ϵ with no weights to train.

[1]The logistic-normal distribution is the exponentiated and normalized (i.e. taking softmax) normal distribution.

Using the reparameterization trick we were able train our model with continuous latent variables drawn from the logistic-normal distribution. An alternative to this setup would be to use the Concrete Distribution (Maddison et al., 2017) with discrete latent variables.

3.3.3 Encoding x

Moving on to the decoder part of our model, in the third LSTM, we embed[2] and encode $\tilde{x}$:

$$h_t^x = \left(f_x^{\rightarrow}(\tilde{x}_t, h_{t-1}^{x,\rightarrow}); f_x^{\leftarrow}(\tilde{x}_t, h_{t+1}^{x,\leftarrow})\right) \tag{3.8}$$

When x is observed (during supervised training, or when making predictions) instead of the distribution $\tilde{x}_t$ we use the one-hot encoded word x_t as the input to this part of the model.

3.3.4 Reconstructing y

In the final LSTM, we decode into y:

$$p(\hat{y}|\tilde{x}) = \prod_{t=1}^{L_y} p(\hat{y}_t|\{\hat{y}_1, \cdots, \hat{y}_{t-1}\}, h^x) \tag{3.9}$$

Equation 3.9 is implemented as an LSTM attending over h^x producing a sequence of symbols $\hat{y}$ based on recurrent states $h^{\hat{y}}$, aiming to reproduce input y:

$$h_t^{\hat{y}} = f_{\hat{y}}(\hat{y}_{t-1}, h_{t-1}^{\hat{y}}, h^x) \tag{3.10}$$

$$\hat{y}_t \sim \text{softmax}(l'(h_t^{\hat{y}})) \tag{3.11}$$

where $f_{\hat{y}}$ is the non-linear function, and the actual probabilities are given by a softmax function after a linear transformation l' of $h^{\hat{y}}$. At training time, rather than $\hat{y}_{t-1}$ we feed the ground truth y_{t-1}.

3.3.5 Loss function

The complete model described in this section gives a reconstruction function $y \mapsto \hat{y}$. We define a loss on this reconstruction which accommodates the unsupervised case, where x is not observed in the training data, and the supervised case, where (x, y) pairs are available. Together, these allow us to train the SEQ4 model in a semi-supervised setting, which experiments will show provides some benefits over a purely supervised training regime.

[2]Multiplying the distribution over words and an embedding matrix averages the word embedding of the entire vocabulary weighted by their probabilities.

Unsupervised case When x isn't observed, the loss we minimize during training is the reconstruction loss on y, expressed as the negative log-likelihood $NLL(\hat{y}, y)$ of the true labels y relative to the predictions $\hat{y}$. Note, this loss is differentiable with respect to all parameters of both the encoder and the decoder, and includes sampling of ϵ (see Equation 3.6). To this, we add as a regularizing term the KL divergence $KL[q(\gamma|y)\|p(\gamma)]$ which effectively penalizes the mean and variance of $q(\gamma|y)$ from diverging from those of a prior $p(\gamma)$, which we model as a diagonal Gaussian $\mathcal{N}(0, I)$. This has the effect of smoothing the logistic normal distribution from which we draw the distributions over symbols of x, guarding against overfitting of the latent distributions over x to symbols seen in the supervised case discussed below. The unsupervised loss is therefore formalized as

$$\mathcal{L}_{unsup} = NLL(\hat{y}, y) + \alpha KL[q(\gamma|y)\|p(\gamma)] \tag{3.12}$$

with regularizing factor α is tuned on validation, and

$$KL[q(\gamma|y)\|p(\gamma)] = \sum_{i=1}^{L_x} KL[q(\gamma_i|y)\|p(\gamma)]. \tag{3.13}$$

We use a closed form of these individual KL divergences, described by Kingma and Welling (2014).

Supervised case When x is observed, we additionally minimize the prediction loss on x, expressed as the negative log-likelihood $NLL(\tilde{x}, x)$ of the true labels x relative to the predictions $\tilde{x}$, and do not impose the KL loss. The supervised loss is thus

$$\mathcal{L}_{sup} = NLL(\tilde{x}, x) + NLL(\hat{y}, y) \tag{3.14}$$

In both the supervised and unsupervised case, because of the continuous relaxation on generating $\tilde{x}$ and the reparameterization trick, the gradient of the losses with regard to the model parameters is well defined throughout SEQ4.

Semi-supervised training and inference We train with a weighted combination of the supervised and unsupervised losses described above. In particular, we alternated optimizing supervised and unsupervised minibatched with a tunable weight for the unsupervised loss. Once trained, we simply use the $x \mapsto y$ decoder segment of the model to predict y from sequences of symbols x represented as one-hot vectors. When the decoder is trained without the encoder in a fully supervised manner, it serves as our supervised sequence-to-sequence baseline model under the name S2S.

3.4 Tasks and Data Generation

We apply our model to three tasks outlined in this section. Moreover, we explain how we generated additional unsupervised training data for two of these tasks. Examples from all datasets are in Table 3.1.

3.4.1 GeoQuery

The first task we consider is the prediction of a query on the GEO corpus which is a frequently used benchmark for semantic parsing. The corpus contains 880 questions about US geography together with executable queries representing those questions. We follow the approach established by Zettlemoyer and Collins (2005) and split the corpus into 600 training and 280 test cases. Following common practice, we augment the dataset by referring to the database during training and test time. In particular, we use the database to identify and anonymize variables (cities, states, countries and rivers) following the method described in Dong and Lapata (2016).

Most prior work on the GEO corpus relies on standard semantic parsing methods together with custom heuristics or pipelines for this corpus. The recent paper by Dong and Lapata (2016) is of note, as it uses a sequence-to-sequence model for training which is the unidirectional equivalent to S2S, and also to the decoder part of our SEQ4 network.

3.4.2 Open Street Maps

The second task we tackle with our model is the NLMAPS dataset by Haas and Riezler (2016). The dataset contains 1,500 training and 880 testing instances of natural language questions with corresponding machine readable queries over the geographical OpenStreetMap database. The dataset contains natural language question in both English and German but we focus only on single language (English) semantic parsing, similar to the first task in Haas and Riezler (2016). We use the data as it is, with the only pre-processing step being the tokenization of both natural language and query form[3].

[3]We removed quotes, added spaces around `()`, and separated the question mark from the last word in each question.

3.4.3 Navigational Instructions to Actions

The SAIL corpus and task were developed to train agents to follow free-form navigational route instructions in a maze environment (MacMahon et al., 2006; Chen and Mooney, 2011). It consists of a small number of mazes containing features such as objects, wall and floor types. These mazes come together with a large number of human instructions paired with the required actions[4] to reach the goal state described in those instructions.

We use the sentence-aligned version of the SAIL route instruction dataset containing 3,236 sentences (Chen and Mooney, 2011). Following previous work, we accept an action sequence as correct if and only if the final position and orientation exactly match those of the gold data. We do not perform any pre-processing on this dataset.

3.4.4 Data Generation

As argued earlier, we are focusing on tasks where aligned data is sparse and expensive to obtain, while it should be cheap to get unsupervised, monomodal data, from a distribution similar to the unaligned training data. Albeit that is a reasonable assumption for real world data, the datasets considered have no such component, thus the approach taken here is to generate random database queries or maze paths, i.e. the machine readable side of the data, and train a semi-supervised model. The alternative not explored here would be to generate natural language questions or instructions instead, but that is more difficult to achieve without human intervention. For this reason, we generate the machine readable side of the data for GEOQUERY and SAIL tasks[5].

For GEOQUERY, we fit a 3-gram Kneser-Ney (Chen and Goodman, 1999) model to the queries in the training set and sample about 7 million queries from it. We ensure that the sampled queries are different from the training queries, but do not enforce validity. This intentionally simplistic approach is to demonstrate the applicability of our model.

The SAIL dataset has only three mazes. We added a fourth one and over 150k random paths, including duplicates. The new maze is larger (21×21 grid) than the existing ones, and seeks to approximately replicate the key statistics of the other three mazes (maximum corridor length, distribution of objects, etc). Paths within that maze are created by randomly sampling start and end positions.

[4]There are four actions: `LEFT`, `RIGHT`, `GO`, `STOP`.

[5]Our randomly generated unsupervised datasets can be downloaded from `http://deepmind.com/publications`

3.5 Experiments

We evaluate our model on the three tasks in multiple settings. First, we establish a supervised baseline to compare the S2S model with prior work. Next, we train our SEQ4 model in a semi-supervised setting on the entire dataset with the additional monomodal training data described in the previous section.

Finally, we perform an "ablation" study where we discard some of the training data and compare S2S to SEQ4. S2S is trained solely on the reduced parallel data in a supervised manner, while SEQ4 is once again trained semi-supervised on the same reduced data plus the machine readable part of the discarded data (SEQ4-) or on the extra machine readable generated data (SEQ4+), the latter of which is much larger but not exactly from the original training data distribution (as it is generated).

Training We train the model using standard gradient descent methods. As none of the datasets used here contain development sets, we tune hyperparameters by cross-validating on the training data. In the case of the SAIL corpus we train on three folds (two mazes for training and validation, one for test each) and report weighted results across the folds following prior work (Mei et al., 2016). All numbers reported were tuned separately using Google Vizier (Golovin et al., 2017) for supervised and unsupervised batch sized, the weighting factor between the two losses, the KL weight α, learning rate, hidden sizes, dropout, and number of optimization steps. The models were further trained three times with the best hyperparameters and averaged.

3.5.1 GeoQuery

The evaluation metric for GEOQUERY is the accuracy of exactly predicting the machine readable query. As results in Table 3.3 show, our supervised S2S baseline model performs slightly better than the comparable model by Dong and Lapata (2016). The semi-supervised SEQ4 model with the additional generated queries improves on it further.

The ablation study in Table 3.4 demonstrates a widening gap between supervised and semi-supervised as the amount of labelled training data gets smaller. This suggests that our model can leverage unlabelled data even when only small amount of labelled data is available.

[6] Jia and Liang (2016) used hand crafted grammars to generate additional supervised training data.

Model	Accuracy
Zettlemoyer and Collins (2005)	79.3
Zettlemoyer and Collins (2007)	86.1
Liang et al. (2013)	87.9
Kwiatkowski et al. (2011)	88.6
Zhao and Huang (2014)	88.9
Kwiatkowski et al. (2013)	89.0
Dong and Lapata (2016)	84.6
Jia and Liang (2016)[6]	89.3
S2S	86.5
SEQ4	87.3

Table 3.3: Non-neural and neural model results on GEOQUERY using the train/test split from (Zettlemoyer and Collins, 2005).

Sup. data	S2S	SEQ4-	SEQ4+
5%	21.9	30.1	26.2
10%	39.7	42.1	42.1
25%	62.4	70.4	67.1
50%	80.3	81.2	80.4
75%	85.3	84.1	85.1
100%	86.5	86.5	87.3

Table 3.4: Results of the GEOQUERY ablation study. The supervised dataset contains 600 training and 280 test examples. The unsupervised data contains about 7 million non-validated queries.

Model	Accuracy
Haas and Riezler (2016)	68.30
S2S	78.03

Table 3.5: Results on the NLMAPS corpus.

Sup. data	S2S	SEQ4-
5%	3.22	3.74
10%	17.61	17.12
25%	33.74	33.50
50%	49.52	53.72
75%	66.93	66.45
100%	78.03	78.03

Table 3.6: Results of the NLMAPS ablation study.

3.5.2 Open Street Maps

We report results for the NLMAPS corpus in Table 3.5, comparing the supervised S2S model to the results posted by Haas and Riezler (2016). While their model used a semantic parsing pipeline including alignment, stemming, language modelling and CFG inference, the strong performance of the S2S model demonstrates the strength of fairly vanilla attention-based sequence-to-sequence models. It should be pointed out that the previous work reports the number of correct answers when queries were executed against the dataset, while we evaluate on the strict accuracy of the generated queries. While we expect these numbers to be nearly equivalent, our evaluation is strictly harder as it does not allow for reordering of query arguments and similar relaxations.

We investigate the SEQ4 model only via the ablation study in Table 3.6 and find little gain through the semi-supervised objective. Our attempt at cheaply generating unsupervised data for this task was not successful, likely due to the complexity of the underlying database.

3.5.3 Navigational Instructions to Actions

Model extension The experiments for the SAIL task differ slightly from the other two tasks in that the language input does not suffice for choosing an action. While a simple instruction such as *'turn left'* can easily be translated into the action sequence `LEFT-STOP`,

Model	Accuracy
Chen and Mooney (2011)	54.40
Kim and Mooney (2012)	57.22
Andreas and Klein (2015)	59.60
Kim and Mooney (2013)	62.81
Artzi et al. (2014)	64.36
Artzi and Zettlemoyer (2013)	65.28
Mei et al. (2016)	69.98
S2S	58.60
SEQ4	63.25

Table 3.7: Results on the SAIL corpus.

more complex instructions such as *'Walk forward until you see a lamp'* require knowledge of the agent's position in the maze.

To accomplish this we modify the model as follows. First, when encoding action sequences, we concatenate each action with a representation of the maze at the given position, representing the maze-state akin to Mei et al. (2016) with a bag-of-features vector. Second, when decoding action sequences, the RNN outputs an action which is used to update the agent's position and the representation of that new position is fed into the RNN as its next input.

Training regime We cross-validate over the three mazes in the dataset and report overall results weighted by test size (cf. Mei et al. (2016)). Both our supervised and semi-supervised model perform worse than the state-of-the-art (see Table 3.7), but the latter enjoys a comfortable margin over the former. As the S2S model broadly reimplements the work of Mei et al. (2016), we put the discrepancy in performance down to the particular design choices that we did not follow in order to keep the model here as general as possible and comparable across tasks.

The ablation studies (Table 3.8) show little gain for the semi-supervised approach when only using data from the original training set, but substantial improvement with the additional unsupervised data.

Sup. data	S2S	SEQ4-	SEQ4+
5%	37.79	41.48	43.44
10%	40.77	41.26	48.67
25%	43.76	43.95	51.19
50%	48.01	49.42	55.97
75%	48.99	49.20	57.40
100%	49.49	49.49	58.28

Table 3.8: Results of the SAIL ablation study. Results are from models trained on *L* and *Jelly* maps, tested on *Grid* only, hence the discrepancy between the 100% result and S2S in Table 3.7.

3.6 Discussion

Supervised training The prediction accuracies of our supervised baseline S2S model are mixed with respect to prior results on their respective tasks. For GEOQUERY, S2S has a substantial lead over the most similar model from the literature (Dong and Lapata, 2016), mostly due to the fact that y and x are encoded with bidirectional LSTMs. With a unidirectional LSTM we get similar results to theirs.

On the SAIL corpus, S2S performs worse than the state of the art. As the models are broadly equivalent we attribute this difference to a number of task-specific choices and optimizations[7] made in Mei et al. (2016) which we did not reimplement for the sake of using a common model across all three tasks.

For NLMAPS, S2S performs much better than the state-of-the-art, exceeding the previous best result by 11% despite a very simple tokenization method and a lack of any form of entity anonymization.

Semi-supervised training In both the case of GEOQUERY and the SAIL task we found the semi-supervised model to convincingly outperform the fully supervised model. The effect was particularly notable in the case of the SAIL corpus, where performance increased from 58.60% accuracy to 63.25% (see Table 3.7). It is worth remembering that the supervised training regime consists of three folds of tuning on two maps with subsequent testing on the third map, which carries a risk of overfitting to the training maps. The introduction of the fourth unsupervised map clearly mitigates this effect. Table 3.2 shows some examples

[7] In particular we don't use beam search and ensembling.

of unsupervised logical forms being transformed into natural language, which demonstrate how the model can learn to sensibly ground unsupervised data.

Ablation performance The experiments with additional unsupervised data prove the feasibility of our approach and clearly demonstrate the usefulness of the SEQ4 model for the general class of sequence-to-sequence tasks where supervised data is hard to come by. To analyse the model further, we also look at the performance of both S2S and SEQ4 when reducing the amount of supervised training data available to the model. We compare three settings: the supervised S2S model with reduced training data, SEQ4- which uses the removed training data in an unsupervised fashion (throwing away the natural language) and SEQ4+ which uses the randomly generated unsupervised data described in Section 3.4. The S2S model behaves as expected on all three tasks, its performance dropping with the size of the training data. The performance of SEQ4- and SEQ4+ requires more analysis.

In the case of GEOQUERY, having unlabelled data from the true distribution (SEQ4-) is a good thing when there is enough of it, as clearly seen when only 5% of the original dataset is used for supervised training and the remaining 95% is used for unsupervised training. The gap shrinks as the amount of supervised data is increased, which is as expected. On the other hand, using a large amount of extra, generated data from an approximating distribution (SEQ4+) does not help as much initially when compared with the unsupervised data from the true distribution. However, as the size of the unsupervised dataset in SEQ4- becomes the bottleneck this gap closes and eventually the model trained on the extra data achieves higher accuracy.

For the SAIL task the semi-supervised models do better than the supervised results throughout, with the model trained on randomly generated additional data consistently outperforming the model trained only on the original data. This gives further credence to the risk of overfitting to the training mazes already mentioned above.

Finally, in the case of the NLMAPS corpus, the semi-supervised approach does not appear to help much at any point during the ablation. These indistinguishable results are likely due to the task's complexity, causing the ablation experiments to either have too little supervised data to sufficiently ground the latent space to make use of the unsupervised data, or in the higher percentages then too little unsupervised data to meaningfully improve the model.

3.7 Summary

We described a method for augmenting a supervised sequence transduction objective with an autoencoding objective, thereby enabling semi-supervised training where previously a scarcity of aligned data might have held back model performance. Across multiple semantic parsing tasks we demonstrated the effectiveness of this approach, improving model performance by training on randomly generated unsupervised data in addition to the original data.

Chapter 4

Deep Learning for Reading Comprehension

Chapter Abstract

Teaching machines to read natural language documents remains an elusive challenge. Machine reading systems can be tested on their ability to answer questions posed on the contents of documents that they have seen, but until now large scale training and test datasets have been missing for this type of evaluation. In this work we define a new methodology that resolves this bottleneck and provides large scale supervised reading comprehension data. This allows us to develop a class of attention based deep neural networks that learn to read real documents and answer complex questions with minimal prior knowledge of language structure.

4.1 Introduction

Progress on the path from shallow bag-of-words information retrieval algorithms to machines capable of reading and understanding documents has been slow. Traditional approaches to machine reading and comprehension have been based on either hand engineered grammars (Riloff and Thelen, 2000), or information extraction methods of detecting

The work presented in this chapter was originally presented in Hermann et al. (2015). I contributed to the task setup, running of the experiments, model analysis, writing, and I implemented and optimized the attention-based neural models.

predicate argument triples that can later be queried as a relational database (Poon et al., 2010). Supervised machine learning approaches have largely been absent from this space due to both the lack of large scale training datasets, and the difficulty in structuring statistical models flexible enough to learn to exploit document structure.

While obtaining supervised natural language reading comprehension data has proved difficult, some researchers have explored generating synthetic narratives and queries (Weston et al., 2015b; Sukhbaatar et al., 2015). Such approaches allow the generation of almost unlimited amounts of supervised data and enable researchers to isolate the performance of their algorithms on individual simulated phenomena. Work on such data has shown that neural network based models hold promise for modelling reading comprehension, something that we will build upon here. Historically, however, many similar approaches in Computational Linguistics have failed to manage the transition from synthetic data to real environments, as such closed worlds inevitably fail to capture the complexity, richness, and noise of natural language (Winograd, 1972b).

In this work we seek to directly address the lack of real natural language training data by introducing a novel approach to building a supervised reading comprehension data set. We observe that summary and paraphrase sentences, with their associated documents, can be readily converted to context–query–answer triples using simple entity detection and anonymization algorithms. Using this approach we have collected two new corpora of roughly a million news stories with associated queries from the CNN and Daily Mail websites.

We demonstrate the efficacy of our new corpora by building novel deep learning models for reading comprehension. These models draw on recent developments for incorporating attention mechanisms into recurrent neural network architectures (Bahdanau et al., 2015; Mnih et al., 2014; Gregor et al., 2015; Sukhbaatar et al., 2015). This allows a model to focus on the aspects of a document that it believes will help it answer a question, and also allows us to visualizes its inference process. We compare these neural models to a range of baselines and heuristic benchmarks based upon a traditional frame semantic analysis provided by a state-of-the-art natural language processing (NLP) pipeline. Our results indicate that the neural models achieve a higher accuracy, and do so without any specific encoding of the document or query structure.

	CNN			Daily Mail		
	train	valid	test	train	valid	test
# months	95	1	1	56	1	1
# documents	90,266	1,220	1,093	196,961	12,148	10,397
# queries	380,298	3,924	3,198	879,450	64,835	53,182
Max # entities	527	187	396	371	232	245
Avg # entities	26.4	26.5	24.5	26.5	25.5	26.0
Avg # tokens	762	763	716	813	774	780
Vocab size	118,497			208,045		

Table 4.1: Corpus statistics. Articles were collected starting in April 2007 for CNN and June 2010 for the Daily Mail, both until the end of April 2015. Validation data is from March, test data from April 2015. Articles of over 2000 tokens and queries whose answer entity did not appear in the context were filtered out.

4.2 Supervised Training Data for Reading Comprehension

The reading comprehension task naturally lends itself to a formulation as a supervised learning problem. Specifically we seek to estimate the conditional probability $p(a|c, q)$, where c is a context document, q a query relating to that document, and a the answer to that query. For a focused evaluation we wish to be able to exclude additional information, such as world knowledge gained from co-occurrence statistics, in order to test a model's core capability to detect and understand the linguistic relationships between entities in the context document.

Such an approach requires a large training corpus of document–query–answer triples and until now such corpora have been limited to hundreds of examples and thus mostly of use only for testing (Richardson et al., 2013b). This limitation has meant that most work in this area has taken the form of unsupervised approaches which use templates or syntactic/semantic analysers to extract relation tuples from the document to form a knowledge graph that can be queried.

Here we propose a methodology for creating real-world, large scale supervised training data for learning reading comprehension models. Inspired by work in summarization (Svore et al., 2007; Woodsend and Lapata, 2010), we create two machine reading corpora by exploiting online newspaper articles and their matching summaries. We have collected

93k articles from the CNN[1] and 220k articles from the Daily Mail[2] websites. Both news providers supplement their articles with a number of bullet points, summarizing aspects of the information contained in the article. Of key importance is that some of these summary points are abstractive and do not simply copy sentences from the documents. We construct a corpus of document–query–answer triples by turning these bullet points into Cloze (Taylor, 1953) style questions (i.e. fill-in-a-word sentences) by replacing one entity at a time with a placeholder. This results in a combined corpus of roughly 1M data points (Table 4.1). Code to replicate our datasets—and to apply this method to other sources—is available online[3].

4.2.1 Entity Replacement and Permutation

Note that the focus of this chapter is to provide a corpus for evaluating a model's ability to read and comprehend a single document, not world knowledge or co-occurrence. To understand that distinction consider for instance the following Cloze form queries (created from headlines in the Daily Mail validation set): *a*) The hi-tech bra that helps you beat breast **X**; *b*) Could Saccharin help beat **X** ?; *c*) Can fish oils help fight prostate **X** ? An n-gram language model trained on the Daily Mail would easily correctly predict that (**X** = *cancer*), regardless of the contents of the context document, simply because this is a very frequently cured entity in the Daily Mail corpus.

To prevent such degenerate solutions and create a focused task we anonymize and randomize our corpora with the following procedure, *a*) use a coreference system to establish coreferents in each data point; *b*) replace all entities with abstract entity markers according to coreference; *c*) randomly permute these entity markers whenever a data point is loaded.

Compare the original and anonymized version of the example in Table 4.2. Clearly a human reader can answer both queries correctly. However in the anonymized 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 our two corpora truly measures reading comprehension capability. Naturally a production system would

[1] `www.cnn.com`
[2] `www.dailymail.co.uk`
[3] `http://www.github.com/deepmind/rc-data/`

Original Version	Anonymized Version
Context	
The BBC producer allegedly struck by Jeremy Clarkson will not press charges against the "Top Gear" host, his lawyer said Friday. Clarkson, who hosted one of the most-watched television shows in the world, was dropped by the BBC Wednesday after an internal investigation by the British broadcaster found he had subjected producer Oisin Tymon "to an unprovoked physical and verbal attack." ...	the *ent381* producer allegedly struck by *ent212* will not press charges against the " *ent153* " host , his lawyer said friday . *ent212* , who hosted one of the most - watched television shows in the world , was dropped by the *ent381* wednesday after an internal investigation by the *ent180* broadcaster found he had subjected producer *ent193* " to an unprovoked physical and verbal attack . " ...
Query	
Producer **X** will not press charges against Jeremy Clarkson, his lawyer says.	producer **X** will not press charges against *ent212* , his lawyer says .
Answer	
Oisin Tymon	*ent193*

Table 4.2: Original and anonymized version of a data point from the Daily Mail validation set. The anonymized entity markers are constantly permuted during training and testing.

benefit from using all available information sources, such as clues through language and co-occurrence statistics.

Table 4.3 gives an indication of the difficulty of the task, showing how frequent the correct answer is contained in the top N entity markers in a given document. Note that our models don't distinguish between entity markers and regular words. This makes the task harder and the models more general.

4.3 Models

So far we have motivated the need for better datasets and tasks to evaluate the capabilities of machine reading models. We proceed by describing a number of baselines, benchmarks and new models to evaluate against this paradigm. We define two simple baselines, the majority baseline (`maximum frequency`) picks the entity most frequently observed in the context document, whereas the exclusive majority (`exclusive frequency`) chooses the entity most frequently observed in the context but not observed in the query. The idea behind this exclusion is that the placeholder is unlikely to be mentioned twice in a single Cloze form query.

Top N	Cumulative %	
	CNN	Daily Mail
1	30.5	25.6
2	47.7	42.4
3	58.1	53.7
5	70.6	68.1
10	85.1	85.5

Table 4.3: Percentage of time that the correct answer is contained in the top N most frequent entities in a given document.

4.3.1 Symbolic Matching Models

Traditionally, a pipeline of NLP models has been used for attempting question answering, that is models that make heavy use of linguistic annotation, structured world knowledge and semantic parsing and similar NLP pipeline outputs. Building on these approaches, we define a number of NLP-centric models for our machine reading task.

Frame-Semantic Parsing Frame-semantic parsing attempts to identify predicates and their arguments, allowing models access to information about "who did what to whom". Naturally this kind of annotation lends itself to being exploited for question answering. We develop a benchmark that makes use of frame-semantic annotations which we obtained by parsing our model with a state-of-the-art frame-semantic parser (Das et al., 2013; Hermann et al., 2014). As the parser makes extensive use of linguistic information we run these benchmarks on the unanonymized version of our corpora. There is no significant advantage in this as the frame-semantic approach used here does not possess the capability to generalize through a language model beyond exploiting one during the parsing phase. Thus, the key objective of evaluating machine comprehension abilities is maintained. Extracting entity-predicate triples—denoted as (e_1, V, e_2)—from both the query q and context document d, we attempt to resolve queries using a number of rules with an increasing recall/precision trade-off as follows (Table 4.4).

For reasons of clarity, we pretend that all PropBank triples are of the form (e_1, V, e_2). In practice, we take the argument numberings of the parser into account and only compare like with like, except in cases such as the permuted frame rule, where ordering is relaxed. In the case of multiple possible answers from a single rule, we randomly choose one.

	Strategy	Pattern $\in q$	Pattern $\in d$	Example (Cloze / Context)
1	Exact match	(p, V, y)	$(\boldsymbol{x}, V, y)$	X loves Suse / **Kim** loves Suse
2	be.01.V match	$(p, \textit{be.01.V}, y)$	$(\boldsymbol{x}, \textit{be.01.V}, y)$	X is president / **Mike** is president
3	Correct frame	(p, V, y)	$(\boldsymbol{x}, V, z)$	X won Oscar / **Tom** won Academy Award
4	Permuted frame	(p, V, y)	$(y, V, \boldsymbol{x})$	X met Suse / Suse met **Tom**
5	Matching entity	(p, V, y)	$(\boldsymbol{x}, Z, y)$	X likes candy / **Tom** loves candy
6	Back-off strategy	*Pick the most frequent entity from the context that doesn't appear in the query*		

Table 4.4: Resolution strategies using PropBank triples. $\boldsymbol{x}$ denotes the entity proposed as answer, V is a fully qualified PropBank frame (e.g. *give.01.V*). Strategies are ordered by precedence and answers determined accordingly. This heuristic algorithm was iteratively tuned on the validation data set.

Word Distance Benchmark We consider another baseline that relies on word distance measurements. Here, we align the placeholder of the Cloze form question with each possible entity in the context document and calculate a distance measure between the question and the context around the aligned entity. This score is calculated by summing the distances of every word in q to their nearest aligned word in d, where alignment is defined by matching words either directly or as aligned by the coreference system. We tune the maximum penalty per word ($m = 8$) on the validation data.

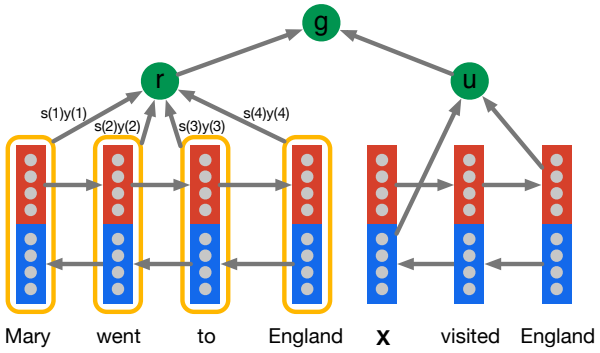
4.3.2 Neural Network Models

Neural networks have successfully been applied to a range of tasks in NLP. This includes classification tasks such as sentiment analysis (Kalchbrenner et al., 2014) or POS tagging (Collobert et al., 2011), as well as generative problems such as language modelling or machine translation (Sutskever et al., 2014b). We propose three neural models for estimating the probability of word type a from document d answering query q:

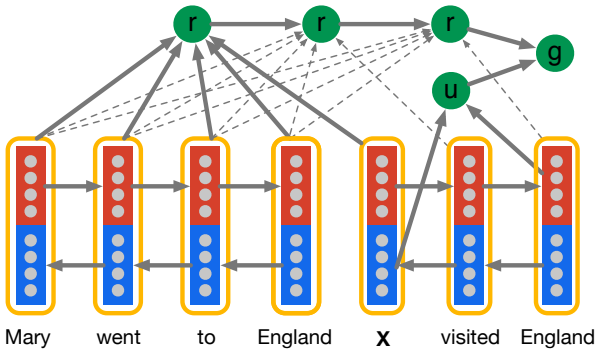
$$p(a|d, q) \propto \exp\left(W(a)g(d, q)\right), \quad \text{s.t. } a \in V,$$

where V is the vocabulary[4], and $W(a)$ indexes row a of weight matrix $W \in \mathbb{R}^{|V| \times d_{\text{h}}}$, where d_{h} is the hidden size in the network, and through a slight abuse of notation word types double as indexes. Note that we do not privilege entities or variables, the model must learn to differentiate these in the input sequence. The function $g(d, q)$ returns a vector embedding of a document and query pair.

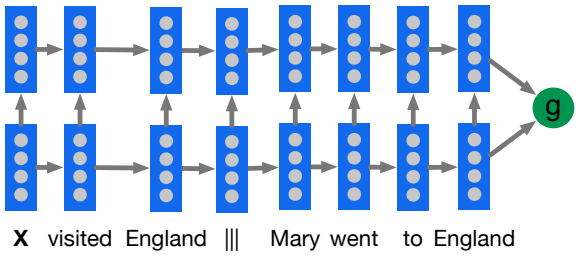
[4]The vocabulary includes all the word types in the documents, questions, the entity maskers, and the question unknown entity marker.



(a) Attentive Reader.



(b) Impatient Reader.



(c) A two layer Deep LSTM Reader with the question encoded before the document.

Figure 4.1: Document and query embedding models. Shows the hidden states of the forward/backward LSTMs; for document for the Attentive Reader, and document and question for the Impatient Reader we concatenate the forward and backward hidden states at each time step (shown in yellow).

The Deep LSTM Reader Long short-term memory (LSTM, (Hochreiter and Schmidhuber, 1997)) networks have recently seen considerable success in tasks such as machine translation and language modelling (Sutskever et al., 2014b). When used for translation, Deep LSTMs (Graves, 2012) have shown a remarkable ability to embed long sequences into a vector representation which contains enough information to generate a full translation in another language. Our first neural model for reading comprehension tests the ability of Deep LSTM encoders to handle significantly longer sequences. We feed our documents one word at a time into a Deep LSTM encoder, after a delimiter we then also feed the query into the encoder. Alternatively we also experiment with processing the query then the document. The result is that this model processes each document query pair as a single long sequence. Given the embedded document and query the network predicts which token in the document answers the query. See Figure 4.1c.

We employ a Deep LSTM cell with skip connections from each input $x(t)$ to every hidden layer, and from every hidden layer to the output $y(t)$, as described in Section 2.2.2. Thus our Deep LSTM Reader is defined by $g^{\text{LSTM}}(d, q) = y(|d| + |q|)$ with input $x(t)$ the concatenation of d and q separated by the delimiter $|||$.

The Attentive Reader The Deep LSTM Reader must propagate dependencies over long distances in order to connect queries to their answers. The fixed width hidden vector forms a bottleneck for this information flow that we propose to circumvent using an attention mechanism inspired by recent results in translation and image recognition (Bahdanau et al., 2015; Mnih et al., 2014). This attention model first encodes the document and the query using separate bidirectional single layer LSTMs (Graves, 2012).

We denote the outputs of the forward and backward LSTMs as $\overrightarrow{y}(t) \in \mathbb{R}^{d_{\text{h}}}$ and $\overleftarrow{y}(t) \in \mathbb{R}^{d_{\text{h}}}$ respectively. The encoding u of a query of length $|q|$ is formed by the concatenation of the final forward and backward outputs, $u = \overrightarrow{y_q}(|q|) \mid\mid \overleftarrow{y_q}(1)$.

For the document the composite output for each token at position t is $y_d(t) = \overrightarrow{y_d}(t) \mid\mid \overleftarrow{y_d}(t)$, $y_d(t) \in \mathbb{R}^{2d_{\text{h}}}$. The representation r of the document d is formed by a weighted sum of these output vectors. These weights are interpreted as the degree to which the network attends to

a particular token in the document when answering the query:

$$
\begin{aligned}
m(t) &= \tanh\left(W_{ym} y_d(t) + W_{um} u\right), \\
s(t) &\propto \exp\left(\mathrm{w}_{ms}^{\intercal} m(t)\right), \\
r &= y_d s,
\end{aligned}
$$

where we are interpreting y_d as a matrix with each column being the composite representation $y_d(t) \in \mathbb{R}^{2d_{\mathrm{h}}}$ of document token t, $W_{ym}, W_{um} \in \mathbb{R}^{d_{\mathrm{h}} \times 2d_{\mathrm{h}}}$ and $w_{ms} \in \mathbb{R}^{d_{\mathrm{h}}}$ are linear transformations, and $\tanh, \exp$ are applied element wise. The variable $s(t)$ is the normalized attention at token t. Given this attention score the embedding of the document $r \in \mathbb{R}^{2d_{\mathrm{h}}}$ is computed as the weighted sum of the token embeddings. The model is completed with the definition of the joint document and query embedding via a non-linear combination

$$
g^{\mathrm{AR}}(d, q) = \tanh\left(W_{rg} r + W_{ug} u\right),
$$

where $W_{rg}, W_{ug} \in \mathbb{R}^{d_{\mathrm{h}} \times 2d_{\mathrm{h}}}$ and $\tanh$ is applied element wise.

The Attentive Reader (Figure 4.1a) can be viewed as a generalization of the application of Memory Networks to question answering (Weston et al., 2015b). That model employs an attention mechanism at the sentence level where each sentence is represented by a bag of embeddings. The Attentive Reader employs a finer grained token level attention mechanism where the tokens are embedded given their entire future and past context in the input document.

The Impatient Reader The Attentive Reader is able to focus on the passages of a context document that are most likely to inform the answer to the query. We can go further by equipping the model with the ability to reread from the document as each query token is read (see Figure 4.1b). At each token i of the query q the model computes a document representation vector $r(i)$ using the bidirectional embedding $y_q(i) = \overrightarrow{y_q}(i) \mid\mid \overleftarrow{y_q}(i) \in \mathbb{R}^{2d_{\mathrm{h}}}$:

$$
\begin{aligned}
m(i,t) &= \tanh\left(W_{dm} y_d(t) + W_{rm} r(i-1) + W_{qm} y_q(i)\right), \quad 1 \le i \le |q|, \\
s(i,t) &\propto \exp\left(\mathrm{w}_{ms}^{\intercal} m(i,t)\right), \\
r(0) &= \mathbf{r_0}, \quad r(i) = y_d^{\intercal} s(i) + \tanh\left(W_{rr} r(i-1)\right) \quad 1 \le i \le |q|,
\end{aligned}
$$

where $W_{dm}, W_{rm}, W_{qm} \in \mathbb{R}^{d_{\text{h}} \times 2d_{\text{h}}}$, $w_{ms} \in \mathbb{R}^{d_{\text{h}}}$, $W_{rr} \in \mathbb{R}^{2d_{\text{h}} \times 2d_{\text{h}}}$ are weights matrices, $\tanh, \exp$ are applied element wise, and $\mathbf{r_0} \in \mathbb{R}^{d_{\text{h}}}$ is a learned vector. The result is an attention mechanism that allows the model to recurrently accumulate information from the document as it sees each query token, ultimately outputting a final joint document query representation for the answer prediction,

$$g^{\text{IR}}(d, q) = \tanh\left(W_{rg} r(|q|) + W_{qg} u\right),$$

where $W_{rg}, W_{qg} \in \mathbb{R}^{d_{\text{h}} \times 2d_{\text{h}}}$ and $\tanh$ is applied element wise.

The Uniform Reader As a baseline, to quantify the contribution of the attention mechanism, we consider the Uniform Reader where we set all the $m(t)$ parameters to be equal.

4.4 Empirical Evaluation

Having described a number of models in the previous section, we next evaluate these models on our reading comprehension corpora. Our hypothesis is that neural models should in principle be well suited for this task. However, we argued that simple recurrent models such as the LSTM probably have insufficient expressive power for solving tasks that require complex inference. We expect that the attention-based models would therefore outperform the pure LSTM-based approaches.

Considering the second dimension of our investigation, the comparison of traditional versus neural approaches to NLP, we do not have a strong prior favouring one approach over the other. While numerous publications in the past few years have demonstrated neural models outperforming classical methods, it remains unclear how much of that is a side-effect of the language modelling capabilities intrinsic to any neural model for NLP. The entity anonymization and permutation aspect of the task presented here may end up levelling the playing field in that regard, favouring models capable of dealing with syntax rather than just semantics.

With these considerations in mind, the experimental part of this chapter is designed with a three-fold aim. First, we want to establish the difficulty of our machine reading task by applying a wide range of models to it. Second, we compare the performance of parse-based methods versus that of neural models. Third, within the group of neural models examined, we want to determine what each component contributes to the end performance; that is,

	CNN		Daily Mail	
	valid	test	valid	test
Maximum frequency	30.5	33.2	25.6	25.5
Exclusive frequency	36.6	39.3	32.7	32.8
Frame-semantic model	36.3	40.2	35.5	35.5
Word distance model	50.5	50.9	56.4	55.5
Deep LSTM Reader	55.0	57.0	63.3	62.2
Uniform Reader	39.0	39.4	34.6	34.4
Attentive Reader	61.6	63.0	**70.5**	**69.0**
Impatient Reader	**61.8**	**63.8**	69.0	68.0

Table 4.5: Accuracy of the models and benchmarks on the CNN and Daily Mail datasets. The Uniform Reader baseline sets all of the $m(t)$ parameters to be equal.

Model	Hidden Size	Learning Rate	Batch Size	Dropout
Uniform, CNN	256	5E-5	32	0.2
Attentive, CNN	256	5E-5	32	0.2
Impatient, CNN	256	5E-5	32	0.3
Uniform, Daily Mail	256	5E-5	32	0.2
Attentive, Daily Mail	256	2.5E-5	32	0.1
Impatient, Daily Mail	256	5E-5	32	0.1

Table 4.6: Model hyperparameters

we want to analyse the extent to which an LSTM can solve this task, and to what extent various attention mechanisms impact performance.

All model hyperparameters were tuned on the respective validation sets of the two corpora. For the Deep LSTM Reader, we consider hidden layer sizes $[64, 128, \underline{256}]$, depths $[1, \underline{2}, 4]$, initial learning rates $[1\text{E}{-3}, 5\text{E}{-4}, \underline{1\text{E}{-4}}, 5\text{E}{-5}]$, batch sizes $[16, \underline{32}]$ and dropout $[0.0, \underline{0.1}, 0.2]$. We evaluate two types of feeds. In the *cqa* setup we feed first the context document and subsequently the question into the encoder, while the *qca* model starts by feeding in the question followed by the context document. We report results on the best model (underlined hyperparameters, *qca* setup). For the attention models we consider hidden layer sizes $[64, 128, 256]$, single layer, initial learning rates $[1\text{E}{-4}, 5\text{E}{-5}, 2.5\text{E}{-5}, 1\text{E}{-5}]$, batch sizes $[8, 16, 32]$ and dropout $[0, 0.1, 0.2, 0.5]$. For all models we used asynchronous RmsProp (Tieleman and Hinton, 2012) with a momentum of 0.9 and a decay of 0.95. The precise hyperparameters used for the various attentive models are as in Table 4.6.

Our experimental results are in Table 4.5, with the Attentive and Impatient Readers performing best across both datasets.

Frame-semantic benchmark While the one frame-semantic model proposed in this paper is clearly a simplification of what could be achieved with annotations from an NLP pipeline, it does highlight the difficulty of the task when approached from a symbolic NLP perspective.

Two issues stand out when analysing the results in detail. First, the frame-semantic pipeline has a poor degree of coverage with many relations not being picked up by our PropBank parser as they do not adhere to the default predicate-argument structure. This effect is exacerbated by the type of language used in the highlights that form the basis of our datasets. The second issue is that the frame-semantic approach does not trivially scale to situations where several sentences, and thus frames, are required to answer a query. This was true for the majority of queries in the dataset.

Word distance benchmark More surprising perhaps is the relatively strong performance of the word distance benchmark, particularly relative to the frame-semantic benchmark, which we had expected to perform better. Here, again, the nature of the datasets used can explain aspects of this result. Where the frame-semantic model suffered due to the language used in the highlights, the word distance model benefited. Particularly in the case of the Daily Mail dataset, highlights frequently have significant lexical overlap with passages in the accompanying article, which makes it easy for the word distance benchmark. For instance the query "*Tom Hanks is friends with* **X***'s manager, Scooter Brown*" has the phrase "*... turns out he is good friends with Scooter Brown, manager for Carly Rae Jepson*" in the context. The word distance benchmark correctly aligns these two while the frame-semantic approach fails to pickup the friendship or management relations when parsing the query. We expect that on other types of machine reading data where questions rather than Cloze queries are used this particular model would perform significantly worse.

Neural models Within the group of neural models explored here, the results paint a clear picture with the Impatient and the Attentive Readers outperforming all other models. This is consistent with our hypothesis that attention is a key ingredient for machine reading and question answering due to the need to propagate information over long distances. The

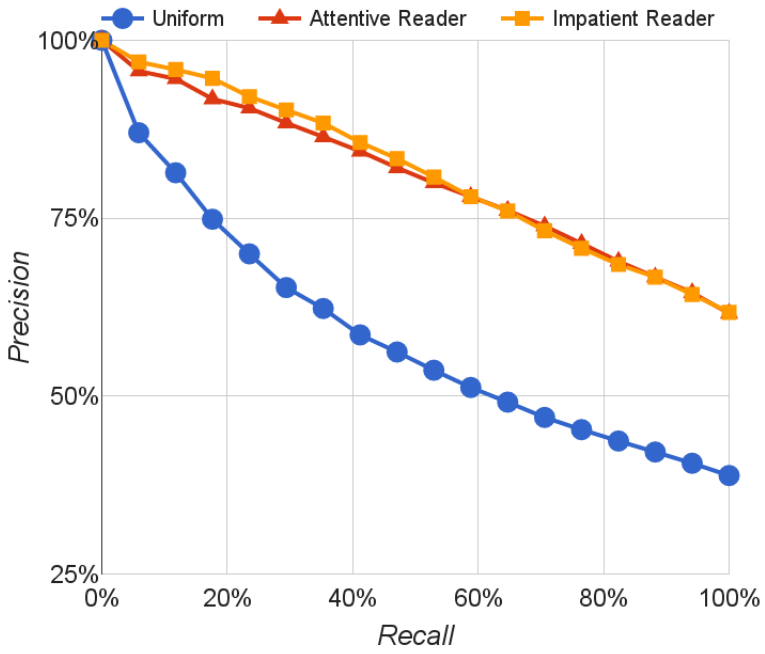


Figure 4.2: Precision@Recall for the attention models on the CNN validation data.

Deep LSTM Reader performs surprisingly well, once again demonstrating that this simple sequential architecture can do a reasonable job of learning to abstract long sequences, even when they are up to two thousand tokens in length. However this model does fail to match the performance of the attention based models, even though these only use single layer LSTMs.[5]

The poor results of the Uniform Reader support our hypothesis of the significance of the attention mechanism in the Attentive model's performance as the only difference between these models is that the attention variables are ignored in the Uniform Reader. The precision@recall statistics in Figure 4.2 again highlight the strength of the attentive approach.

4.4.1 Performance across document length

To understand how the model performance depends on the size of the context, we plot performance versus document lengths in Figures 4.3 and 4.4. The first figure (Fig. 4.3) plots a sliding window of performance across document length, showing that performance of the attentive models degrades slightly as documents increase in length. The second figure (Fig. 4.4) shows the cumulative performance with documents up to length N, showing that

[5]Memory constraints prevented us from experimenting with deeper Attentive Readers.

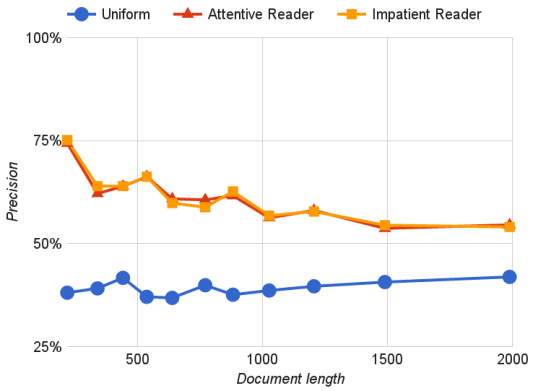


Figure 4.3: Precision@Document Length for the attention models on the CNN validation data. The chart shows the precision for each decile in document lengths across the corpus as well as the precision for the 5% longest articles.

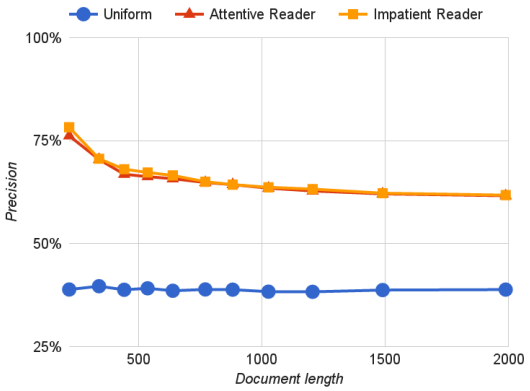


Figure 4.4: Aggregated precision for documents up to a certain lengths. The points mark the i^{th} decile in document lengths across the corpus.

while the length does impact the models' performance, that effect becomes negligible after reaching a length of ~500 tokens.

4.5 Attention Analysis

We can visualize the attention mechanism as a heatmap over a context document to gain further insight into the models' performance. The highlighted words show which tokens in the document were attended to by the model. In addition we must also take into account that the vectors at each token integrate long range contextual information via the bidirectional LSTM encoders. Figure 4.5 depicts heat maps for two queries that were correctly answered by the Attentive Reader.[6] In both cases confidently arriving at the correct answer requires the model to perform both significant lexical generalization, e.g. 'killed' $\rightarrow$ 'deceased', and co-reference or anaphora resolution, e.g. '*ent119* was killed' $\rightarrow$ 'he was identified.' However it is also clear that the model is able to integrate these signals with rough heuristic indicators such as the proximity of query words to the candidate answer.

Below, we investigate further examples for additional queries from the CNN validation dataset. We consider examples from the Attentive Reader as well as the Impatient Reader.

[6]Note that these examples were chosen as they were short, the average CNN validation document contained 763 tokens and 27 entities, thus most instances were significantly harder to answer than these examples.

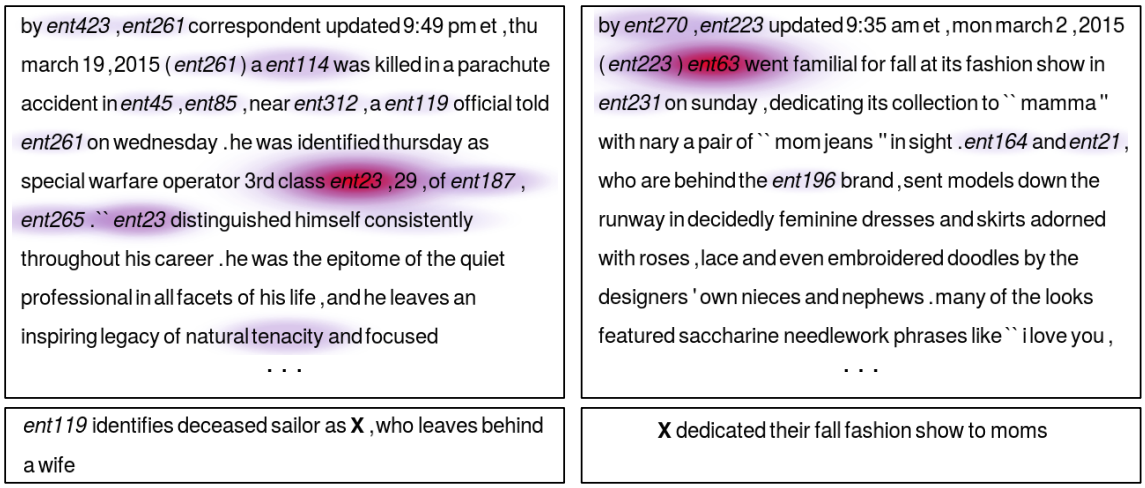


Figure 4.5: Attention heat maps from the Attentive Reader for two correctly answered validation set queries (the correct answers are *ent23* and *ent63*, respectively). Both examples require significant lexical generalization and co-reference resolution in order to be answered correctly by a given model.

4.5.1 Attentive Reader

Positive Instances Figure 4.6 shows two positive examples from the CNN validation set that require reasonable levels of lexical generalization and co-reference in order to be answered. The first query in Figure 4.7 contains strong lexical cues through the quote, but requires identifying the entity quoted, which is non-trivial in the context document. The final positive example (also in Figure 4.7) demonstrates the fearlessness of our model.

Negative Instances Figures 4.8 and 4.9 show examples of queries where the Attentive Reader fails to select the correct answer. The two examples in Figure 4.8 highlight a fairly common phenomenon in the data, namely ambiguous queries, where—at least following the anonymization process—multiple entities are plausible answers even when evaluated manually. Note that in both cases the query searches for an entity marker that describes a geographic location, preceded by the word "in". Here it is unclear whether the placeholder refers to a part of town, town, region or country.

Figure 4.9 contains two additional negative cases. The first failure is caused by the co-reference entity selection process. The correct entity, *ent15*, and the predicted one, *ent81*, both refer to the same person, but not being clustered together. Arguably this is a difficult clustering as one entity refers to "Kate Middleton" and the other to "The Duchess

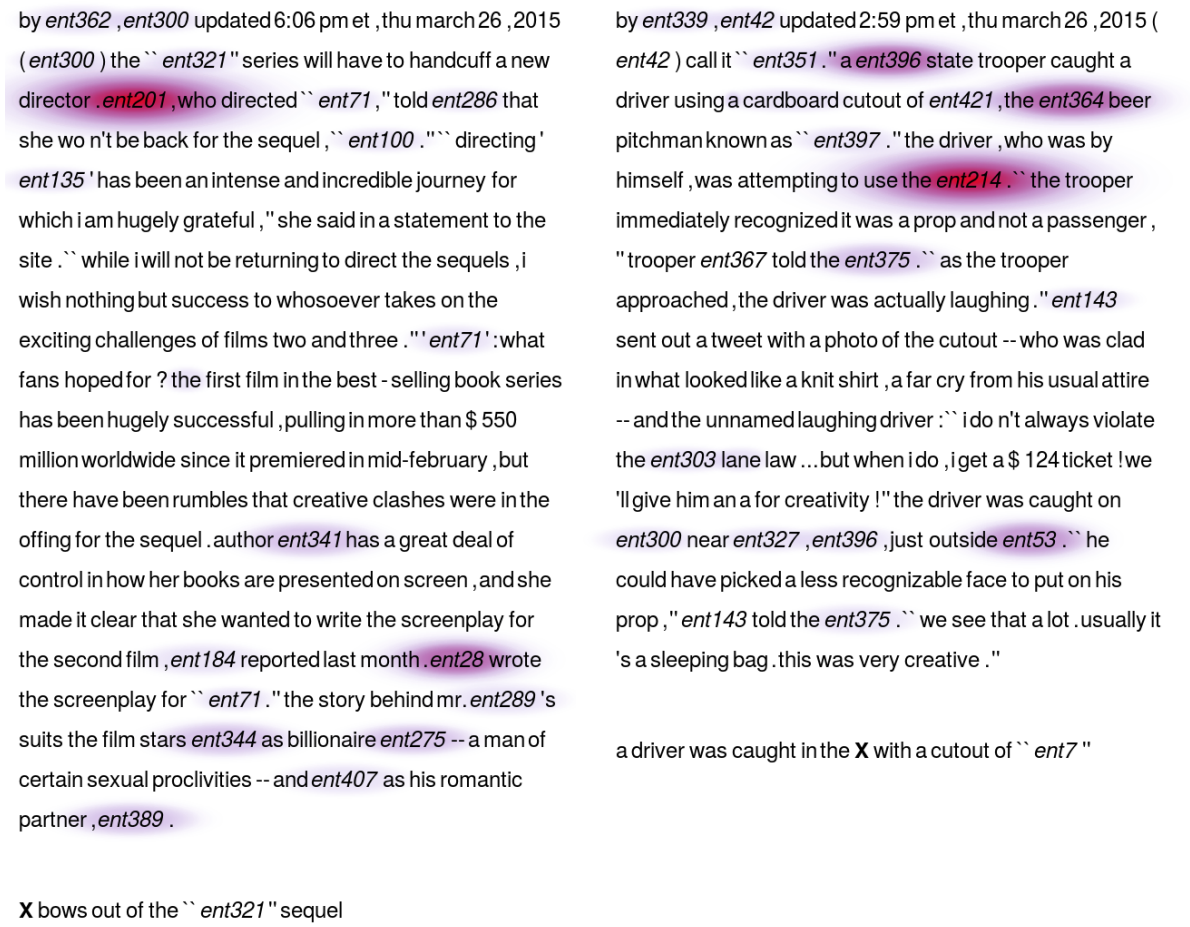


Figure 4.6: Attention heat maps from the Attentive Reader for two more correctly answered validation set queries. Both examples require significant lexical generalization and co-reference resolution to find the correct answers *ent201* and *ent214*, respectively.

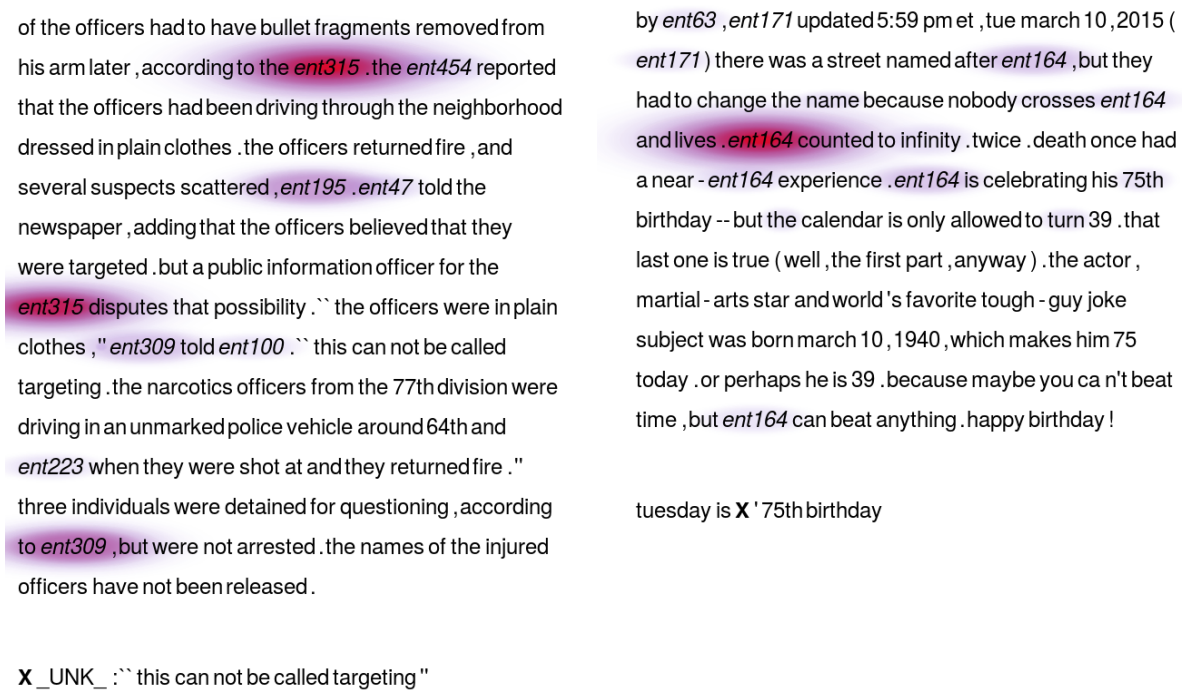


Figure 4.7: Two more correctly answered validation set queries. The left example (entity *ent315*) requires correctly attributing the quote, which does not appear trivial with a number of other candidate entities in the vicinity. The right hand side shows our model is not afraid of Chuck Norris (*ent164*).

of Cambridge". The right example shows a situation in which the model fails as it perhaps gets too little information from the short query and then selects the wrong cue with the term "claims" near the wrongly identified entity *ent1* (correct: *ent74*).

4.5.2 Impatient Reader

To give a better intuition for the behaviour of the Impatient Reader, we use a similar visualization technique as before. However, this time around we highlight the attention at every time step as the model updates its focus while moving through a given query. Figures 4.10–4.13 shows how the attention of the Impatient Reader changes and becomes increasingly more accurate as the model considers larger parts of the query. Note how the attention is distributed fairly arbitrary at first, slowly focusing on the correct entity *ent5* only once the question has sufficiently been parsed.

4.6 Summary

The supervised paradigm for training machine reading and comprehension models provides a promising avenue for making progress on the path to building full natural language understanding systems. We have demonstrated a methodology for obtaining a large number of document-query-answer triples and shown that recurrent and attention based neural networks provide an effective modelling framework for this task. Our analysis indicates that the Attentive and Impatient Readers are able to propagate and integrate semantic information over long distances. In particular we believe that the incorporation of an attention mechanism is the key contributor to these results.

In the next chapter, we present subsequent work using the here introduces datasets. Moreover, we review datasets for reading comprehension introduced after this work.

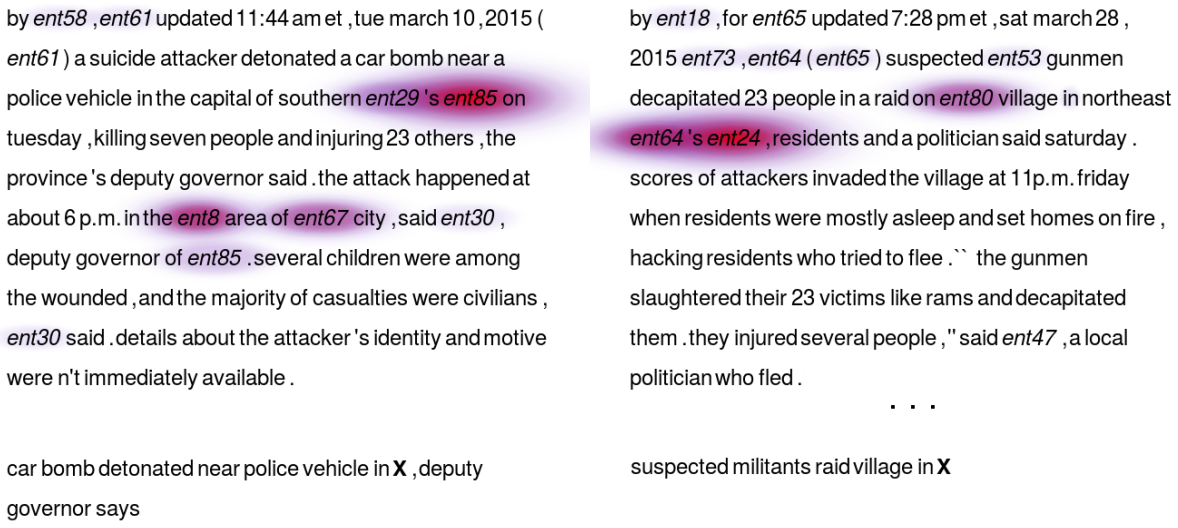


Figure 4.8: Attention heat maps from the Attentive Reader for two wrongly answered validation set queries. In the left case the model returns *ent85* (correct: *ent67*), in the right example it gives *ent24* (correct: *ent64*). In both cases the query is unanswerable due to its ambiguous nature and the model selects a plausible answer.

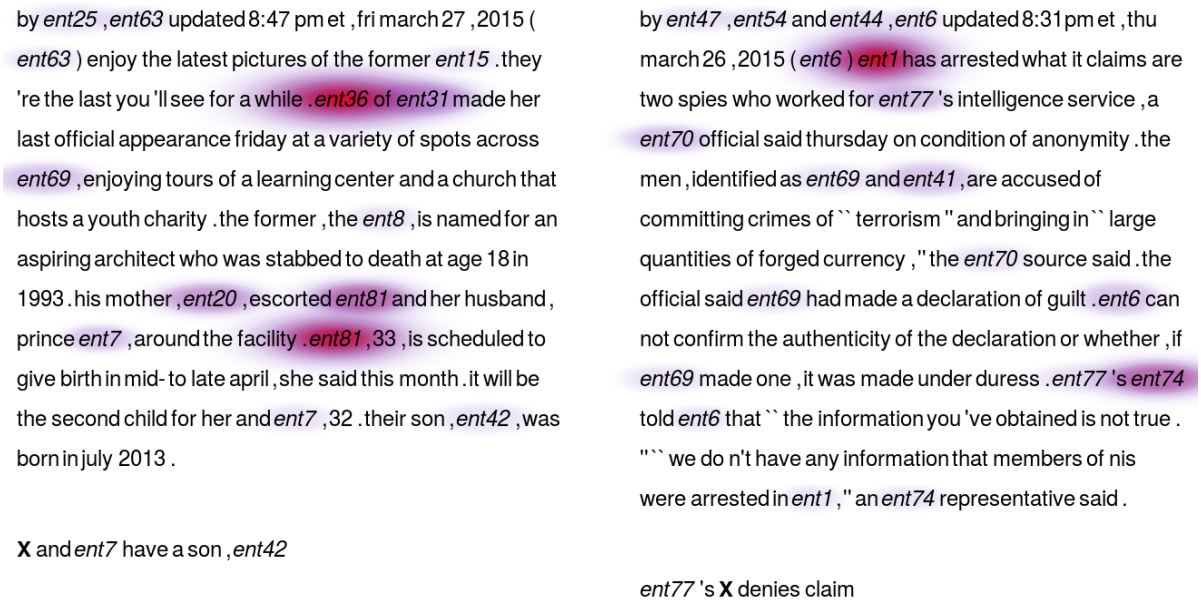


Figure 4.9: Additional heat maps for negative results. Here the left query selected *ent81* instead of *ent15* and the right query *ent1* instead of *ent74*.

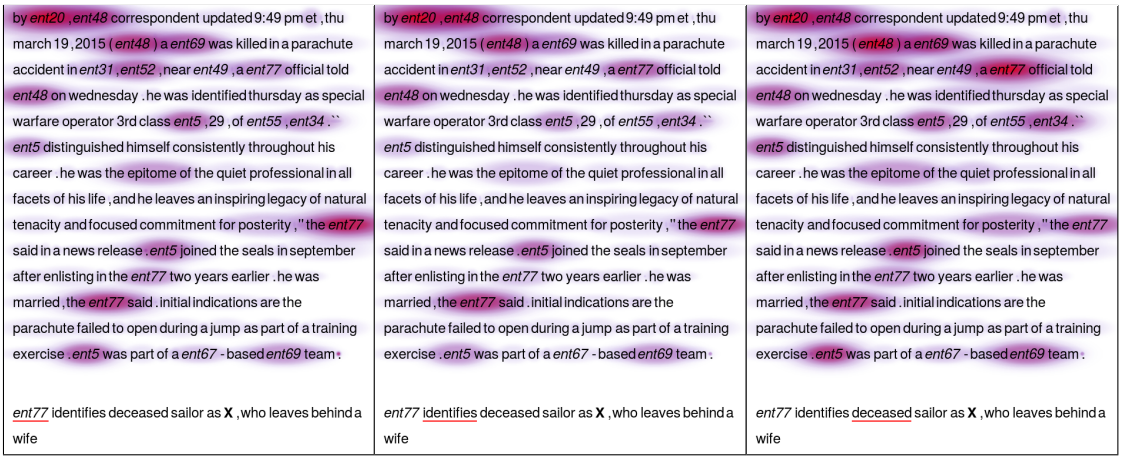


Figure 4.10: Attention of the Impatient Reader at time steps 1, 2 and 3.

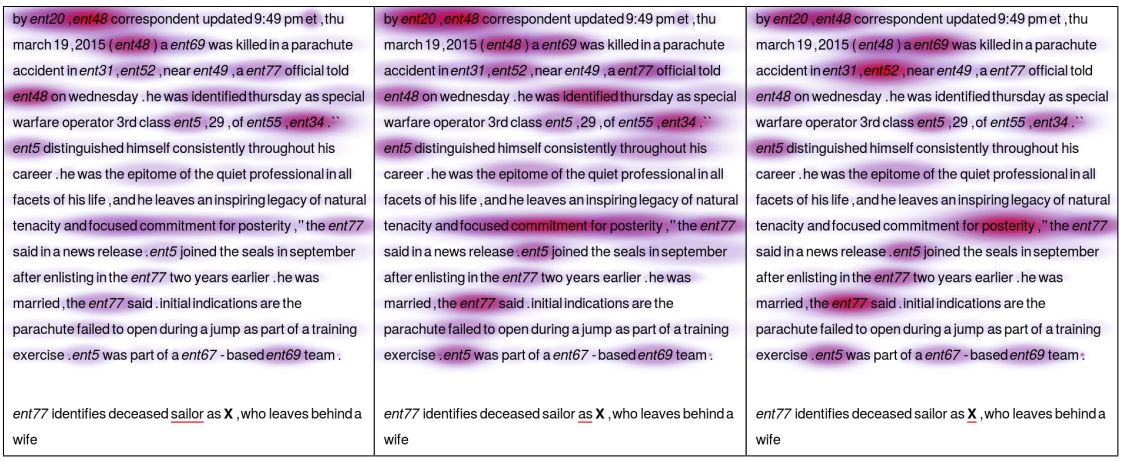


Figure 4.11: Attention of the Impatient Reader at time steps 4, 5 and 6.

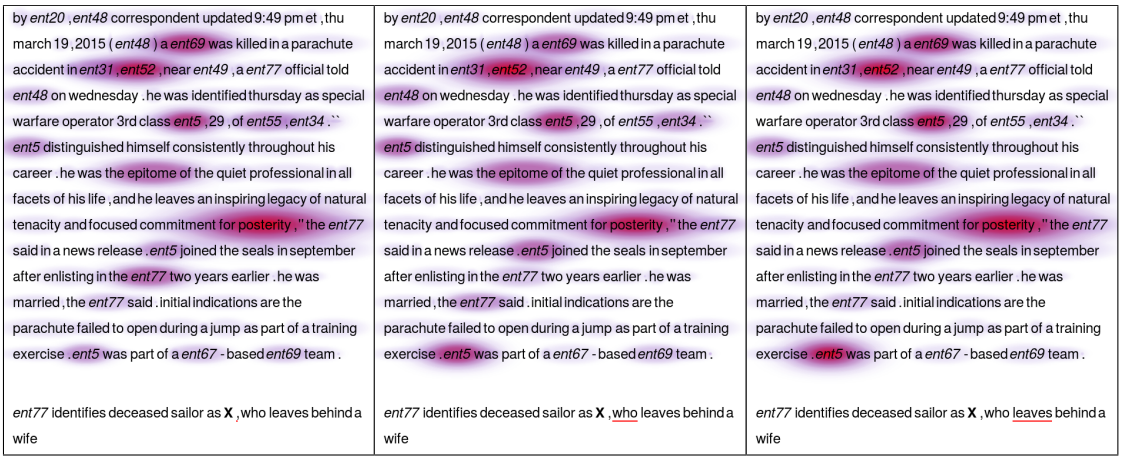


Figure 4.12: Attention of the Impatient Reader at time steps 7, 8 and 9.

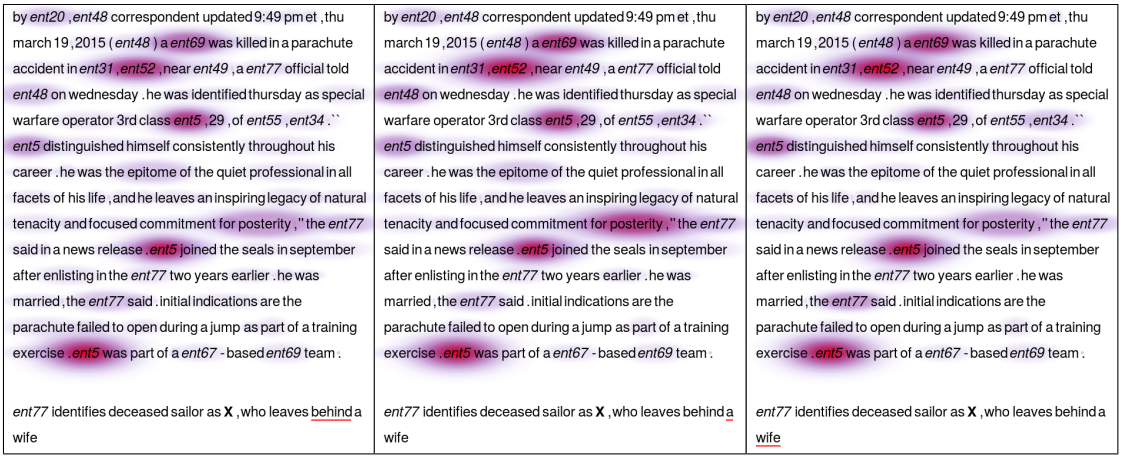


Figure 4.13: Attention of the Impatient Reader at time steps 10, 11 and 12.

Chapter 5

Rise of Reading Comprehension

Chapter Abstract

Our work from previous chapter has provided a scalable challenge that has supported NLP research in reading comprehension. In this chapter we will review the work on the CNN and Daily Mail datasets. Following that, we will review related tasks for machine reading comprehension that emerged later.

5.1 Models for CNN and Daily Mail Tasks

The new reading comprehension task introduced in the previous chapter for the first time allowed effectively training deep learning models for reading comprehension due to the scale of the dataset. In this section we review subsequent work and analyses of the two newly introduced datasets. Note that in this section we focus only on models with results reported on the two new datasets; we discuss subsequent tasks in the next section.

The best model at the time of writing is the Gated Attention Reader (Dhingra et al., 2016) which uses both pre-initialized unsupervised word embeddings and character embeddings to deal with unknown entities, employs a multi hop architecture with custom attention function over the question for each token of the document.

The subsequent work on these datasets focused on developing mainly two kinds of model architectures. Models with several forms of attention and varying simplicity perform quite well; the other direction, motivated by the hypothesis that multiple steps of reasoning are required, led to models with multiple hops of attentions.

	CNN		Daily Mail	
	valid	test	valid	test
Maximum frequency	30.5	33.2	25.6	25.5
Exclusive frequency	36.6	39.3	32.7	32.8
Frame-semantic model	36.3	40.2	35.5	35.5
Word distance model	50.5	50.9	56.4	55.5
Deep LSTM Reader	55.0	57.0	63.3	62.2
Uniform Reader	39.0	39.4	34.6	34.4
Attentive Reader	61.6	63.0	70.5	69.0
Impatient Reader	61.8	63.8	69.0	68.0
Entity-Centric Classifier (Chen et al., 2016a)	67.1	67.9	69.1	68.3
MemNNs (Hill et al., 2016)	63.4	66.8		
AS Reader (Kadlec et al., 2016)	68.6	69.5	75.0	73.9
DER Network (Kobayashi et al., 2016)	71.3	72.9		
Iterative Attentive Reader (Sordoni et al., 2016)	72.6	73.3		
Modified Attentive Reader (relabelling) (Chen et al., 2016b)	73.8	73.6	77.6	76.6
EpiReader (Trischler et al., 2016b)	73.4	74.0		
Attention over Attention Reader (Cui et al., 2016)	73.1	74.4		
ReasoNet (Shen et al., 2016)	72.9	74.7	77.7	76.6
BiDAF (Seo et al., 2017)	76.3	76.9	80.3	79.6
GA Reader (Dhingra et al., 2016)	77.9	77.9	81.5	80.9
Ensembles				
MemNNs (Hill et al., 2016)	66.2	69.4		
AS Reader (Kadlec et al., 2016)	73.9	75.4	78.7	77.7
Iterative Attentive Reader (Sordoni et al., 2016)	75.2	76.1		
Modified Attentive Reader (relabelling) (Chen et al., 2016b)	77.2	77.6	80.2	79.2

Table 5.1: Accuracy of the benchmarks, models (from Table 4.5), and subsequent work on the CNN and Daily Mail datasets.

The models developed considered many forms of scalable attention and multi-step reasoning neural network architectures obtaining 14.1 and 12.9 percentage point improvements in accuracy on the CNN and Daily Mail dataset we introduces here, respectively, compared to our results in Chapter 4. See Table 5.1 for comparison of the model accuracies.

Attention Models

Chen et al. (2016a) do a further examination of the CNN and Daily Mail datasets introduced here. On a study of 100 examples they find that 17% of examples are ambiguous or hard, 8% contain coreference errors, and 75% of examples have a various level of difficulty. Their neural network model does well and there still remains a large number, at least 5%, solvable and hard instances. The work introduces an *Entity-Centric Classifier* with hand-engineered features, and *Modified Attentive Reader*, which is a simplified version of the Attentive Reader introduced in Chapter 4 that uses a bilinear term for attention, uses weighted contextual embeddings directly for prediction, and predicts entities occurring only in the document instead of predicting a word from the entire vocabulary as we did above. A further bias towards entities occurring earlier in the article, introduced by relabelling them, provides further improvements on both tasks (Chen et al., 2016b).

Kadlec et al. (2016) introduce the *Attention Sum Reader (AS Reader)* which is a variation of a Pointer Network (Vinyals et al., 2015a) where the answer token is selected by pointing to the context document, i.e. predicting a distribution over the context-token positions. In the AS Reader, probabilities are aggregated across the token types. This model is simple, and yet surprisingly effective model for this task, which makes it an obvious benchmark for reading comprehension.

Attention over Attention Reader (Cui et al., 2016) uses a single step architecture by applying attention twice. This involves calculating a matching score between each pair of document and question tokens.

The *Bi-Directional Attention Flow (BiDAF)* model (Seo et al., 2017) explores using a more complex attention mechanism in a single hop architecture, similar to Attention over Attention Reader in that it also includes calculating scores for pairs of document and question tokens.

Multi-Hop Models

Hill et al. (2016) explore applying a variation of *Memory Networks (MemNNs)* (Weston et al., 2015b) to the CNN dataset. In this model the context document is embedded and then through attention over the memory the answer is predicted. The paper explores extensions to the original Memory Networks architecture tailored for this specific task. They hypothesize that the memory cells to be attended to are those that correspond to the answer candidates and in particular those that correspond to the answer entity. The best model uses self-supervision and embeds windows of words around entities only.

Iterative Attentive Reader (Sordoni et al., 2016) iteratively attends to both the questions and the document in turn for a tuned, constant number of steps. The Impatient Reader introduced earlier, in contrast, attended to document while reading the question, for each token of the question.

Gated Attention Reader (GA Reader) (Dhingra et al., 2016) is an instance of a multi-hop architecture for reading comprehension. At each hop a previous document-long embedding for the document is produced and attended with hop specific query embedding using a multiplicative, additive, or concatenation attention function.

ReasoNet (Shen et al., 2016) is another example of a multi hop architecture. In contrast with previous work, the number of hops is chosen by a stochastic binary variable and the model is trained using reinforcement learning.

Other Models

Trischler et al. (2016b) design a new model *EpiReader* which approaches the task in two steps. First, the Extractor part which works similarly to the AS Reader, and secondly, Reasoner component which re-ranks the answer candidates by measuring entailment.

Dynamic Entity Representation (DER) Network by Kobayashi et al. (2016) builds entity representations by accumulating information as it reads the document. The paper claims that the model can leverage information from different sentences to answer hard questions.

5.1.1 Summary

The main motivation for the work in Chapter 4 was to learn to understand language and test it by asking questions about a text. We introduced this task with Cloze-style questions, answers being entities which are marked in the text, and the metric being the accuracy of

selected word/entity. The natural ways to improve the accuracy, used in the above work, were to consider context of the marked entities only, that is, to treat those tokens in a special way; to predict only entities instead of a word from the full vocabulary; introduce a bias towards entities occurring earlier in the articles; or match the question tokens, instead of a question representation, with tokens of the context article. All these approaches are specific to this task of Cloze questions with entity answers and may not go, in all instances, towards our original goal of understanding language.

5.2 Recent Work and Datasets

In addition to the explicit purpose of the CNN and Daily Mail datasets for learning reading comprehension, the datasets were also used for summarization (Nallapati et al., 2016; See et al., 2017, e.g.).

Following the introduction of the CNN and Daily Mail datasets, several more datasets were developed for learning models that understand language through reading comprehension. We review some of the recent datasets in Table 5.2 and then further in Section 6.2. The list of datasets is likely not comprehensive and was constructed in mid-2017.

In the next chapter we will introduce a new, much more complex task, which will require future research to depart form using token-level matching attention models to infer answers and instead focus again more on language understanding.

Dataset	Documents	Questions	Answers
MCTest (Richardson et al., 2013a)	660 short stories, grade school level	2640 human generated, based on the document	multiple choice
CNN / Daily Mail (Section 4.2) (Hermann et al., 2015)	93K+220K news articles	387K+997K Cloze-form, based on highlights	entities
Children's Book Test (CBT) (Hill et al., 2016)	687K of 20 sentence passages from 108 children's books	Cloze-form, from the 21st sentence	multiple choice
BookTest (Bajgar et al., 2016)	14.2M, similar to CBT	Cloze-form, similar to CBT	multiple choice
SQuAD (Rajpurkar et al., 2016)	23K paragraphs from 536 Wikipedia articles	108K human generated, based on the paragraphs	spans
NewsQA (Trischler et al., 2016a)	13K news articles from the CNN dataset	120K human generated, based on headline and highlights	spans
MS MARCO (Nguyen et al., 2016)	1M passages from 200K+ documents retrieved using the queries	100K search queries	human generated, based on the passages
SearchQA (Dunn et al., 2017)	6.9m passages retrieved from a search engine using the queries	140k human generated Jeopardy! questions	human generated Jeopardy! answers
TriviaQA (Joshi et al., 2017)	650K web pages from search engine, and Wikipedia pages for entity mentions	96K crawled human generated trivia questions	crawled human generated 92.85% are Wikipedia titles
Quasar (Dhingra et al., 2017)	StackOverflow and ClueWeb09	37k Cloze and 43k trivia questions	entity or human generated
NarrativeQA (Section 6.3) (Kočiský et al., 2018)	1,572 stories (books, movie scripts) & human generated summaries	46,765 human generated, based on summaries	human generated, based on summaries

Table 5.2: Comparison of text-based datasets for question answering and reading comprehension.

Chapter 6

Complex Narrative Understanding

Chapter Abstract

Reading comprehension (RC)—in contrast to information retrieval—requires integrating information and reasoning about events, entities, and their relations across a full document. Question answering is conventionally used to assess RC ability, in both artificial agents and children learning to read. However, existing RC datasets and tasks are dominated by questions that can be solved by selecting answers using superficial information (e.g., local context similarity or global term frequency); they thus fail to test for the essential integrative aspect of RC. To encourage progress on deeper comprehension of language, we present a new dataset and set of tasks in which the reader must answer questions about stories by reading entire books or movie scripts. These tasks are designed so that successfully answering their questions requires understanding the underlying narrative rather than relying on shallow pattern matching or salience. We show that although humans solve the tasks easily, standard RC models struggle on the tasks presented here.

6.1 Introduction

Natural language understanding seeks to create models that read and comprehend text. A common strategy for assessing the language understanding capabilities of comprehension

The work presented in this chapter was originally presented in Kočiský et al. (2018).

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 6.1: Example question–answer pair. The snippets here were extracted by humans from summaries and the full text of movie scripts or books, respectively, and are *not* provided to the model as supervision or at test time. Instead, the model will need to read the full text and locate salient snippets based solely on the question and its reading of the document in order to generate the answer.

models is to demonstrate that they can answer questions about documents they read, akin to how reading comprehension is tested in children when they are learning to read. After reading a document, a reader usually can not reproduce the entire text from memory, but often can answer questions about underlying narrative elements of the document: the salient entities, events, places, and the relations between them. Thus, testing understanding requires the creation of questions that examine high-level abstractions instead of just facts occurring in one sentence at a time.

Unfortunately, superficial questions about a document may often be answered successfully (by both humans and machines) using a shallow pattern matching strategies or guessing based on global salience. In the following section, we survey existing QA datasets, showing that they are either too small or answerable by shallow heuristics (Section 6.2). On the other hand, questions which are not about the surface form of the text, but rather about the underlying narrative, 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 in-

tegrated information. That is, they test that the reader comprehends language, not just that it can pattern match. We present a new task and dataset, which we call NarrativeQA, which will test and reward artificial agents approaching this level of competence (Section 6.3).

The dataset consists of *stories*, which are books and movie scripts, with human written questions and answers based solely on human-generated abstractive *summaries*. For the RC tasks, questions may be answered using just the summaries or the full story text. We give a short example of a sample movie script from this dataset in Figure 6.1. Fictional stories have a number of advantages as a domain. First, they are largely self-contained: beyond the basic fundamental vocabulary of English and basic real-world knowledge and concepts, all the information about salient entities and new concepts required to understand the narrative is present in the document, with the expectation that a reasonably competent language user would be able to understand it.[1] Second, story summaries are abstractive and generally written by independent authors who know the work only as a reader. We make the dataset available online.[2]

6.2 Review of Reading Comprehension Data and Models

There are a large number of datasets and associated tasks available for the training and evaluation of reading comprehension models. We summarize the key features of a collection of popular recent datasets in Table 5.2. In this section, we briefly discuss the nature and limitations of these datasets and their associated tasks.

MCTest (Richardson et al., 2013a) is a collection of short stories, each with multiple questions. Each such question has set of possible answers, one of which is labelled as correct. While this could be used as a QA task, the MCTest corpus is in fact intended as an answer selection corpus. The data is human generated, and the answers can be phrases or sentences. The main limitation of this dataset is that it serves more as an evaluation challenge than as the basis for end-to-end training of models, due to its relatively small size.

[1] For example, new names and words may be coined by the author (e.g. "muggle" in Harry Potter novels) but the reader need only appeal to the book itself to understand the meaning of these concepts, and their place in the narrative. This ability to form new concepts based on the contexts of a text is a crucial aspect of reading comprehension, and is in part tested as part of the question answering tasks we present.

[2] `http://deepmind.com/publications`

In contrast, CNN/Daily Mail (Hermann et al., 2015), Children's Book Test (CBT) (Hill et al., 2016), and BookTest (Bajgar et al., 2016) each provide large amounts of question–answer pairs. Questions are Cloze-form (predict the missing word) and are produced from either short abstractive summaries (CNN/Daily Mail) or from the next sentence in the document the context was taken from (CBT and BookTest). The tasks associated with these datasets are all selecting an answer from a set of options, which is explicitly provided for CBT and BookTest, and is implicit for CNN/Daily Mail, as the answers are always entities from the document. This significantly favours models that operate by pointing to a particular token (or type). Indeed, the most successful models on these datasets, such as the Attention Sum Reader (AS Reader) (Kadlec et al., 2016), exploit precisely this bias in the data. However, these models are inappropriate for answers requiring synthesis of a new answer. This bias towards answers that are shallowly salient is a more serious limitation of the CNN/Daily Mail dataset, since its context documents are news stories which usually contain a small number of salient entities and focus on a single event.

SQuAD (Rajpurkar et al., 2016) and NewsQA (Trischler et al., 2016a) offer a different challenge. 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. While they provide a large number of questions, 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. While the answers are multi-word phrases, the spans are generally short and rarely cross sentence boundaries. Simple models scoring and/or extracting candidate spans conditioned on the question and superficial signal from the rest of the document do well (Seo et al., 2016, e.g.). These models will not trivially generalize to problems where the answers are not spans in the document, supervision for spans is not provided, or several discontinuous spans are needed to generate a correct answer. This restricts the scalability and applicability of models doing well on SQuAD or NewsQA to more complex problems.

MS MARCO (Nguyen et al., 2016) presents a bolder challenge: questions are paired with sets of snippets ("context passages") that contain the information necessary to answer the question, and answers are free-form human generated text. However, as no restriction

was placed on annotators to prevent them from copying answers from source documents, many answers are in fact verbatim copies of short spans from the context passages. Models which do well on SQuAD (e.g. Wang and Jiang (2016), Weissenborn et al. (2017)), extracting spans or pointing, do well here too, and the same concerns as above about the general applicability of solutions to this dataset to larger reading comprehension problems applies.

SearchQA (Dunn et al., 2017) is a recent dataset in which the context for each question is a set of documents retrieved by a search engine using the question as the query. However, in contrast with previous datasets neither questions nor answers were produced by annotating the context documents, but rather the context documents were retrieved after collecting pre-existing question–answer pairs. As such, it is not open to the same annotation bias as the datasets discussed above. However, upon examining answers in the Jeopardy data used to construct this dataset, one finds that 80% of answers are bigrams or unigrams, and 99% are 5 tokens or fewer. Of a sample of 100 answers, 72% are named entities, all are short noun-phrases, such that we expect models similar to pointer networks (Vinyals et al., 2015a), augmented to iteratively point in order to match n-grams, would get traction on this task, as the authors of the SearchQA paper themselves show.

Summary of Limitations. We see several limitations of the scope and depth of the RC problems in existing datasets. First, several datasets are small (MCTest) or not overly naturalistic (bAbI; Weston et al. (2015a)). Second, in more naturalistic documents, a majority of questions require only a single sentence to locate supporting information for answering (Chen et al., 2016a; Rajpurkar et al., 2016). This, we suspect, is largely an artefact of the question generation methodology, in which annotators have created questions from a context document, or where context documents that explicitly answer a question are identified using a search engine. Although the factoid-like Jeopardy questions of SearchQA also appears to favour questions answerable with local context. Finally, we see further evidence of the superficiality of the questions in the architectures that have evolved to solve them, which tend to exploit span selection based on representations derived from local context and the query (Seo et al., 2016; Wang et al., 2017).

6.3 NarrativeQA: A New Dataset

In this section, we introduce our new dataset, NarrativeQA, which addresses many of the limitations identified in existing datasets.

6.3.1 Desiderata

From the above discussed features and limitations, we define our desiderata as follows. We wish to construct a dataset with a large number of question–answer pairs based on either a large number of supporting documents or from a smaller collection of large documents. This permits the training of neural network-based models over word embeddings and provide decent lexical coverage and diversity. The questions and answers should be natural, unconstrained, and human generated, and answering questions should frequently require reference to several parts or a larger span of the context document rather than superficial representations of local context. Furthermore, we want annotators to privilege writing answers expressed in their own words, and consider higher-level relations between entities, places, and events, rather than copy short spans of the document.

Furthermore, we want to evaluate models both on the fluency and correctness of generated free-form answers, and as an answer selection problem, which requires the provision of sensible distractors to the correct answer. Finally, the scope and complexity of the QA problem should be such that current models struggle, while humans are capable of solving the task correctly, so as to motivate further research into the development of models seeking human reading comprehension ability.

6.3.2 Data Collection Method

We will consider complex, self-contained narratives as our documents/stories. To make the annotation tractable and lead annotators towards asking non-localized questions, we will only provide them human written summaries of the stories for generating the question–answer pairs.

We present both books and movie scripts as stories in our dataset. Books were collected from Project Gutenberg[3] and movie scripts are scraped from the web.[4] We matched

[3] `http://www.gutenberg.org/`

[4] Mainly from `http://www.imsdb.com/`, but also `http://www.dailyscript.com/`, `http://www.awesomefilm.com/`.

our stories with plot summaries from Wikipedia using titles and verified the matching with help from human annotators. The annotators were asked to determine if both the story and the summary refer to a movie or a book (as some books are made into movies), or if they are the same part in a series produced in the same year. In this way we obtained 1,567 stories. This provides with a smaller set of documents, compared to the other datasets, but the documents are long which provides us with good lexical coverage and diversity. The bottleneck for obtaining a larger number of publicly available stories was finding corresponding summaries.

Annotators on Amazon Mechanical Turk were instructed to write 10 question–answer pairs each based solely on a given summary[5]. Reading and annotating summaries is tractable unlike writing questions and answers based on the full stories, and moreover, as the annotators never see the full stories we are much less likely to get questions and answers which are extracted from a localized context.

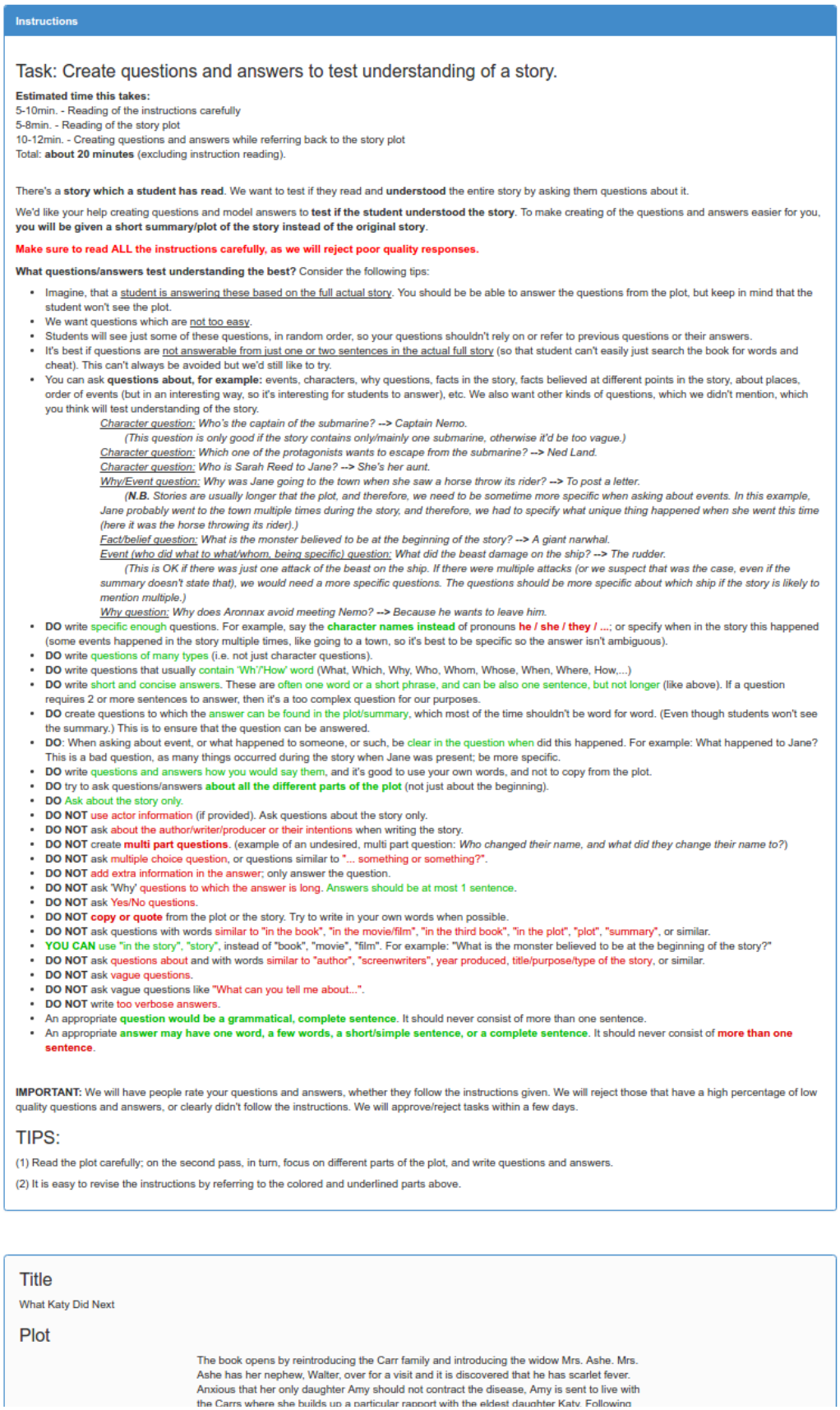
Annotators were instructed to imagine that they are writing questions to test students who have read the full stories but not the summaries. We required questions that are specific enough, given the length and complexity of the narratives, and to provide a diverse set of questions about characters, events, why this happened, and so on. Annotators were encouraged to use their own words and we prevented them from copying.[6] We asked for answers that are grammatical, complete sentences, and explicitly allowed short answers (one word, or a few-word phrase, or a short sentence) as we think that answering with a full sentence is frequently perceived as artificial when asking about factual information. Annotators were asked to avoid extra, unnecessary information in the question or the answer, and to avoid yes/no questions or questions about the author or the actors.

About 30 question–answer pairs per summary were obtained. The result is a collection of human written natural questions and answers. As we have multiple questions per summary/story, this allows us to consider answer selection (from among the 30) as a simpler version of the QA rather than answer generation from scratch. Answer selection (Hewlett et al., 2016) and multiple-choice question answering (Richardson et al., 2013a; Hill et al., 2016) are frequently used.

We additionally collected a second reference answer for each question by asking annotators to judge whether a question is answerable, given the summary, and provide an answer

[5]We payed $1.00 for this task.

[6]This was done both through instructions and JavaScript hard limitations on the annotation site.



Instructions

Task: Create questions and answers to test understanding of a story.

Estimated time this takes:
5-10min. - Reading of the instructions carefully
5-8min. - Reading of the story plot
10-12min. - Creating questions and answers while referring back to the story plot
Total: **about 20 minutes** (excluding instruction reading).

There's a **story which a student has read**. We want to test if they read and **understood** the entire story by asking them questions about it.

We'd like your help creating questions and model answers to **test if the student understood the story**. To make creating of the questions and answers easier for you, **you will be given a short summary/plot of the story instead of the original story**.

Make sure to read ALL the instructions carefully, as we will reject poor quality responses.

What questions/answers test understanding the best? Consider the following tips:

- Imagine, that a student is answering these based on the full actual story. You should be be able to answer the questions from the plot, but keep in mind that the student won't see the plot.
- We want questions which are not too easy.
- Students will see just some of these questions, in random order, so your questions shouldn't rely on or refer to previous questions or their answers.
- It's best if questions are not answerable from just one or two sentences in the actual full story (so that student can't easily just search the book for words and cheat). This can't always be avoided but we'd still like to try.
- You can ask **questions about, for example:** events, characters, why questions, facts in the story, facts believed at different points in the story, about places, order of events (but in an interesting way, so it's interesting for students to answer), etc. We also want other kinds of questions, which we didn't mention, which you think will test understanding of the story.
  - *Character question: Who's the captain of the submarine? --> Captain Nemo.*
    - *(This question is only good if the story contains only/mainly one submarine, otherwise it'd be too vague.)*
  - *Character question: Which one of the protagonists wants to escape from the submarine? --> Ned Land.*
  - *Character question: Who is Sarah Reed to Jane? --> She's her aunt.*
  - *Why/Event question: Why was Jane going to the town when she saw a horse throw its rider? --> To post a letter.*
    - *(**N.B.** Stories are usually longer that the plot, and therefore, we need to be sometime more specific when asking about events. In this example, Jane probably went to the town multiple times during the story, and therefore, we had to specify what unique thing happened when she went this time (here it was the horse throwing its rider).)*
  - *Fact/belief question: What is the monster believed to be at the beginning of the story? --> A giant narwhal.*
  - *Event (who did what to what/whom, being specific) question: What did the beast damage on the ship? --> The rudder.*
    - *(This is OK if there was just one attack of the beast on the ship. If there were multiple attacks (or we suspect that was the case, even if the summary doesn't state that), we would need a more specific questions. The questions should be more specific about which ship if the story is likely to mention multiple.)*
  - *Why question: Why does Aronnax avoid meeting Nemo? --> Because he wants to leave him.*
- **DO** write specific enough questions. For example, say the **character names instead** of pronouns **he / she / they / ...**; or specify when in the story this happened (some events happened in the story multiple times, like going to a town, so it's best to be specific so the answer isn't ambiguous).
- **DO** write questions of many types (i.e. not just character questions).
- **DO** write questions that usually contain 'Wh'/'How' word (What, Which, Why, Who, Whom, Whose, When, Where, How,...)
- **DO** write short and concise answers. These are often one word or a short phrase, and can be also one sentence, but not longer (like above). If a question requires 2 or more sentences to answer, then it's a too complex question for our purposes.
- **DO** create questions to which the answer can be found in the plot/summary, which most of the time shouldn't be word for word. (Even though students won't see the summary.) This is to ensure that the question can be answered.
- **DO**: When asking about event, or what happened to someone, or such, be clear in the question when did this happened. For example: What happened to Jane? This is a bad question, as many things occurred during the story when Jane was present; be more specific.
- **DO** write questions and answers how you would say them, and it's good to use your own words, and not to copy from the plot.
- **DO** try to ask questions/answers **about all the different parts of the plot** (not just about the beginning).
- **DO** Ask about the story only.
- **DO NOT** use actor information (if provided). Ask questions about the story only.
- **DO NOT** ask about the author/writer/producer or their intentions when writing the story.
- **DO NOT** create **multi part questions**. (example of an undesired, multi part question: *Who changed their name, and what did they change their name to?*)
- **DO NOT** ask multiple choice question, or questions similar to "... something or something?".
- **DO NOT** add extra information in the answer; only answer the question.
- **DO NOT** ask 'Why' questions to which the answer is long. Answers should be at most 1 sentence.
- **DO NOT** ask Yes/No questions.
- **DO NOT copy or quote** from the plot or the story. Try to write in your own words when possible.
- **DO NOT** ask questions with words similar to "in the book", "in the movie/film", "in the third book", "in the plot", "plot", "summary", or similar.
- **YOU CAN** use "in the story", "story", instead of "book", "movie", "film". For example: "What is the monster believed to be at the beginning of the story?"
- **DO NOT** ask questions about and with words similar to "author", "screenwriters", year produced, title/purpose/type of the story, or similar.
- **DO NOT** ask vague questions.
- **DO NOT** ask vague questions like "What can you tell me about...".
- **DO NOT** write too verbose answers.
- An appropriate **question would be a grammatical, complete sentence**. It should never consist of more than one sentence.
- An appropriate **answer may have one word, a few words, a short/simple sentence, or a complete sentence**. It should never consist of **more than one sentence**.

IMPORTANT: We will have people rate your questions and answers, whether they follow the instructions given. We will reject those that have a high percentage of low quality questions and answers, or clearly didn't follow the instructions. We will approve/reject tasks within a few days.

TIPS:

(1) Read the plot carefully; on the second pass, in turn, focus on different parts of the plot, and write questions and answers.

(2) It is easy to revise the instructions by referring to the colored and underlined parts above.

Title

What Katy Did Next

Plot

The book opens by reintroducing the Carr family and introducing the widow Mrs. Ashe. Mrs. Ashe has her nephew, Walter, over for a visit and it is discovered that he has scarlet fever. Anxious that her only daughter Amy should not contract the disease, Amy is sent to live with the Carrs where she builds up a particular rapport with the eldest daughter Katy. Following

Figure 6.2: Instructions for annotators writing question-answer pairs based on a summary.

	train	valid	test
# documents	1,102	115	355
... books	548	58	177
... movie scripts	554	57	178
# question–answer pairs	32,747	3,461	10,557
Avg. #tok. in summaries	659	638	654
Max #tok. in summaries	1,161	1,189	1,148
Avg. #tok. in stories	62,528	62,743	57,780
Max #tok. in stories	430,061	418,265	404,641
Avg. #tok. in questions	9.83	9.69	9.85
Avg. #tok. in answers	4.73	4.60	4.72

Table 6.1: NarrativeQA dataset statistics.

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%

Table 6.2: Frequency of first token of the question in the training set.

if it was. All but 2.3% of the questions were judged as answerable.

6.3.3 Core Statistics

We collected 1,567 stories, evenly split between books and movie scripts. We partitioned the dataset into non-overlapping training, validation, and test portions, along stories/summaries. See Table 6.1 for detailed statistics.

The dataset contains 46,765 question–answer pairs. The questions are grammatical questions written by human annotators, average 9.8 tokens in length, and are mostly formed as 'WH'-questions (see Table 6.2). We categorized a sample of 300 questions in Table 6.3. We observe a good variety of question types. An interesting category are questions which ask for something related to or occurring together/before/after with an event, of which there are about 15%.

Category	Frequency	Examples Question	Answer
Person	30.54%	Who revealed the entire con to Price?	Lily
Description	24.50%	What does Podzdnyshev do to hide his jealousy?	Leaves on a trip
		What is Frank Saltram's chief talent?	Witty conversation
		What was Clifford's previous job?	He was a cop
Location	9.73%	Where does Virgil die?	In the hotel
Why/reason	9.40%	Why doesn't Pozdnyshev run after the violinist?	Because he is wearing socks
How/method	8.05%	How did Jake survive being shot?	He used fake bullets and blood packets
Event	4.36%	When does Reiko realize the curse is still unbroken?	After her husband calls her
Entity	4.03%	What creature from 1935 still lives in 1999 along with Paul?	Mr. Jingles, Del's mouse
Object	3.36%	What does Pozdnyshev kill his wife with?	A dagger
Numeric	3.02%	How many times was Sharon shot in the chest?	Twice
Duration	1.68%	How long is Jason in cryogenic stasis?	445 years
Relation	1.34%	What is Nora's relationship to Michael?	Nora is Michael's sister

Table 6.3: Question categories on a sample of 300 questions from the validation set.

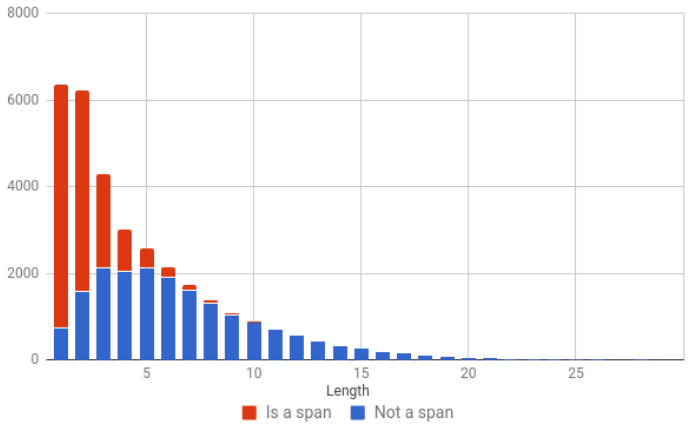


Figure 6.3: Answer length histogram on the training set with proportion of answers that are spans in the summary. There are 44.05% answers that appear as spans of the summaries.

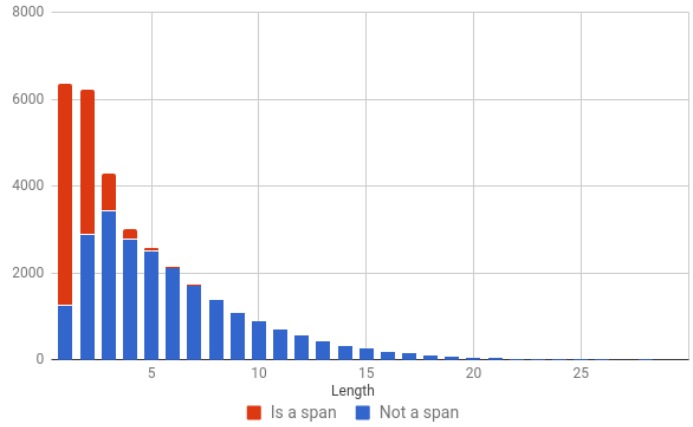


Figure 6.4: Answer length histogram on the training set with proportion of answers that are spans in the story. There are 29.57% answers that appear as spans of the stories.

Answers in the dataset are human written, short, averaging 4.73 tokens, but not restricted to spans from the documents. In Figures 6.3 and 6.4 we present the histogram of answer lengths and proportion which are spans; as expected, lower proportion of answers are spans on stories compared to summaries on which they were constructed. There are 44.05% and 29.57% answers that appear as spans of the summaries and the stories, respectively; as expected, lower proportion of answers are spans on stories compared to summaries on which they were constructed.

6.3.4 Tasks

We present scope and complexity: we consider either the summary or the story as context, and for each we evaluate answer generation and answer selection.

The task of answering questions based on summaries is similar in scope to previous datasets. However, summaries contain more complex relationships and timelines than news articles or short paragraphs from the web and thus provide a task different in nature. We hope that NarrativeQA will motivate the design of architectures capable of modelling such relationships. This setting is similar to the previous tasks in that the questions and answers were constructed based on these supporting documents.

The full version of NarrativeQA requires reading and understanding entire stories (i.e., books and movie scripts). At present, this task is intractable for existing neural models out of the box. We further discuss the challenges and possible approaches in the following sections.

We require the use of metrics for generated text. We evaluate using BLEU-1, BLEU-4 (Papineni et al., 2002), METEOR (Denkowski and Lavie, 2011), and ROUGE-L (Lin, 2004), using two references for each question,[7] except for the human baseline where we evaluate one reference against the other. We also evaluate our models using a ranking metric. This allows us to evaluate how good our model is at reading comprehension regardless of how good it is at generating answers. We rank answers for questions associated with the same summary/story and compute the mean reciprocal rank (MRR).[8]

6.4 Baselines and Oracles

In this section, we show that NarrativeQA presents a challenging problem for current approaches to reading comprehension by evaluating several baselines based on information retrieval (IR) techniques and neural models. Since neural models use quite different processes for generating answers (e.g., predicting a single word or entity, selecting a span of the document context, or open generation of the answer sequence), we present results on each. We also report the human performance by scoring the second reference answer against the first.

6.4.1 Simple IR Baselines

We consider basic IR baselines which retrieve an answer by selecting a span of tokens from the context document based on a similarity measure between the candidate span and a query. We compare two queries: the question and (as an oracle) the gold standard answer. The answer oracle provides an upper bound on the performance of span retrieval models, including the neural models discussed below. When using the question as the query, we obtain generalization results of IR methods. Test set results are computed by extracting either 4-gram, 8-gram, or full-sentence spans according to the best performance on the validation set.[9]

[7] We lowercase both the candidates and the references and remove the end of sentence marker and the final full stop.

[8] MRR is the mean over examples of $1/r$, where $r \in \{1, 2, \ldots\}$ is the rank of the correct answer among candidates.

[9] Note that we do not consider the span's context when computing the MRR for IR baselines, as the candidate spans (i.e. all answers to questions on the story) are given and simply ranked by their similarity to the query.

We consider three similarity metrics for extracting spans: BLEU-1, ROUGE-L, and the cosine similarity between bag-of-words embedding of the query and the candidate span using pre-trained GloVe word embeddings (Pennington et al., 2014).

6.4.2 Neural Benchmarks

As a first benchmark we consider a simple bi-directional LSTM sequence to sequence (Seq2Seq) model (Sutskever et al., 2014a) predicting the answer directly from the query. Importantly, we provide no context information from either summary or story. Such a model might classify the question and predict an answer of a similar topic or category.

Previous reading comprehension tasks such as CNN/Daily Mail motivated models constrained to predicting a single token from the input sequence. The AS Reader (Kadlec et al., 2016) considers the entire context and predicts a distribution over unique word types. We adapt the model for sequence prediction by using an LSTM sequence decoder and choosing a token from the input at each step of the output sequence.

As a span-prediction model we consider a simplified version of the Bi-Directional Attention Flow network (Seo et al., 2016). We omit the character embedding layer and learn a mapping from words to a vector space rather than making use of pre-trained embeddings; and we use a single layer bi-directional LSTM to model interactions among context words conditioned on the query (modelling layer). As proposed, we adopt the output-layer tailored for span-prediction and leave the rest unchanged. It was not our aim to use the state-of-the-art model for other datasets but rather to provide a strong benchmark.

Span prediction models can be trained by obtaining supervision on the training set from the oracle IR model. 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.

6.4.3 Neural Benchmarks on Stories

The design of the NarrativeQA dataset makes the straight-forward application of the existing neural architectures computationally infeasible, as this would require running an recurrent neural network on sequences of hundreds of thousands of time steps or computing a distribution over the entire input for attention, as is common.

We split the task into two steps: first, we retrieve a small number of relevant passages from the story using an IR system, and subsequently, apply one of the neural models above on the resulting document. The question becomes the query for retrieval. This IR problem is much harder that traditional document retrieval, as the documents, the passages here, are very similar, and the question is short and entities mentioned likely occur many times in the story.

Our retrieval system considers chunks of 200 words (irrespective of sentence boundaries) from the story and computes representations for all chunks and the query. We then select a varying number of such chunks based on their similarity to the query. For the Span Prediction model we tuned the number of chunks on the validation set. We experiment with different representations and similarity measures in Section 6.5. Finally, we concatenate the selected chunks in the correct temporal order and insert delimiters between them to obtain a much shorter document. For span prediction models, we then further select a span from the retrieved chunks as described in Section 6.4.2.

6.5 Experiments

In this section, we describe the data preparation methodology we used, and experimental results on the summary-reading task as well as the full story task.

6.5.1 Data Preparation

The provided narratives contain a large number of named entities (such as names of characters or places). Inspired by Hermann et al. (2015), we replace such entities with markers, such as `@entity42`. 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. Entities are replaced according to a simple heuristic based on a capital first character and the respective word not appearing in lowercase.

Model	Validation / Test				
	BLEU-1	BLEU-4	METEOR	ROUGE-L	MRR
IR Baselines					
BLEU-1 given question (1 sentence)	10.48/10.75	3.02/ 3.34	11.93/12.33	14.34/14.90	0.176/0.171
ROUGE-L given question (8-gram)	11.74/11.01	2.18/ 1.99	7.05/ 6.50	12.58/11.74	0.168/0.161
Cosine given question (1 sentence)	7.49/ 7.51	1.88/ 1.97	10.18/10.35	12.01/12.28	0.170/0.171
Random rank					0.133/0.133
Neural Benchmarks					
Seq2Seq (no context)	16.10/15.89	1.40/ 1.26	4.22/ 4.08	13.29/13.15	0.211/0.202
Attention Sum Reader	23.54/23.20	5.90/ 6.39	8.02/ 7.77	23.28/22.26	**0.269/0.259**
Span Prediction	**33.45/33.72**	**15.69/15.53**	**15.68/15.38**	**36.74/36.30**	—
Oracle IR Models					
BLEU-1 given answer (ans. length)	54.60/55.55	26.71/27.78	31.32/32.08	58.90/59.77	1.000/1.000
ROUGE-L given answer (ans. length)	52.94/54.14	27.18/28.18	30.81/31.50	59.09/59.92	1.000/1.000
Cosine given answer (ans. length)	46.69/47.95	24.25/25.25	27.02/27.81	44.64/45.66	0.836/0.838
Human agreement given summaries	44.24/44.43	18.17/19.65	23.87/24.14	57.17/57.02	—

Table 6.4: Experiments on summaries. Higher is better for all metrics. Sections 6.4.1 and 6.4.2 explain the IR and neural models, respectively.

6.5.2 Reading Summaries Only

Reading comprehension of summaries is similar to a number of previous reading comprehension tasks where questions were constructed based on the context document. However, plot summaries tend to contain more intricate event time lines and a larger number of characters, and in this sense, are more complex to follow than news articles or paragraphs from Wikipedia. See Table 6.4 for the results.

Given that questions were constructed based on the summaries, we expected that both neural models and span-selection models would perform well. This is indeed the case, with the neural span prediction model significantly outperforming all other proposed methods. However, there remains a significant room for improvement when compared with the oracle and human scores. Note that oracle scores can sometime be better than human agreement due to the scoring metric.

Both the plain sequence to sequence model and the AS Reader, successfully applied to the CNN/DailyMail reading comprehension task, also perform well on this task. We observe that the AS Reader tends to copy subsequent tokens from the context, thus behaving like a span prediction model. An additional inductive bias results in higher performance for the span prediction model. Similar observations between AS Reader and span models have also been made by Wang and Jiang (2016).

Note that we have tuned each model separately on the development set twice, once

Model	Validation / Test				
	BLEU-1	BLEU-4	METEOR	ROUGE-L	MRR
IR Baselines					
BLEU-1 given question (8-gram)	6.73/ 6.52	0.30/ 0.34	3.58/ 3.35	6.73/ 6.45	0.176/0.171
ROUGE-L given question (1 sentence)	5.78/ 5.69	0.25/ 0.32	3.71/ 3.64	6.36/ 6.26	0.168/0.161
Cosine given question (8-gram)	6.40/ 6.33	0.28/ 0.29	3.54/ 3.28	6.50/ 6.43	0.171/0.171
Random rank					0.133/0.133
Neural Benchmarks					
Attention Sum Reader given 1 chunk	16.95/16.08	1.26/1.08	3.84/3.56	12.12/11.94	0.164/0.161
Attention Sum Reader given 2 chunks	18.54/17.76	0.0/1.1	4.2/4.01	13.5/12.83	0.169/0.169
Attention Sum Reader given 5 chunks	18.91/18.36	1.37/1.64	4.48/4.24	14.47/13.4	0.171/0.173
Attention Sum Reader given 10 chunks	**20.0/19.09**	2.23/1.81	4.45/4.29	**14.47/14.03**	0.182/0.177
Attention Sum Reader given 20 chunks	19.79/19.06	**1.79/2.11**	**4.6/4.37**	14.86/14.02	0.182/0.179
Span Prediction	5.82/5.68	0.22/0.25	3.84/3.72	6.33/6.22	—
Oracle IR Models					
BLEU-1 given answer (ans. length)	41.81/42.37	7.03/ 7.70	19.10/19.52	46.40/47.15	1.000/1.000
ROUGE-L given answer (ans. length)	39.17/39.50	7.81/ 8.46	18.13/18.55	48.91/49.94	1.000/1.000
Cosine given answer (4-gram)	38.21/38.92	7.78/ 8.43	12.58/12.60	31.24/31.70	0.842/0.845
Human agreement given summaries	44.24/44.43	18.17/19.65	23.87/24.14	57.17/57.02	—

Table 6.5: Experiments on full stories. Each chunk contains 200 tokens. Higher is better for all metrics. Sections 6.4.1 and 6.4.2 explain the IR and neural models, respectively.

selecting the best model based on ROUGE-L and report the first four metrics, and a second time selecting based on MRR. We tuned the learning rate, hidden sizes, dropout.

6.5.3 Reading Full Stories Only

Table 6.5 summarizes the results on the full NarrativeQA task, where the context documents are full stories. As expected (and desired), we observe a decline in performance of the span-selection oracle IR model, compared with the results on summaries. This is unsurprising as the questions were constructed on summaries and confirms the initial motivation for designing this task. As previously, we considered all spans of a given length across the entire story for this model. For short answers of one or two words—typically main characters in a story—the candidate, i.e. the closest span to the reference answer, is easily found due to being mentioned throughout the text. For longer answers it becomes much less likely, compared to the summaries, that a high-scoring span can be found in the story. Note that this distinguishes NarrativeQA from many of the reviewed datasets.

In our IR plus neural two-step approach to the task, we first retrieve relevant chunks of the stories and then apply existing reading comprehension models. We use the questions to guide the IR system for chunk extraction, with the results of the standalone IR baselines giving an indication of the difficulty of this aspect of the task. The retrieval quality has

a direct effect on the performance of all neural models—a challenge which models on summaries are not presented with. We considered several approaches to chunk selection: we retrieve chunks based on the highest ROUGE-L or BLEU-1 scoring span with respect to the question in the story; comparing topic distributions from an LDA model (Blei et al., 2003) between questions and chunks according to their symmetric Kullback–Leibler divergence. Finally, we also consider the cosine similarity of TF-IDF[10] representations. We found that this approach lead to the best performance of the subsequently applied model on the validation set, irrespective of the number of chunks. Note that we used the answer as the query on the training, and the question for the validation and test.

Given the retrieved chunks, we experimented with several neural models using them as context. The AS Reader, which was the better-performing model on the summaries task, underperforms the simple no-context Seq2Seq baseline (shown in Table 6.4) in terms of MRR. While it does slightly better on the other metrics, it clearly fails to make use of the retrieved context to gain a distinctive margin over the no-context Seq2Seq model. Increasing the number of retrieved chunks, and thereby recall of possibly relevant parts of the story, had only a minor positive effect. The span prediction model—which here also uses selected chunks for context—does especially poorly in this setup. While this model provided the best neural results on the summaries task, we suspect that its performance was particularly badly hurt by the fact that there is so little lexical and grammatical overlap between the source of the questions (summaries) and the context provided (stories). As with the AS Reader, we observed no significant differences for varying number of chunks.

These results leave a large gap in human performance, highlighting the success of our design objective to build a task that is realistic and straight-forward for humans while very difficult for current reading comprehension models.

6.6 Qualitative Analysis and Challenges

We find that the proposed dataset meets the desiderata we set out in Section 6.3.1. In particular, we constructed a dataset with a number of long documents, characterized by good lexical coverage and diversity. The questions and answers are human generated and natural sounding. And, based on a small manual examination (of ‘Ghostbusters II’, ‘Airplane’,

[10]Term frequency–inverse document frequency, an information retrieval statistic of word importance in a document relative to a corpus of documents.

Title: Armageddon 2419 A.D.
Question: In what year did Rogers awaken from his deep slumber?
Answer: 2419
Summary snippet: ...Rogers remained in sleep for 492 years. He awakes in 2419 and,...
Story snippet: I should state therefore, that I, Anthony Rogers, am, so far as I know, the only man alive whose normal span of eighty-one years of life has been spread over a period of 573 years. To be precise, I lived the first twenty-nine years of my life between 1898 and 1927; the other fifty-two since 2419. The gap between these two, a period of nearly five hundred years, I spent in a state of suspended animation, free from the ravages of katabolic processes, and without any apparent effect on my physical or mental faculties. When I began my long sleep, man had just begun his real conquest of the air...

Figure 6.5: Example question–answer pair with snippets from the summary and the story.

'Jacob's Ladder'), only a small number of questions and answers are shallow paraphrases of sentences in the full document. Most questions require reading segments at least several paragraphs long, and in some cases even multiple segments spread throughout the story.

Computational challenges identified in Section 6.5.3 naturally suggest a retrieval procedure as the first step. We found that the retrieval is challenging, even for humans not familiar with the presented narrative. In particular, the task often requires referring to larger parts of the story, in addition to knowing at least some background about entities. This makes the search procedure, based on only a short question, a challenging and interesting task in itself.

We show example question–answer pairs in Figures 6.1, 6.5, 6.12. These examples were chosen from a small set of manually annotated question–answer pairs to be representative of this collection. In particular, the examples show that larger parts of the story are required to answer questions. Consider Figure 6.12. While the relevant paragraph depicting the injury appears early on, it is not until the next snippet (which appears at the end of the narrative) that the lethal consequences of the injury are revealed. This illustrates an iterative reasoning process as well as extremely long temporal dependencies we encountered during manual annotation. As shown in Figure 6.1, reading comprehension on movie scripts requires understanding of written dialogue. This is a challenge as dialogue is typically non-descriptive, whereas the questions were asked based on descriptive summaries, requiring models to "read between the lines".

We expect that understanding narratives as complex as those presented in NarrativeQA

will require transferring text understanding capability from other supervised learning tasks. For example, developing models with language understanding ability in a multitask setting (Wang et al., 2018, e.g.) and applying them in parallel to the full stories, conditioned on the question, as a preprocessing step.

6.7 NarrativeQA Examples

See Figures 6.6–6.11.

Title: Ghostbusters II
Question: What job does the mayor want to have?
Answer: Governor
Summary snippet: The Ghostbusters go to the mayor with their suspicions, but are dismissed; the mayor's assistant, Jack Hardemeyer, has them committed to a psychiatric hospital to protect the mayor's campaign for governor.
Story snippet:

HARDEMEYER (bristling)
Look, you stay away from the mayor. Next fall, barring a disaster, he's going to be elected governor of this state and the last thing we need is for him to be associated with two-bit frauds and publicity hounds like you and your friends. You read me?

Figure 6.6: Example question–answer pair from NarrativeQA.

Title: Ghostbusters II
Question: What New York City landmark helps the Ghostbusters get into the museum?
Answer: The Statue of Liberty
Summary snippet: As they arrive at the museum , the slime begins to recede and they use the Statue s flaming torch to break through a skylight. . .
Story snippet:
INT. MUSEUM - JANOSZ'S POV - SKYLIGHT - NIGHT
The Statue of Liberty is looming over the skylight looking down on Janosz with an expression of righteous anger on it's face.

EXT. MUSEUM - NIGHT (CONTINUOUS ACTION)
Kneeling beside the museum, the statue draws back it's mighty right arm and smashes the skylight with its torch.

Figure 6.7: Example question–answer pair from NarrativeQA.

Title: Ghostbusters II
Question: How does the slime get into Dana's apartment?
Answer: Through the bathtub
Summary snippet: Later, the slime invades Dana's apartment via the bathtub and attacks her and Oscar.
Story snippet:
INT. DANA'S APARTMENT - NIGHT
Dana brings Oscar into the bathroom and lays him on the bassinet. She's wearing a robe over her nightgown, preparing to bathe the baby. She turns the taps on the old claw-footed bathtub, checks the water temperature, then turns away and starts to undress the baby.

DANA (talking sweetly to the baby)
Look at you. I think we got more food on your shirt than we got in your mouth.

BATHTUB
The water pouring from the faucet changes to slime and settles at the bottom of the tub. Dana reaches over and turns off the water without looking into the tub. When she turns away, both taps start to spin by themselves and the tub flexes and bulges.

Figure 6.8: Example question–answer pair from NarrativeQA.

Title: Ghostbusters II
Question: What group did Dana's ex husband join causing their divorce?
Answer: London Symphony Orchestra
Summary snippet: Peter's former girlfriend Dana Barrett has had a son, Oscar, with a violinist whom she married then divorced when he received an offer to join the London Symphony Orchestra.
Story snippet:

VENKMAN
So what happened to Mr. Right? I hear he ditched you and the kid and moved to Europe.

DANA
He didn't "ditch" me. We had some problems, he got a good offer from an orchestra in England and he took it.

Figure 6.9: Example question–answer pair from NarrativeQA.

Title: Ghostbusters II
Question: Who is the first person that falls under Vigo's spell?
Answer: Dr. Janosz Poha
Summary snippet: Meanwhile, Dana's colleague Dr. Janosz Poha has become increasingly obsessed with the glowering image of Vigo in the painting and falls under its spell. Vigo, whose spirit inhabits the painting, orders Janosz...
Story snippet:
PAINTING
The figure of Vigo comes to life, turns toward Janosz and gestures dramatically at him. Then he speaks to Janosz in a commanding voice.

VIGO
I, Vigo, the scourge of Carpathia, the sorrow of Moldavia, command you.

JANOSZ (in agony)
Command me, lord.

VIGO
On a mountain of skulls in a castle of pain, I sat on a throne of blood. What was will be, what is will be no more. Now is the season of evil. Find me a child that I might live again.

Bolts of red-hot energy shoot from the eyes of Vigo into Janosz's eyes. He screams and falls to his knees.

Figure 6.10: Example question–answer pair from NarrativeQA.

Title: Ghostbusters II
Question: Where does the spirit kidnap Oscar from?
Answer: Peter's apartment.
Summary snippet: Meanwhile, a spirit resembling Janosz as a nanny kidnaps Oscar from Peter's apartment...
Story snippet:
INT. VENKMAN'S BEDROOM - NIGHT (CONTINUOUS ACTION)
Dana enters and immediately notices that the crib is empty and the window is open.

DANA (screams)

Louis!

Frantic now, Dana rushes to the window and looks out, as Louis and Janine come running in.

EXT. WINDOW LEDGE - DANA'S POV - NIGHT (ECLIPSE)
The baby is standing out on the ledge at the corner of the building, fifty feet above the street, staring off into the distance as if he's waiting for something.

EXT. WINDOW LEDGE - NIGHT (ECLIPSE) (CONTINUOUS ACTION)
Dana climbs out onto the ledge and starts inching slowly toward the baby. Then she stops as a miraculous apparition materializes.

LOUIS AND JANINE
They lean out the window, gaping at the apparition.

EXT. VENKMAN'S LEDGE - APPARITION A sweet, kindly-looking English nanny appears, pushing a pram, strolling on thin air parallel to the ledge high above the ground. Her face looks remarkably like Janosz Poha's. The nanny extends her hand to the BABY who GURGLES sweetly as he reaches out to take it.

DANA
She watches in helpless horror.

DANA (screams)

No!!

GHOST NANNY
She picks up the baby and lays it gently in the pram, then turns and smiles at Dana. The smile turns to a hideous grin, then the nanny shrieks at Dana and takes off like a shot with the baby.

Figure 6.11: Example question–answer pair from NarrativeQA.

6.8 Related Work

This chapter is the first large-scale question answering dataset on full-length books and movie scripts. However, although we are the first to look at the QA task, learning to understand books through other modelling objectives has become an important subproblem in NLP (see Section 2.1.2). In computer vision, the MovieQA dataset (Tapaswi et al., 2016) fulfils a similar role as NarrativeQA. It seeks to test the ability of models to comprehend movies via question answering, and part of the dataset includes full length scripts.

6.9 Summary

We have introduced a new dataset and a set of tasks for training and evaluating reading comprehension systems, born from an analysis of the limitations of existing datasets and tasks. While our QA task resembles tasks provided by existing datasets, it exposes new challenges because of its domain: fiction. Fictional stories—in contrast to news stories—are self-contained and describe richer set of entities, events, and the relations between them. We have a range of tasks, from simple (which requires models to read *summaries* of books and movie scripts, and generate or rank fluent English answers to human-generated questions) to more complex (which requires models to read the full *stories* to answer the questions, with no access to the summaries).

In addition to the issue of scaling neural models to large documents, the larger tasks are significantly more difficult as questions formulated based on one or two sentences of a summary might require appealing to possibly discontiguous sentences or paragraphs from the source text. This requires potential solutions to these tasks to jointly model the process of searching for information (possibly in several steps) to serve as support for generating an answer, alongside the process of generating the answer entailed by said support. End-to-end mechanisms for both searching for information, such as attention, do not scale beyond selecting words or n-grams in short contexts such as sentences and small documents. Likewise, neural models for mapping documents to answers, or determining entailment between supporting evidence and a hypothesis, typically operate on the scale of sentences rather than sets of paragraphs.

We have provided baseline and benchmark results for both sets of tasks, demonstrating that while existing models give sensible results out of the box on summaries, they do not get

Title: Jacob's Ladder
Question: What is the fatal injury that Jacob sustains which ultimately leads to his death ?
Answer: A bayonete stabbing to his gut.
Summary snippet: A terrified Jacob flees into the jungle, only to be bayoneted in the gut by an unseen assailant.

[. . .]

In a wartime triage tent in 1971, military doctors fruitlessly treating Jacob reluctantly declare him dead
Story snippet: As he spins around one of the attackers jams all eight inches of his bayonet blade into Jacob's stomach. Jacob screams. It is a loud and piercing wail.

[. . .]

Int. Vietnam Field Hospital - Day

A doctor leans his head in front of the lamp and removes his mask. His expression is somber. He shakes his head. His words are simple and final.

DOCTOR
He's gone.

Cut to Jacob Singer . . .

The doctor steps away. A nurse rudely pulls a green sheet up over his head. The doctor turns to one of the aides and throws up his hands in defeat.

Figure 6.12: Example question–answer pair with snippets from the summary and the story.

any traction on the book-scale tasks. Having given a quantitative and qualitative analysis of the difficulty of the more complex tasks, we suggest research directions that may help bridge the gap between existing models and human performance. Our hope is that this dataset will serve not only as a challenge for the machine reading community, but as a driver for the development of a new class of neural models which will take a significant step beyond the level of complexity which existing datasets and tasks permit.

Chapter 7

Conclusions and Future Work

In this thesis we have explored tasks and models that allow us to teach language understanding. We used recent deep learning techniques to tackle semantic parsing with the usual supervised data constraints, and we proposed a series of complex reading comprehension tasks and models for them.

In Chapter 3 our semi-supervised autoencoder model that treats language as a hidden sequential latent variable was effectively able to utilize unpaired logical forms to improve on the supervised model performance. The semi-supervised approach is essential to train deep learning models due to small semantic parsing datasets. We have demonstrated the effectiveness on three semantic parsing tasks. While we focused on tasks with little supervised data and additional unsupervised data in y, it would be straightforward to reverse the model to train it with additional unpaired data in x, i.e. on the natural language side. A natural extension would also be a formulation where semi-supervised training was performed in both x and y. For instance, machine translation lends itself to such a formulation where for many language pairs parallel data may be scarce while there is an abundance of monolingual data. Furthermore, the choice of logistic-normal distribution for the sequence of latent variables representing natural language could be approached alternatively using the Concrete Distribution (Maddison et al., 2017) which might better approximate the intended generative process.

In Chapter 4 we introduced a novel reading comprehension task with a dataset large enough that it permits training deep learning models for the first time, moreover, as language understanding is a complex behaviour, any system that will learn this from scratch will need lots of data. We carefully designed the task, through entity anonymization and shuffling, to avoid the use of the language modelling signal when solving the Cloze-style

questions. We found that our attention models are effective for this task, and the models were further refined by other researches (e.g., Attention Sum Reader).

Our work on the CNN and the Daily Mail datasets has successfully encouraged development of new deep learning reading comprehension models and datasets. However, in our dataset the highlights were sometimes extracted mostly verbatim from the documents, which made the task amenable to shallow text matching using attention models. Furthermore, the subsequent datasets often annotate questions based on the document we want to read and comprehend, which through limitations of the annotation procedure, leads to questions answerable using a short localized context. We have reviewed this work in Chapter 5.

NarrativeQA, a complex narrative understanding dataset of books and movie scripts, which we introduced in Chapter 6, provides the next interesting challenge for reading comprehension models. The dataset introduces several challenges, which might be approachable independently, including processing much longer documents, and more fine understanding of semantics. We have seen that the questions often require understanding larger parts of the context and sometime even multiple locations in the context. The task we constructed seems to be quite hard and will hopefully provide a challenge for the future. The dataset contains 1,567 documents with 60,000 tokens on average, and about 32,000 training question-answer pairs—it is unclear whether the dataset is large enough to teach models reading comprehension, due to possibly weak signal where to find answers in the long documents.

Future directions to tackle this problem could focus on improving the information retrieval part of the models, which was inadequate for this specific retrieval problem, perhaps by a similar approach to Chen et al. (2017). Another approach to retrieve relevant parts of documents, specifically movie scripts, was explored by Gorinski and Lapata (2015) and could be possibly extended to the setting of NarrativeQA. To sidestep the explicit retrieval problem, a model could focus on learning about entities/characters, as these are the central parts of narratives, similar to Entity Networks (Henaff et al., 2017). Models that leverage transfer of natural language understanding ability from other tasks could be potentially also be used to process in parallel the narratives in NarrativeQA.

There still remain many open questions on how to train models for understanding language. We believe they should focus on two directions: more precise and fine under-

standing of semantics, and applying this to large corpora of documents and data of other modalities. The approaches we will see will need to utilize better transfer learning for understanding of language, and perhaps utilization of existing datasets through self-imposed model restrictions to avoid shallow solutions. Furthermore, there is an increasing number of heterogeneous reading comprehension datasets, and perhaps we should try to leverage all this data at once for learning, and test on a challenging task.

We tackled the problem of learning and testing natural language understanding through the task of reading comprehension. We introduced a number of challenging tasks which we hope will encourage future research.

References

W. Ammar, C. Dyer, and N. A. Smith. 2014. Conditional Random Field Autoencoders for Unsupervised Structured Prediction. In *Proceedings of NIPS*.

J. Andreas and D. Klein. 2015. Alignment-based Compositional Semantics for Instruction Following. In *Proceedings of EMNLP*, September.

J. Andreas, A. Vlachos, and S. Clark. 2013. Semantic Parsing as Machine Translation. In *Proceedings of ACL*, August.

Y. Artzi and L. Zettlemoyer. 2013. Weakly Supervised Learning of Semantic Parsers for Mapping Instructions to Actions. *Transactions of the Association for Computational Linguistics (TACL)*, 1(1):49–62.

Y. Artzi, D. Das, and S. Petrov. 2014. Learning Compact Lexicons for CCG Semantic Parsing. In *Proceedings of EMNLP*, October.

D. Bahdanau, K. Cho, and Y. Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, California, May.

O. Bajgar, R. Kadlec, and J. Kleindienst. 2016. Embracing data abundance: BookTest Dataset for Reading Comprehension. *CoRR*, abs/1610.00956.

D. Bamman, B. O'Connor, and N. A. Smith. 2014a. Learning latent personas of film characters. In *Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL)*, page 352.

D. Bamman, T. Underwood, and N. A. Smith. 2014b. A Bayesian Mixed Effects Model of Literary Character. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 370–379, Baltimore, Maryland, June. Association for Computational Linguistics.

J. Berant and P. Liang. 2014. Semantic Parsing via Paraphrasing. In *Proceedings of ACL*, June.

J. Berant, A. Chou, R. Frostig, and P. Liang. 2013. Semantic Parsing on Freebase from Question-Answer Pairs. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1533–1544.

J. B. Black and R. Wilensky. 1979. An Evaluation of Story Grammars. *Cognitive Science*, 3(3):213–229.

D. M. Blei, A. Y. Ng, and M. I. Jordan. 2003. Latent dirichlet allocation. *Journal of machine Learning research*, 3(Jan):993–1022.

A. Bordes, N. Usunier, S. Chopra, and J. Weston. 2015. Large-scale Simple Question Answering with Memory Networks. *CoRR*, abs/1506.02075.

C. J. Burges. 2013. Towards the Machine Comprehension of Text: An Essay. Technical report, December.

N. Chambers and D. Jurafsky. 2009. Unsupervised Learning of Narrative Schemas and Their Participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 602–610, Stroudsburg, PA, USA. Association for Computational Linguistics.

S. Chaturvedi, M. Iyyer, and H. Daumé III. 2017. Unsupervised Learning of Evolving Relationships Between Literary Characters. In *Association for the Advancement of Artificial Intelligence*.

S. F. Chen and J. Goodman. 1999. An empirical study of smoothing techniques for language modeling. *Computer Speech & Language*, 13(4):359–393.

D. L. Chen and R. J. Mooney. 2011. Learning to Interpret Natural Language Navigation Instructions from Observations. In *Proceedings of AAAI*, August.

D. Chen, J. Bolton, and C. D. Manning. 2016a. A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL) (Volume 1: Long Papers)*, pages 2358–2367, Berlin, Germany, August. Association for Computational Linguistics.

D. Chen, J. Bolton, and C. D. Manning. 2016b. A Thorough Examination of the CNN/Daily Mail Reading Comprehension Task. *CoRR*, abs/1606.02858.

D. Chen, A. Fisch, J. Weston, and A. Bordes. 2017. Reading Wikipedia to Answer Open-Domain Questions. In *Association for Computational Linguistics (ACL)*.

J. Cheng, L. Dong, and M. Lapata. 2016. Long Short-Term Memory-Networks for Machine Reading. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 551–561, Austin, Texas, November. Association for Computational Linguistics.

K. Cho, B. van Merrienboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. 2014. Learning Phrase Representations using RNN Encoder–Decoder for Statistical Machine Translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, October. Association for Computational Linguistics.

R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa. 2011. Natural Language Processing (Almost) from Scratch. *Journal of Machine Learning Research*, 12:2493–2537, November.

Y. Cui, Z. Chen, S. Wei, S. Wang, T. Liu, and G. Hu. 2016. Attention-over-Attention Neural Networks for Reading Comprehension. *CoRR*, abs/1607.04423.

D. Das, D. Chen, A. F. T. Martins, N. Schneider, and N. A. Smith. 2013. Frame-Semantic Parsing. *Computational Linguistics*, 40(1):9–56.

M. Denkowski and A. Lavie. 2011. Meteor 1.3: Automatic Metric for Reliable Optimization and Evaluation of Machine Translation Systems. In *Proceedings of the EMNLP 2011 Workshop on Statistical Machine Translation*.

B. Dhingra, H. Liu, W. W. Cohen, and R. Salakhutdinov. 2016. Gated-Attention Readers for Text Comprehension. *CoRR*, abs/1606.01549.

B. Dhingra, K. Mazaitis, and W. W. Cohen. 2017. Quasar: Datasets for Question Answering by Search and Reading. *arXiv preprint arXiv:1707.03904*.

L. Dong and M. Lapata. 2016. Language to Logical Form with Neural Attention. *arXiv preprint arXiv:1601.01280*.

M. Dunn, L. Sagun, M. Higgins, U. Guney, V. Cirik, and K. Cho. 2017. SearchQA: A New Q&A Dataset Augmented with Context from a Search Engine. *arXiv preprint arXiv:1704.05179*.

J. L. Elman. 1990. Finding structure in time. *COGNITIVE SCIENCE*, 14(2):179–211.

L. Frermann and G. Szarvas. 2017. Inducing Semantic Micro-Clusters from Deep Multi-View Representations of Novels. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1874–1884. Association for Computational Linguistics.

L. Frermann, S. B. Cohen, and M. Lapata. 2017. Whodunnit? Crime Drama as a Case for Natural Language Understanding. *Transactions of the Association for Computational Linguistics (TACL)*.

M. Galley, M. Hopkins, K. Knight, and D. Marcu. 2004. What's in a translation rule? In *Proceedings of HLT-NAACL*, May.

F. A. Gers, N. N. Schraudolph, and J. Schmidhuber. 2003. Learning Precise Timing with Lstm Recurrent Networks. *Journal of Machine Learning Research*, 3:115–143, March.

C. Goller and A. Küchler. 1996. Learning Task-Dependent Distributed Representations by Backpropagation Through Structure. In *Proceedings of the International Conference on Neural Networks (ICNN-96)*, pages 347–352. IEEE.

D. Golovin, B. Solnik, S. Moitra, G. Kochanski, J. Karro, and D. Sculley. 2017. Google Vizier: A Service for Black-Box Optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '17, pages 1487–1495, New York, NY, USA. ACM.

P. J. Gorinski and M. Lapata. 2015. Movie Script Summarization as Graph-based Scene Extraction. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1066–1076, Denver, Colorado, May–June. Association for Computational Linguistics.

A. Graves. 2012. *Supervised Sequence Labelling with Recurrent Neural Networks*, volume 385 of *Studies in Computational Intelligence*. Springer.

B. Green, A. Wolf, C. Chomsky, and K. Laugherty. 1961. BASEBALL: An automatic question answerer. In *Proceedings Western Joint IRE-AIEE-ACM Computing Conference*, volume 19, pages 219–224, Los Angeles, CA.

K. Gregor, I. Danihelka, A. Graves, and D. Wierstra. 2015. DRAW: A Recurrent Neural Network For Image Generation. *CoRR*, abs/1502.04623.

Ç. Gülçehre, O. Firat, K. Xu, K. Cho, L. Barrault, H. Lin, F. Bougares, H. Schwenk, and Y. Bengio. 2015. On Using Monolingual Corpora in Neural Machine Translation. *arXiv preprint arXiv:1503.03535*.

C. Haas and S. Riezler. 2016. A Corpus and Semantic Parser for Multilingual Natural Language Querying of OpenStreetMap. In *Proceedings of NAACL*, June.

M. Henaff, J. Weston, A. Szlam, A. Bordes, and Y. LeCun. 2017. Tracking the World State with Recurrent Entity Networks.

K. M. Hermann, D. Das, J. Weston, and K. Ganchev. 2014. Semantic Frame Identification with Distributed Word Representations. In *Proceedings of ACL*. Association for Computational Linguistics, June.

K. M. Hermann, T. Kočiský, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom. 2015. Teaching Machines to Read and Comprehend. In *Proceedings of NIPS*.

D. Hewlett, A. Lacoste, L. Jones, I. Polosukhin, A. Fandrianto, J. Han, M. Kelcey, and D. Berthelot. 2016. WikiReading: A Novel Large-scale Language Understanding Task over Wikipedia. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1535–1545. Association for Computational Linguistics.

F. Hill, A. Bordes, S. Chopra, and J. Weston. 2016. The Goldilocks Principle: Reading Children's Books with Explicit Memory Representations. In *Proceedings of the 4th International Conference on Learning Representations (ICLR)*, San Juan, Puerto Rico, May.

L. Hirschman, M. Light, E. Breck, and J. D. Burger. 1999. Deep Read: A Reading Comprehension System. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics on Computational Linguistics*, ACL '99, pages 325–332, Stroudsburg, PA, USA. Association for Computational Linguistics.

S. Hochreiter and J. Schmidhuber. 1997. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November.

M. Iyyer, A. Guha, S. Chaturvedi, J. Boyd-Graber, and H. Daumé III. 2016. Feuding Families and Former Friends: Unsupervised Learning for Dynamic Fictional Relationships. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1534–1544. Association for Computational Linguistics.

R. Jia and P. Liang. 2016. Data Recombination for Neural Semantic Parsing. In *Association for Computational Linguistics (ACL)*.

B. Jones, J. Andreas, D. Bauer, K. M. Hermann, and K. Knight. 2012. Semantics-Based Machine Translation with Hyperedge Replacement Grammars. In *Proceedings of COLING 2012*, December.

M. Joshi, E. Choi, D. S. Weld, and L. Zettlemoyer. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. *CoRR*, abs/1705.03551.

D. Jurafsky and J. H. Martin. 2009. *Speech and Language Processing (2Nd Edition)*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA.

R. Kadlec, M. Schmid, O. Bajgar, and J. Kleindienst. 2016. Text Understanding with the Attention Sum Reader Network. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL) (Volume 1: Long Papers)*, pages 908–918, Berlin, Germany, August. Association for Computational Linguistics.

N. Kalchbrenner, E. Grefenstette, and P. Blunsom. 2014. A Convolutional Neural Network for Modelling Sentences. In *Proceedings of ACL*, June.

J. Kim and R. J. Mooney. 2012. Unsupervised PCFG Induction for Grounded Language Learning with Highly Ambiguous Supervision. In *Proceedings of EMNLP-CoNLL*, July.

J. Kim and R. Mooney. 2013. Adapting Discriminative Reranking to Grounded Language Learning. In *Proceedings of ACL*, August.

D. P. Kingma and M. Welling. 2014. Auto-Encoding Variational Bayes. In *Proceedings of ICLR*.

S. Kobayashi, R. Tian, N. Okazaki, and K. Inui. 2016. Dynamic Entity Representation with Max-pooling Improves Machine Reading. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 850–855, San Diego, California, June. Association for Computational Linguistics.

O. Kolomiyets and M.-F. Moens. 2011. A Survey on Question Answering Technology from an Information Retrieval Perspective. *Information Sciences*, 181(24):5412–5434, December.

T. Kočiský, G. Melis, E. Grefenstette, C. Dyer, W. Ling, P. Blunsom, and K. M. Hermann. 2016. Semantic Parsing with Semi-Supervised Sequential Autoencoders. In *EMNLP*, November.

T. Kočiský, J. Schwarz, P. Blunsom, C. Dyer, K. M. Hermann, G. Melis, and E. Grefenstette. 2018. The NarrativeQA Reading Comprehension Challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

T. Kwiatkowski, L. Zettlemoyer, S. Goldwater, and M. Steedman. 2011. Lexical Generalization in CCG Grammar Induction for Semantic Parsing. In *Proceedings of EMNLP*.

T. Kwiatkowski, E. Choi, Y. Artzi, and L. Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *In Proceedings of EMNLP*.

P. Liang, M. I. Jordan, and D. Klein. 2011. Learning Dependency-based Compositional Semantics. In *Proceedings of the ACL-HLT*.

P. Liang, M. I. Jordan, and D. Klein. 2013. Learning dependency-based compositional semantics. *Computational Linguistics*, 39(2):389–446.

C.-Y. Lin. 2004. ROUGE: A Package for Automatic Evaluation of summaries. In *Proc. ACL workshop on Text Summarization Branches Out*, page 10.

M. MacMahon, B. Stankiewicz, and B. Kuipers. 2006. Walk the Talk: Connecting Language, Knowledge, and Action in Route Instructions. In *Proceedings of AAAI*.

C. J. Maddison, A. Mnih, and Y. W. Teh. 2017. The Concrete Distribution: A Continuous Relaxation of Discrete Random Variables. In *Proceedings of ICLR*.

D. Marcheggiani and I. Titov. 2016. Discrete-State Variational Autoencoders for Joint Discovery and Factorization of Relations. *Transactions of ACL*.

H. Mei, M. Bansal, and M. R. Walter. 2016. Listen, Attend, and Walk: Neural Mapping of Navigational Instructions to Action Sequences. In *Proceedings of AAAI*.

V. Mnih, N. Heess, A. Graves, and K. Kavukcuoglu. 2014. Recurrent Models of Visual Attention. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 2204–2212. Curran Associates, Inc.

R. Nallapati, B. Xiang, and B. Zhou. 2016. Sequence-to-Sequence RNNs for Text Summarization. *CoRR*, abs/1602.06023.

T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and L. Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *CoRR*, abs/1611.09268.

K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of ACL*.

J. Pennington, R. Socher, and C. D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *Processing of EMNLP*.

H. Poon, J. Christensen, P. Domingos, O. Etzioni, R. Hoffmann, C. Kiddon, T. Lin, X. Ling, Mausam, A. Ritter, S. Schoenmackers, S. Soderland, D. Weld, F. Wu, and C. Zhang. 2010. Machine Reading at the University of Washington. In *Proceedings of the NAACL HLT 2010 First International Workshop on Formalisms and Methodology for Learning by Reading*, FAM-LbR '10, pages 87–95.

P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of EMNLP*.

S. Reddy, M. Lapata, and M. Steedman. 2014. Large-scale Semantic Parsing without Question-Answer Pairs. *Transactions of the Association for Computational Linguistics*, 2:377–392.

M. Richardson, C. J. Burges, and E. Renshaw. 2013a. 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 (EMNLP)*.

M. Richardson, C. J. C. Burges, and E. Renshaw. 2013b. MCTest: A Challenge Dataset for the Open-Domain Machine Comprehension of Text. In *Proceedings of EMNLP*, pages 193–203. ACL.

E. Riloff and M. Thelen. 2000. A Rule-based Question Answering System for Reading Comprehension Tests. In *Proceedings of the ANLP/NAACL Workshop on Reading Comprehension Tests As Evaluation for Computer-based Language Understanding Sytems*, ANLP/NAACL-ReadingComp '00, pages 13–19, Stroudsburg, PA, USA. Association for Computational Linguistics.

R. C. Schank and R. P. Abelson. 1977. *Scripts, Plans, Goals and Understanding: an Inquiry into Human Knowledge Structures*. L. Erlbaum, Hillsdale, NJ.

J. Schmidhuber. 2014. Deep Learning in Neural Networks: An Overview. *CoRR*, abs/1404.7828.

A. See, P. J. Liu, and C. D. Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083. Association for Computational Linguistics.

M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi. 2016. Bidirectional Attention Flow for Machine Comprehension. *arXiv preprint arXiv:1611.01603*.

M. J. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi. 2017. Bidirectional Attention Flow for Machine Comprehension. In *Proceedings of the 5th International Conference on Learning Representations (ICLR)*, Toulon, France, April.

Y. Shen, P. Huang, J. Gao, and W. Chen. 2016. ReasoNet: Learning to Stop Reading in Machine Comprehension. *CoRR*, abs/1609.05284.

A. Sordoni, P. Bachman, and Y. Bengio. 2016. Iterative Alternating Neural Attention for Machine Reading. *CoRR*, abs/1606.02245.

S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus. 2015. End-To-End Memory Networks. *CoRR*, abs/1503.08895.

S. Suster, I. Titov, and G. van Noord. 2016. Bilingual Learning of Multi-sense Embeddings with Discrete Autoencoders. *CoRR*, abs/1603.09128.

I. Sutskever, O. Vinyals, and Q. V. Le. 2014a. Sequence to sequence learning with neural networks. In *Proceedings of NIPS*.

I. Sutskever, O. Vinyals, and Q. V. V. Le. 2014b. Sequence to Sequence Learning with Neural Networks. In Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3104–3112. Curran Associates, Inc.

K. Svore, L. Vanderwende, and C. Burges. 2007. Enhancing Single-Document Summarization by Combining RankNet and Third-Party Sources. In *Proceedings of EMNLP/CoNLL*, pages 448–457, Prague, Czech Republic, June. Association for Computational Linguistics.

M. Tapaswi, Y. Zhu, R. Stiefelhagen, A. Torralba, R. Urtasun, and S. Fidler. 2016. MovieQA: Understanding Stories in Movies through Question-Answering. In *IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

W. L. Taylor. 1953. "Cloze procedure": a new tool for measuring readability. *Journalism Quarterly*, 30:415–433.

T. Tieleman and G. Hinton. 2012. Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning.

A. Trischler, T. Wang, X. Yuan, J. Harris, A. Sordoni, P. Bachman, and K. Suleman. 2016a. NewsQA: A Machine Comprehension Dataset. *CoRR*, abs/1611.09830.

A. Trischler, Z. Ye, X. Yuan, P. Bachman, A. Sordoni, and K. Suleman. 2016b. Natural Language Comprehension with the EpiReader. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 128–137, Austin, Texas, November. Association for Computational Linguistics.

A. M. Turing. 1950. Computing Machinery and Intelligence. *Mind*, 59(236):433–460.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. 2017. Attention is All you Need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 5998–6008. Curran Associates, Inc.

O. Vinyals, M. Fortunato, and N. Jaitly. 2015a. Pointer networks. In *Proceedings of NIPS*.

O. Vinyals, L. Kaiser, T. Koo, S. Petrov, I. Sutskever, and G. Hinton. 2015b. Grammar as a Foreign Language. In *Proceedings of NIPS*.

S. Wang and J. Jiang. 2016. Machine comprehension using match-lstm and answer pointer. *arXiv preprint arXiv:1608.07905*.

W. Wang, N. Yang, F. Wei, B. Chang, and M. Zhou. 2017. Gated Self-Matching Networks for Reading Comprehension and Question Answering. In *Proceedings of ACL*.

A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. 2018. GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding. arXiv preprint 1804.07461.

D. Weissenborn, G. Wiese, and L. Seiffe. 2017. FastQA: A Simple and Efficient Neural Architecture for Question Answering. *CoRR*, abs/1703.04816.

J. Weizenbaum. 1966. ELIZA&Mdash;a Computer Program for the Study of Natural Language Communication Between Man and Machine. *Commun. ACM*, 9(1):36–45, January.

J. Weston, A. Bordes, S. Chopra, and T. Mikolov. 2015a. Towards AI-Complete Question Answering: A Set of Prerequisite Toy Tasks. *CoRR*, abs/1502.05698.

J. Weston, S. Chopra, and A. Bordes. 2015b. Memory Networks. In *Proceedings of the 3rd International Conference on Learning Representations (ICLR)*, San Diego, California, May.

R. Wilensky. 1978. Why John married Mary: Understanding stories involving recurring goals. *Cognitive Science*, 2(3):235–266.

T. Winograd. 1972a. Procedures as a Representation for Data in a Computer Program for Understanding Natural Language. *Cognitive Psychology*, 3(1):1–191.

T. Winograd. 1972b. *Understanding Natural Language*. Academic Press, Inc., Orlando, FL, USA.

Y. W. Wong and R. J. Mooney. 2006. Learning for Semantic Parsing with Statistical Machine Translation. In *Proceedings of NAACL*.

W. A. Woods, R. M. Kaplan, and B. Nash-Webber. 1972. The Lunar Sciences Natural Language Information System: Final report. Technical Report 2378, Bolt, Beranek, and Newman, Inc., Cambridge, MA.

K. Woodsend and M. Lapata. 2010. Automatic Generation of Story Highlights. In *Proceedings of ACL*, pages 565–574, Stroudsburg, PA, USA. Association for Computational Linguistics.

J. M. Zelle and R. J. Mooney. 1996. Learning to Parse Database Queries using Inductive Logic Programming. In *Proceedings of AAAI/IAAI*, pages 1050–1055, August.

L. S. Zettlemoyer and M. Collins. 2005. Learning to Map Sentences to Logical Form: Structured Classification with Probabilistic Categorial Grammars. In *UAI*, pages 658–666. AUAI Press.

L. Zettlemoyer and M. Collins. 2007. Online Learning of Relaxed CCG Grammars for Parsing to Logical Form. In *Proceedings of EMNLP-CoNLL*, June.

K. Zhao and L. Huang. 2014. Type-driven incremental semantic parsing with polymorphism. *arXiv preprint arXiv:1411.5379*.

J. Zhou and W. Xu. 2015. End-to-end Learning of Semantic Role Labeling Using Recurrent Neural Networks. In *Proceedings of ACL*.

B. Zoph and Q. V. Le. 2016. Neural Architecture Search with Reinforcement Learning. *CoRR*, abs/1611.01578.