Text Understanding with the Attention Sum Reader Network

Rudolf Kadlec, Martin Schmid, Ondrej Bajgar & Jan Kleindienst
IBM Watson
V Parku 4, Prague, Czech Republic
{rudolf_kadlec,martin.schmid,obajgar,jankle}@cz.ibm.com

Abstract

Several large cloze-style context-question-answer datasets have been introduced recently: the CNN and Daily Mail news data and the Children's Book Test. Thanks to the size of these datasets, the associated text comprehension task is well suited for deep-learning techniques that currently seem to outperform all alternative approaches. We present a new, simple model that uses attention to directly pick the answer from the context as opposed to computing the answer using a blended representation of words in the document as is usual in similar models. This makes the model particularly suitable for question-answering problems where the answer is a single word from the document. Ensemble of our models sets new state of the art on all evaluated datasets.

1 Introduction

Most of the information humanity has gathered up to this point is stored in the form of plain text. Hence the task of teaching machines how to understand this data is of utmost importance in the field of Artificial Intelligence. One way of testing the level of text understanding is simply to ask the system questions for which the answer can be inferred from the text. A well-known example of a system that could make use of a huge collection of unstructured documents to answer questions is for instance IBM's Watson system used for the Jeopardy challenge (Ferrucci et al., 2010).

Cloze style questions (Taylor, 1953), i.e. questions formed by removing a phrase from a sentence, are an appealing form of such questions (for example see Figure 1). While the task is easy to evaluate, one can vary the context, the question sentence or the specific phrase missing in the question to dramatically change the task structure and difficulty.

Document: What was supposed to be a fantasy sports car ride at Walt Disney World Speedway turned deadly when a Lamborghini crashed into a guardrail. The crash took place Sunday at the Exotic Driving Experience, which bills itself as a chance to drive your dream car on a racetrack. The Lamborghini's passenger, 36-year-old Gary Terry of Davenport, Florida, died at the scene, Florida Highway Patrol said. The driver of the Lamborghini, 24-year-old Tavon Watson of Kissimmee, Florida, lost control of the vehicle, the Highway Patrol said. (...)
Question: Officials say the driver, 24-year-old Tavon Watson, lost control of a _____
Answer candidates: Tavon Watson, Walt Disney World Speedway, Highway Patrol, Lamborghini, Florida, (...)
Answer: Lamborghini

Figure 1: Each example consists of a context document, question, answer cadidates and, in the training data, the correct answer. This example was taken from the CNN dataset (Hermann et al., 2015). Anonymization of this example that makes the task harder is shown in Table 3.

One way of altering the task difficulty is to vary the word type being replaced, as in (Hill et al., 2015). The complexity of such variation comes from the fact that the level of context understanding needed in order to correctly predict different types of words varies greatly. While predicting prepositions can easily be done using relatively simple models with very little context knowledge, predicting named entities requires a deeper understanding of the context.

Also, as opposed to selecting a random sentence from a text (as done in (Hill et al., 2015)), the questions can be formed from a specific part of a document, such as a short summary or a list of

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 908–918,
Berlin, Germany, August 7-12, 2016.

	CNN			Daily Mail			CBT CN			CBT NE		
	train	valid	test	train	valid	test	train	valid	test	train	valid	test
# queries	380,298	3,924	3,198	879,450	64,835	53,182	120,769	2,000	2,500	108,719	2,000	2,500
Max # options	527	187	396	371	232	245	10	10	10	10	10	10
Avg # options	26.4	26.5	24.5	26.5	25.5	26.0	10	10	10	10	10	10
Avg # tokens	762	763	716	813	774	780	470	448	461	433	412	424
Vocab. size	118,497			208,045			53,185			53,063		

Table 1: Statistics on the 4 data sets used to evaluate the model. CBT CN stands for CBT Common Nouns and CBT NE stands for CBT Named Entites. Statistics were taken from (Hermann et al., 2015) and the statistics provided with the CBT data set.

tags. Since such sentences often paraphrase in a condensed form what was said in the text, they are particularly suitable for testing text comprehension (Hermann et al., 2015).

An important property of cloze style questions is that a large amount of such questions can be automatically generated from real world documents. This opens the task to data-hungry techniques such as deep learning. This is an advantage compared to smaller machine understanding datasets like MCTest (Richardson et al., 2013) that have only hundreds of training examples and therefore the best performing systems usually rely on handcrafted features (Sachan et al., 2015; Narasimhan and Barzilay, 2015).

In the first part of this article we introduce the task at hand and the main aspects of the relevant datasets. Then we present our own model to tackle the problem. Subsequently we compare the model to previously proposed architectures and finally describe the experimental results on the performance of our model.

2 Task and datasets

In this section we introduce the task that we are seeking to solve and relevant large-scale datasets that have recently been introduced for this task.

2.1 Formal Task Description

The task consists of answering a cloze style question, the answer to which is dependent on the understanding of a context document provided with the question. The model is also provided with a set of possible answers from which the correct one is to be selected. This can be formalized as follows:

The training data consist of tuples $(\mathbf{q}, \mathbf{d}, a, A)$, where $\mathbf{q}$ is a question, $\mathbf{d}$ is a document that contains the answer to question $\mathbf{q}$, A is a set of possible answers and $a \in A$ is the ground truth answer. Both $\mathbf{q}$ and $\mathbf{d}$ are sequences of words from vocabulary V. We also assume that all possible answers are words from the vocabulary, that is $A \subseteq V$, and that the ground truth answer a appears in the document, that is $a \in \mathbf{d}$.

2.2 Datasets

We will now briefly summarize important features of the datasets.

2.2.1 News Articles — CNN and Daily Mail

The first two datasets[1] (Hermann et al., 2015) were constructed from a large number of news articles from the CNN and Daily Mail websites. The main body of each article forms a context, while the cloze style question is formed from one of short highlight sentences, appearing at the top of each article page. Specifically, the question is created by replacing a named entity from the summary sentence (*e.g. "Producer X will not press charges against Jeremy Clarkson, his lawyer says."*).

Furthermore the named entities in the whole dataset were replaced by anonymous tokens which were further shuffled for each example so that the model cannot build up any world knowledge about the entities and hence has to genuinely rely on the context document to search for an answer to the question.

Qualitative analysis of reasoning patterns needed to answer questions in the CNN dataset together with human performance on this task are provided in (Chen et al., 2016).

[1]The CNN and Daily Mail datasets are available at `https://github.com/deepmind/rc-data`

2.2.2 Children's Book Test

The third dataset[2], the Children's Book Test (CBT) (Hill et al., 2015), is built from books that are freely available thanks to Project Gutenberg[3]. Each context document is formed by 20 consecutive sentences taken from a children's book story. Due to the lack of summary, the cloze style question is then constructed from the subsequent (21^{st}) sentence.

One can also see how the task complexity varies with the type of the omitted word (named entity, common noun, verb, preposition). (Hill et al., 2015) have shown that while standard LSTM language models have human level performance on predicting verbs and prepositions, they lack behind on named entities and common nouns. In this article we therefore focus only on predicting the first two word types.

Basic statistics about the CNN, Daily Mail and CBT datasets are summarized in Table 1.

3 Our Model — Attention Sum Reader

Our model called the *Attention Sum Reader (AS Reader)*[4] is tailor-made to leverage the fact that the answer is a word from the context document. This is a double-edged sword. While it achieves state-of-the-art results on all of the mentioned datasets (where this assumption holds true), it cannot produce an answer which is not contained in the document. Intuitively, our model is structured as follows:

1. We compute a vector embedding of the query.
2. We compute a vector embedding of each individual word in the context of the whole document (*contextual embedding*).
3. Using a dot product between the question embedding and the contextual embedding of each occurrence of a candidate answer in the document, we select the most likely answer.

3.1 Formal Description

Our model uses one word embedding function and two encoder functions. The word embedding function e translates words into vector representations. The first encoder function is a document encoder f that encodes every word from the document $\mathbf{d}$ in the context of the whole document. We call this the *contextual embedding*. For convenience we will denote the contextual embedding of the i-th word in $\mathbf{d}$ as $f_i(\mathbf{d})$. The second encoder g is used to translate the query $\mathbf{q}$ into a fixed length representation of the same dimensionality as each $f_i(\mathbf{d})$. Both encoders use word embeddings computed by e as their input. Then we compute a weight for every word in the document as the dot product of its contextual embedding and the query embedding. This weight might be viewed as an attention over the document $\mathbf{d}$.

To form a proper probability distribution over the words in the document, we normalize the weights using the *softmax* function. This way we model probability s_i that the answer to query $\mathbf{q}$ appears at position i in the document $\mathbf{d}$. In a functional form this is:

$$s_i \propto \exp\left(f_i(\mathbf{d}) \cdot g(\mathbf{q})\right) \tag{1}$$

Finally we compute the probability that word w is a correct answer as:

$$P(w|\mathbf{q},\mathbf{d}) \propto \sum_{i \in I(w,\mathbf{d})} s_i \tag{2}$$

where $I(w,\mathbf{d})$ is a set of positions where w appears in the document $\mathbf{d}$. We call this mechanism *pointer sum attention* since we use attention as a pointer over discrete tokens in the context document and then we directly sum the word's attention across all the occurrences. This differs from the usual use of attention in sequence-to-sequence models (Bahdanau et al., 2015) where attention is used to blend representations of words into a new embedding vector. Our use of attention was inspired by Pointer Networks (Ptr-Nets) (Vinyals et al., 2015).

A high level structure of our model is shown in Figure 2.

3.2 Model instance details

In our model the document encoder f is implemented as a bidirectional Gated Recurrent Unit (GRU) network (Cho et al., 2014; Chung et al., 2014) whose hidden states form the contextual word embeddings, that is $f_i(\mathbf{d}) = \overrightarrow{f_i}(\mathbf{d}) \mid\mid \overleftarrow{f_i}(\mathbf{d})$, where $\mid\mid$ denotes vector concatenation and $\overrightarrow{f_i}$ and

[2]The CBT dataset is available at `http://www.thespermwhale.com/jaseweston/babi/CBTest.tgz`

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

[4]An implementation of AS Reader is available at `https://github.com/rkadlec/asreader`

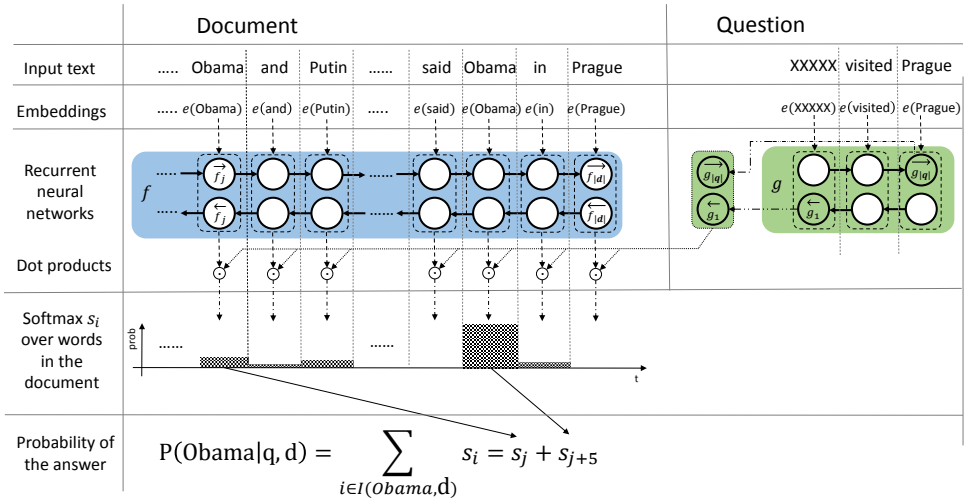


Figure 2: Structure of the model.

...

what was supposed to be a fantasy sports car ride at @entity3 turned deadly when a @entity4 crashed into a guardrail . the crash took place sunday at the @entity8 , which bills itself as a chance to drive your dream car on a racetrack . the @entity4 's passenger , 36 - year - old @entity14 of @entity15 , @entity16 , died at the scene , @entity13 said . the driver of the @entity4 , 24 - year - old @entity18 of @entity19 , @entity16 , lost control of the vehicle , the @entity13 said .

...

0 1

officials say the driver , 24 - year - old @entity18 , lost control of a _____

Figure 3: Attention in an example with anonymized entities where our system selected the correct answer. Note that the attention is focused only on named entities.

...

@entity11 film critic @entity29 writes in his review that "anyone nostalgic for childhood dreams of transformation will find something to enjoy in an uplifting movie that invests warm sentiment in universal themes of loss and resilience , experience and maturity . " more : the best and worst adaptations of "@entity" @entity43, @entity44 and @entity46 star in director @entity48's crime film about a hit man trying to save his estranged son from a revenge plot. @entity11 chief film critic @entity52 writes in his review that the film

...

0 1

_____ stars in crime film about hit man trying to save his estranged son

Figure 4: Attention over an example where our system failed to select the correct answer (entity43). The system was probably mislead by the co-occurring word 'film'. Namely, entity11 occurs 7 times in the whole document and 6 times it is together with the word 'film'. On the other hand, the correct answer occurs only 3 times in total and only once together with 'film'.

$\overleftarrow{f_i}$ denote forward and backward contextual embeddings from the respective recurrent networks. The query encoder g is implemented by another bidirectional GRU network. This time the last hidden state of the forward network is concatenated with the last hidden state of the backward network to form the query embedding, that is $g(\mathbf{q}) = \overrightarrow{g_{|\mathbf{q}|}}(\mathbf{q}) \mid\mid \overleftarrow{g_1}(\mathbf{q})$. The word embedding function e is implemented in a usual way as a look-up table $\mathbf{V}$. $\mathbf{V}$ is a matrix whose rows can be indexed by words from the vocabulary, that is $e(w) = V_w, w \in V$. Therefore, each row of $\mathbf{V}$ contains embedding of one word from the vocabulary. During training we jointly optimize parameters of f, g and e.

4 Related Work

Several recent deep neural network architectures (Hermann et al., 2015; Hill et al., 2015; Chen et al., 2016; Kobayashi et al., 2016) were applied to the task of text comprehension. The last two architectures were developed independently at the same time as our work. All of these architectures use an attention mechanism that allows them to highlight places in the document that might be relevant to answering the question. We will now briefly describe these architectures and compare

them to our approach.

4.1 Attentive and Impatient Readers

Attentive and *Impatient Readers* were proposed in (Hermann et al., 2015). The simpler Attentive Reader is very similar to our architecture. It also uses bidirectional document and query encoders to compute an attention in a similar way we do. The more complex Impatient Reader computes attention over the document after reading every word of the query. However, empirical evaluation has shown that both models perform almost identically on the CNN and Daily Mail datasets.

The key difference between the Attentive Reader and our model is that the Attentive Reader uses attention to compute a fixed length representation r of the document $\mathbf{d}$ that is equal to a weighted sum of contextual embeddings of words in $\mathbf{d}$, that is $r = \sum_i s_i f_i(\mathbf{d})$. A joint query and document embedding m is then a non-linear function of r and the query embedding $g(\mathbf{q})$. This joint embedding m is in the end compared against all candidate answers $a' \in A$ using the dot product $e(a') \cdot m$, in the end the scores are normalized by softmax. That is: $P(a'|\mathbf{q}, \mathbf{d}) \propto \exp(e(a') \cdot m)$.

In contrast to the Attentive Reader, we select the answer from the context directly using the computed attention rather than using such attention for a weighted sum of the individual representations (see Eq. 2). The motivation for such simplification is the following.

Consider a context "*A UFO was observed above our city in January and again in March.*" and question "*An observer has spotted a UFO in ___ .*"

Since both January and March are equally good candidates, the attention mechanism might put the same attention on both these candidates in the context. The blending mechanism described above would compute a vector between the representations of these two words and propose the closest word as the answer - this may well happen to be February (it is indeed the case for Word2Vec trained on Google News). By contrast, our model would correctly propose January or March.

4.2 Chen et al. 2016

A model presented in (Chen et al., 2016) is inspired by the Attentive Reader. One difference is that the attention weights are computed with a bilinear term instead of simple dot-product, that is: $s_i \propto \exp(f_i(\mathbf{d})^\intercal W\, g(\mathbf{q}))$. The document embedding r is computed using a weighted sum as in the Attentive Reader: $r = \sum_i s_i f_i(\mathbf{d})$. In the end $P(a'|\mathbf{q}, \mathbf{d}) \propto \exp(e'(a') \cdot r)$, where e' is a new embedding function.

Even though it is a simplification of the Attentive Reader this model performs significantly better than the original.

4.3 Memory Networks

MemNNs (Weston et al., 2014) were applied to the task of text comprehension in (Hill et al., 2015).

The best performing memory networks model setup - window memory - uses windows of fixed length (8) centered around the candidate words as memory cells. Due to this limited context window, the model is unable to capture dependencies out of scope of this window. Furthermore, the representation within such window is computed simply as the sum of embeddings of words in that window. By contrast, in our model the representation of each individual word is computed using a recurrent network, which not only allows it to capture context from the entire document but also the embedding computation is much more flexible than a simple sum.

To improve on the initial accuracy, a heuristic approach called *self supervision* is used in (Hill et al., 2015) to help the network to select the right supporting "memories" using an attention mechanism showing similarities to the ours. Plain MemNNs without this heuristic are not competitive on these machine reading tasks. Our model does not need any similar heuristics.

4.4 Dynamic Entity Representation

The Dynamic Entity Representation model (Kobayashi et al., 2016) has a complex architecture also based on the weighted attention mechanism and max pooling over contextual embeddings of vectors for each named entity.

4.5 Pointer Networks

Our model architecture was inspired by Ptr-Nets (Vinyals et al., 2015) in using an attention mechanism to select the answer in the context rather than to blend words from the context into an answer representation. While a Ptr-Net consists of an encoder as well as a decoder, which uses the attention to select the output at each step, our model outputs the answer in a single step. Furthermore,

the pointer networks assume that no input in the sequence appears more than once, which is not the case in our settings.

4.6 Summary

Our model combines the best features of the architectures mentioned above. We use recurrent networks to "read" the document and the query as done in (Hermann et al., 2015; Chen et al., 2016; Kobayashi et al., 2016) and we use attention in a way similar to Ptr-Nets. We also use summation of attention weights in a way similar to MemNNs (Hill et al., 2015).

From a high level perspective we simplify all the discussed text comprehension models by removing all transformations past the attention step. Instead we use the attention directly to compute the answer probability.

5 Evaluation

In this section we evaluate our model on the CNN, Daily Mail and CBT datasets. We show that despite the model's simplicity its ensembles achieve state-of-the-art performance on each of these datasets.

5.1 Training Details

To train the model we used stochastic gradient descent with the ADAM update rule (Kingma and Ba, 2015) and learning rate of 0.001 or 0.0005. During training we minimized the following negative log-likelihood with respect to θ:

$$-logP_\theta(a|\mathbf{q},\mathbf{d}) \tag{3}$$

where a is the correct answer for query $\mathbf{q}$ and document $\mathbf{d}$, and θ represents parameters of the encoder functions f and g and of the word embedding function e. The optimized probability distribution $P(a|\mathbf{q},\mathbf{d})$ is defined in Eq. 2.

The initial weights in the word embedding matrix were drawn randomly uniformly from the interval $[-0.1, 0.1]$. Weights in the GRU networks were initialized by random orthogonal matrices (Saxe et al., 2014) and biases were initialized to zero. We also used a gradient clipping (Pascanu et al., 2012) threshold of 10 and batches of size 32.

During training we randomly shuffled all examples in each epoch. To speedup training, we always pre-fetched 10 batches worth of examples and sorted them according to document length. Hence each batch contained documents of roughly the same length.

For each batch of the CNN and Daily Mail datasets we randomly reshuffled the assignment of named entities to the corresponding word embedding vectors to match the procedure proposed in (Hermann et al., 2015). This guaranteed that word embeddings of named entities were used only as semantically meaningless labels not encoding any intrinsic features of the represented entities. This forced the model to truly deduce the answer from the single context document associated with the question. We also do not use pre-trained word embeddings to make our training procedure comparable to (Hermann et al., 2015).

We did not perform any text pre-processing since the original datasets were already tokenized.

We do not use any regularization since in our experience it leads to longer training times of single models, however, performance of a model ensemble is usually the same. This way we can train the whole ensemble faster when using multiple GPUs for parallel training.

For Additional details about the training procedure see Appendix A.

5.2 Evaluation Method

We evaluated the proposed model both as a single model and using ensemble averaging. Although the model computes attention for every word in the document we restrict the model to select an answer from a list of candidate answers associated with each question-document pair.

For single models we are reporting results for the best model as well as the average of accuracies for the best 20% of models with best performance on validation data since single models display considerable variation of results due to random weight initialization[5], even for identical hyperparameter values. Single model performance may consequently prove difficult to reproduce.

What concerns ensembles, we used simple averaging of the answer probabilities predicted by ensemble members. For ensembling we used 14, 16, 84 and 53 models for CNN, Daily Mail and CBT CN and NE respectively. The ensemble models were chosen either as the top 70% of all trained models, we call this *avg ensemble*. Alternatively we use the following algorithm: We started with

[5]The standard deviation for models with the same hyperparameters was between 0.6-2.5% in absolute test accuracy.

	CNN		Daily Mail	
	valid	test	valid	test
Attentive Reader †	61.6	63.0	70.5	69.0
Impatient Reader †	61.8	63.8	69.0	68.0
MemNNs (single model) ‡	63.4	66.8	NA	NA
MemNNs (ensemble) ‡	66.2	69.4	NA	NA
Dynamic Entity Repres. (max-pool) ♯	71.2	70.7	NA	NA
Dynamic Entity Repres. (max-pool + byway)♯	70.8	72.0	NA	NA
Dynamic Entity Repres. + w2v ♯	71.3	72.9	NA	NA
Chen et al. (2016) (single model)	72.4	72.4	76.9	75.8
AS Reader (single model)	68.6	69.5	75.0	73.9
AS Reader (avg for top 20%)	68.4	69.9	74.5	73.5
AS Reader (avg ensemble)	73.9	**75.4**	78.1	77.1
AS Reader (greedy ensemble)	74.5	74.8	78.7	**77.7**

Table 2: Results of our AS Reader on the CNN and Daily Mail datasets. Results for models marked with † are taken from (Hermann et al., 2015), results of models marked with ‡ are taken from (Hill et al., 2015) and results marked with ♯ are taken from (Kobayashi et al., 2016). Performance of ‡ and ♯ models was evaluated only on CNN dataset.

	Named entity		Common noun	
	valid	test	valid	test
Humans (query) (*)	NA	52.0	NA	64.4
Humans (context+query) (*)	NA	***81.6***	NA	***81.6***
LSTMs (context+query) ‡	51.2	41.8	62.6	56.0
MemNNs (window memory + self-sup.) ‡	70.4	66.6	64.2	63.0
AS Reader (single model)	73.8	68.6	68.8	63.4
AS Reader (avg for top 20%)	73.3	68.4	67.7	63.2
AS Reader (avg ensemble)	74.5	70.6	71.1	**68.9**
AS Reader (greedy ensemble)	76.2	**71.0**	72.4	67.5

Table 3: Results of our AS Reader on the CBT datasets. Results marked with ‡ are taken from (Hill et al., 2015). (*)Human results were collected on 10% of the test set.

the best performing model according to validation performance. Then in each step we tried adding the best performing model that had not been previously tried. We kept it in the ensemble if it did improve its validation performance and discarded it otherwise. This way we gradually tried each model once. We call the resulting model a *greedy ensemble*.

5.3 Results

Performance of our models on the CNN and Daily Mail datasets is summarized in Table 2, Table 3 shows results on the CBT dataset. The tables also list performance of other published models that were evaluated on these datasets. Ensembles of our models set the new state-of-the-art results on all evaluated datasets.

Table 4 then measures accuracy as the proportion of test cases where the ground truth was among the top k answers proposed by the greedy ensemble model for $k = 1, 2, 5$.

CNN and Daily Mail. The CNN dataset is the most widely used dataset for evaluation of text comprehension systems published so far. Perfor-

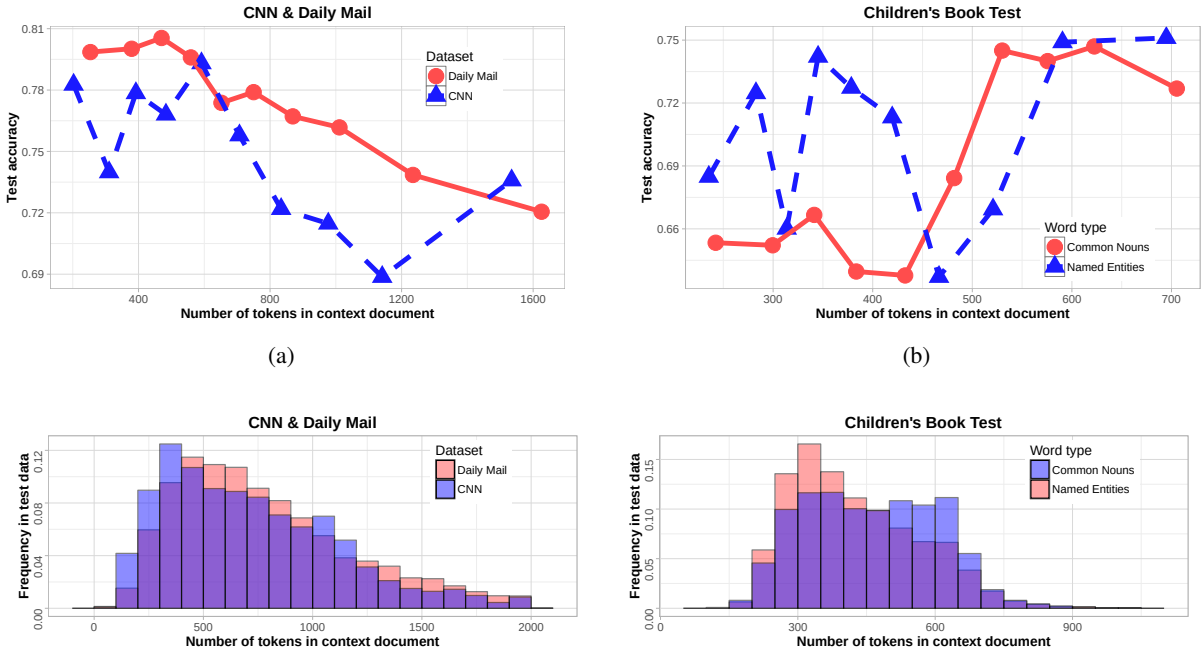


Figure 5: Sub-figures (a) and (b) plot the test accuracy against the length of the context document. The examples were split into ten buckets of equal size by their context length. Averages for each bucket are plotted on each axis. Sub-figures (c) and (d) show distributions of context lengths in the four datasets.

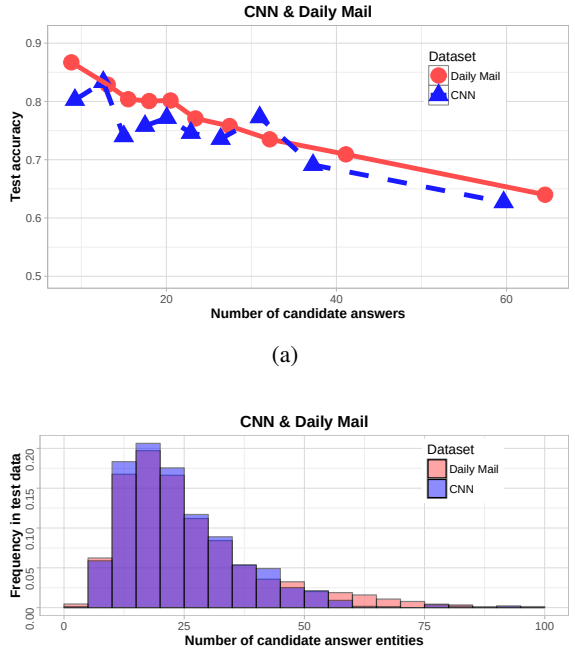


Figure 6: Subfigure (a) illustrates how the model accuracy decreases with an increasing number of candidate named entities. Subfigure (b) shows the overall distribution of the number of candidate answers in the news datasets.

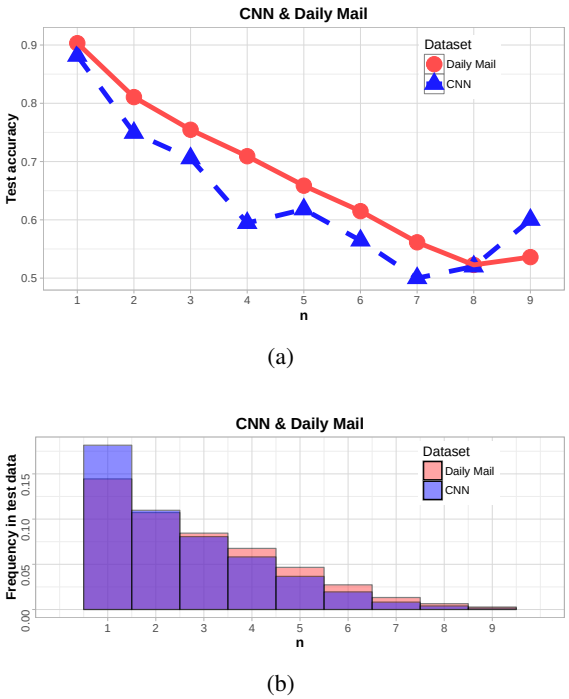


Figure 7: Subfigure (a) shows the model accuracy when the correct answer is the n^{th} most frequent named entity for $n \in [1, 10]$. Subfigure (b) shows the number of test examples for which the correct answer was the n–th most frequent entity. The plots for CBT look almost identical (see Appendix B).

mance of our single model is a little bit worse than performance of simultaneously published models (Chen et al., 2016; Kobayashi et al., 2016). Compared to our work these models were trained with Dropout regularization (Srivastava et al., 2014) which might improve single model performance. However, ensemble of our models outperforms these models even though they use pre-trained word embeddings.

On the CNN dataset our single model with best validation accuracy achieves a test accuracy of 69.5%. The average performance of the top 20% models according to validation accuracy is 69.9% which is even 0.5% better than the single best-validation model. This shows that there were many models that performed better on test set than the best-validation model. Fusing multiple models then gives a significant further increase in accuracy on both CNN and Daily Mail datasets..

CBT. In named entity prediction our best single model with accuracy of 68.6% performs 2% absolute better than the MemNN with self supervision, the averaging ensemble performs 4% absolute better than the best previous result. In common noun prediction our single models is 0.4% absolute better than MemNN however the ensemble improves the performance to 69% which is 6% absolute better than MemNN.

Dataset	$k = 1$	$k = 2$	$k = 5$
CNN	74.8	85.5	94.8
Daily Mail	77.7	87.6	94.8
CBT NE	71.0	86.9	96.8
CBT CN	67.5	82.5	95.4

Table 4: Proportion of test examples for which the top k answers proposed by the greedy ensemble included the correct answer.

6 Analysis

To further analyze the properties of our model, we examined the dependence of accuracy on the length of the context document (Figure 5), the number of candidate answers (Figure 6) and the frequency of the correct answer in the context (Figure 7).

On the CNN and Daily Mail datasets, the accuracy decreases with increasing document length (Figure 5a). We hypothesize this may be due to multiple factors. Firstly long documents may make the task more complex. Secondly such cases are quite rare in the training data (Figure 5b) which motivates the model to specialize on shorter contexts. Finally the context length is correlated with the number of named entities, i.e. the number of possible answers which is itself negatively correlated with accuracy (see Figure 6).

On the CBT dataset this negative trend seems to disappear (Fig. 5c). This supports the later two explanations since the distribution of document lengths is somewhat more uniform (Figure 5d) and the number of candidate answers is constant (10) for all examples in this dataset.

The effect of increasing number of candidate answers on the model's accuracy can be seen in Figure 6a. We can clearly see that as the number of candidate answers increases, the accuracy drops. On the other hand, the amount of examples with large number of candidate answers is quite small (Figure 6b).

Finally, since the summation of attention in our model inherently favours frequently occurring tokens, we also visualize how the accuracy depends on the frequency of the correct answer in the document. Figure 7a shows that the accuracy significantly drops as the correct answer gets less and less frequent in the document compared to other candidate answers. On the other hand, the correct answer is likely to occur frequently (Fig. 7a).

7 Conclusion

In this article we presented a new neural network architecture for natural language text comprehension. While our model is simpler than previously published models, it gives a new state-of-the-art accuracy on all the evaluated datasets.

An analysis by (Chen et al., 2016) suggests that on CNN and Daily Mail datasets a significant proportion of questions is ambiguous or too difficult to answer even for humans (partly due to entity anonymization) so the ensemble of our models may be very near to the maximal accuracy achievable on these datasets.

Acknowledgments

We would also like to thank Tim Klinger for providing us with masked softmax code that we used in our implementation.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2015. Neural Machine Translation by Jointly Learning to Align and Translate. *International Conference on Learning Representations*.

Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, James Bergstra, Ian J. Goodfellow, Arnaud Bergeron, Nicolas Bouchard, and Yoshua Bengio. 2012. Theano: new features and speed improvements. Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A Thorough Examination of the CNN / Daily Mail Reading Comprehension Task. In *Association for Computational Linguistics (ACL)*.

Kyunghyun Cho, Bart van Merrienboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation. *Empirical Methods in Natural Language Processing (EMNLP)*.

Junyoung Chung, Caglar Gulcehre, Kyunghyun Cho, and Yoshua Bengio. 2014. Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling. *arXiv*, pages 1–9.

David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya a. Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John Prager, Nico Schlaefer, and Chris Welty. 2010. Building Watson: An Overview of the DeepQA Project. *AI Magazine*, 31(3):59–79.

Karl Moritz Hermann, Tomas Kocisky, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, and Phil Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems*, pages 1684–1692.

Felix Hill, Antoine Bordes, Sumit Chopra, and Jason Weston. 2015. The goldilocks principle: Reading children's books with explicit memory representations. *arXiv preprint arXiv:1511.02301*.

Diederik P. Kingma and Jimmy Lei Ba. 2015. Adam: a Method for Stochastic Optimization. *International Conference on Learning Representations*, pages 1–13.

Sosuke Kobayashi, Ran Tian, Naoaki Okazaki, and Kentaro Inui. 2016. Dynamic Entity Representation with Max-pooling Improves Machine Reading. *Proceedings of the North American Chapter of the Association for Computational Linguistics and Human Language Technologies (NAACL-HLT)*.

Karthik Narasimhan and Regina Barzilay. 2015. Machine Comprehension with Discourse Relations. *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1253–1262.

Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2012. On the difficulty of training recurrent neural networks. *Proceedings of The 30th International Conference on Machine Learning*, pages 1310–1318.

Matthew Richardson, Christopher J C Burges, and Erin Renshaw. 2013. MCTest: A Challenge Dataset for the Open-Domain Machine Comprehension of Text. *Empirical Methods in Natural Language Processing (EMNLP)*, pages 193–203.

Mrinmaya Sachan, Avinava Dubey, Eric P Xing, and Matthew Richardson. 2015. Learning Answer-Entailing Structures for Machine Comprehension. *Association for Computational Linguistics (ACL)*, pages 239–249.

Andrew M Saxe, James L Mcclelland, and Surya Ganguli. 2014. Exact solutions to the nonlinear dynamics of learning in deep linear neural networks. *International Conference on Learning Representations*.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: prevent NN from overfitting. *Journal of Machine Learning Research*, 15:1929–1958.

Wilson L Taylor. 1953. Cloze procedure: a new tool for measuring readability. *Journalism and Mass Communication Quarterly*, 30(4):415.

Bart van Merrienboer, Dzmitry Bahdanau, Vincent Dumoulin, Dmitriy Serdyuk, David Warde-farley, Jan Chorowski, and Yoshua Bengio. 2015. Blocks and Fuel : Frameworks for deep learning. pages 1–5.

Oriol Vinyals, Meire Fortunato, and Navdeep Jaitly. 2015. Pointer networks. In *Advances in Neural Information Processing Systems*, pages 2674–2682.

Jason Weston, Sumit Chopra, and Antoine Bordes. 2014. Memory networks. *arXiv preprint arXiv:1410.3916*.

Appendix A Training Details

During training we evaluated the model performance after each epoch and stopped the training when the error on the validation set started increasing.

The models usually converged after two epochs of training. Time needed to complete a single epoch of training on each dataset on an Nvidia K40 GPU is shown in Table 5.

Dataset	Time per epoch
CNN	10h 22min
Daily Mail	25h 42min
CBT Named Entity	1h 5min
CBT Common Noun	0h 56min

Table 5: Average duration of one epoch of training on the four datasets.

The hyperparameters, namely the recurrent hidden layer dimension and the source embedding dimension, were chosen by grid search. We started with a range of 128 to 384 for both parameters and subsequently kept increasing the upper bound by 128 until we started observing a consistent decrease in validation accuracy. The region of the parameter space that we explored together with the parameters of the model with best validation accuracy are summarized in Table 6.

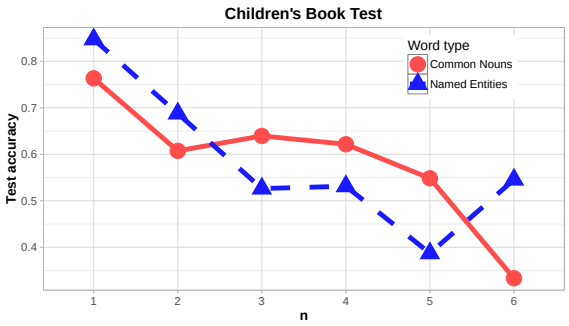
	Rec. Hid. Layer			Embedding		
Dataset	min	max	best	min	max	best
CNN	128	512	384	128	512	128
Daily Mail	128	1024	512	128	512	384
CBT NE	128	512	384	128	512	384
CBT CN	128	1536	256	128	512	384

Table 6: Dimension of the recurrent hidden layer and of the source embedding for the best model and the range of values that we tested. We report number of hidden units of the unidirectional GRU; the bidirectional GRU has twice as many hidden units.

Our model was implemented using Theano (Bastien et al., 2012) and Blocks (van Merrienboer et al., 2015).

Appendix B Dependence of accuracy on the frequency of the correct answer

In Section 6 we analysed how the test accuracy depends on how frequent the correct answer is compared to other answer candidates for the news datasets. The plots for the Children's Book Test looks very similar, however we are adding it here for completeness.



(a)

(b)

Figure 8: Subfigure (a) shows the model accuracy when the correct answer is among n most frequent named entities for $n \in [1, 10]$. Subfigure (b) shows the number of test examples for which the correct answer was the n–th most frequent entity.