Gated-Attention Readers for Text Comprehension

Bhuwan Dhingra[*] **Hanxiao Liu**[*] **William W. Cohen** **Ruslan Salakhutdinov**
School of Computer Science
Carnegie Mellon University
{bdhingra, hanxiaol, wcohen, rsalakhu}@cs.cmu.edu

Abstract

In this paper we study the problem of answering cloze-style questions over short documents. We introduce a new attention mechanism which uses multiplicative interactions between the query embedding and intermediate states of a recurrent neural network reader. This enables the reader to build query-specific representations of tokens in the document which are further used for answer selection. Our model, the Gated-Attention Reader, outperforms all state-of-the-art models on several large-scale benchmark datasets for this task—the CNN & Daily Mail news stories and Children's Book Test. We also provide a detailed analysis of the performance of our model and several baselines over a subset of questions manually annotated with certain linguistic features. The analysis sheds light on the strengths and weaknesses of several existing models.

1 Introduction

A recent trend to measure progress towards machine reading is to test a system's ability of answering questions over the text it has to comprehend. Towards this end, several large-scale datasets of cloze-style questions over a context document have been introduced recently which allow the training of supervised machine learning systems (Hermann et al., 2015; Hill et al., 2015). The relatively unambiguous nature of answers to cloze-questions provides an objective benchmark to measure a system's performance at text comprehension.

[*]BD and HL contributed equally to this work.

Deep learning methods have been recently shown to outperform traditional shallow approaches for this task (Hermann et al., 2015). Their performance is further boosted by attention mechanisms borrowed from the machine translation literature (Bahdanau et al., 2014). This is not surprising, as answering a cloze-style query requires one to only attend to parts of the document relevant to the query. Current attention models aim to mimic this by weighting representations of document sub-units (such as words or sentences) by a score which determines their relevance to the query (Hermann et al., 2015; Hill et al., 2015; Kadlec et al., 2016).

An alternative way to selectively "attend" to different parts of the document while reading it is by gating the input to the reader with the query representation. This can be intuitively viewed as masking unimportant features of the input (for the task at hand) and emphasizing the important ones. Specifically, we compute an element-wise product between the query embedding and document representations at the intermediate layers of a Bidirectional Gated Recurrent Unit (GRU) (Cho et al., 2014) reader. Our method was inspired by the observation that the GRU itself provides an attention mechanism through the use of its reset and forget gates; the element-wise product simply helps it focus on the query. We call this model the Gated-Attention (GA) Reader. The effectiveness of the proposed gated-attention mechanism is verified by the consistent improvement of the GA reader over various existing models on four different benchmark datasets.

Besides achieving strong performance, we believe an equally important metric is being able to interpret such performance. We thus quantitatively investigate the performance of several representative models over a subset of questions with annotated linguistic features. This detailed analysis further justifies the advantages of the GA Reader by showing its superior performance in handling questions of diverse linguistic nature.

The paper is organized as follows: We introduce existing models (and their relationship to our model) in Section 2 and then the GA Reader in Section 3. Section 4 presents results of empirical evaluations, followed by a detailed analysis in Section 5. We summarize our findings in Section 6.

2 Related Work

The cloze-style QA task involves tuples of the form (d, q, a), where d is a document, q is a cloze-style question over the contents of that document and a is the answer to this query. The answer comes from a fixed vocabulary A, which, depending on the dataset, may consist of all words in the vocabulary or a list of candidates from the current document. The task can then be described as: given a document-query pair (d, q) find $a \in A$ which answers q.

In the following we provide an overview of existing neural architectures which have been applied to this problem with encouraging results.

2.1 LSTMs with Attention

Several architectures introduced in (Hermann et al., 2015) employ LSTM units to compute a combined document + query representation $g(d, q)$. Given this representation and a lookup-table of embeddings of possible answers W (which is shared with the lookup-table used by the LSTM), the probability of an answer is computed as:

$$p(a|q) \propto \exp(W(a)g(d, q)), \quad a \in V. \tag{1}$$

The motivation behind (1) is that $g(d, q)$ should only contain information relevant to the task at hand—that of getting the answer a and should be close in the vector-space to the correct answer, hence maximizing the inner product between these representations. Training can be performed using standard SGD over a loss-function such as cross-entropy between true answer and its predicted probability.

With this formulation, there are three related architectures proposed in (Hermann et al., 2015). The **DeepLSTM Reader** performs a single forward pass through the concatenated *(document,query)* pair to compute $g(d, q)$. It further has two possible modes: *cqa* where the document is placed before the query, and *qca* where the query is placed before the document. The **Attentive Reader** computes a document vector r and query vector q separately and then combines them through an additional feed-forward layer to give $g(d, q)$. The document vector r is computed using a weighted average of the intermediate token representations learned by a bi-directional LSTM, where the weights are computed conditioned on the query vector q. This is the conventional attention mechanism which favors certain tokens over others based on the query. The **Impatient Reader** is similar to the Attentive reader, but computes the document vector r iteratively such that the weight distribution over tokens in each iteration is conditioned on a particular query token (or its representation from the corresponding bi-directional LSTM).

The Attentive and Impatient Readers give similar performance on the CNN and Daily Mail datasets, and both outperform the DeepLSTM Reader (cf. (Hermann et al., 2015)). In our model we also use recurrent neural networks to read the document and query, but use a different attention mechanism and restrict candidate answers to the tokens appearing in the document.

2.2 Memory Networks

The Memory Network architecture was proposed in (Weston et al., 2014) and applied to the datasets considered here in (Hill et al., 2015). In MemNets, each sentence s in the input document is encoded to a memory m. This encoding uses a simple averaging of the vectors assigned to a window within the sentence around candidate answers. In line with conventional attention, the relevance of a particular memory m_i to the query is computed by taking its dot-product with the query vector q and passing it through a softmax layer to get a distribution over all memories. Let this be α_i. The overall memory in this first pass, relevant to the query, is then $m_{o1} = \sum_i \alpha_i m_i$. This process is repeated for several iterations, and each time the input query is $q_k = q_{k-1} + m_{ok-1}$, with $q_0 = q$.

The intuition behind this computation is that multiple hops over the same set of memories allow the model to reason over them individually. In each hop, the model can select a relevant memory and add it to the input for the next hop. Indeed, experiments on a different dataset from (Weston et al., 2014) show the effectiveness of the model to reason over documents. Combined with a set of heuristics, such as intermediate supervision and model ensembling, this architecture outperforms all the models discussed in section 2.1.

In this work, we retain the key idea of using multiple layers in the reader to effectively allow reasoning over the document, but employ recurrent neural networks for encoding memories (in our case a document) and the query.

2.3 Attention Sum Reader

The AS reader, presented in (Kadlec et al., 2016), obtains the current state-of-the-art performance on the datasets under consideration here. This is a simplified version of the DeepLSTM Reader which uses a bi-directional GRU network (Cho et al., 2014) to encode the query and document each into vectors. Then, candidate answers which appear in the document are evaluated by extracting their contextual representations from the bi-GRU and taking a dot-product with the query vector q. This gives a score for each possible answer, which is normalized by the softmax function. Finally, multiple mentions of the same candidate entity are combined by adding up their probabilities, and the token with the maximum aggregated probability is selected as the predicted answer. This aggregation scheme is known as the *pointer sum attention*.

An important consequence of summing multiple mentions of an answer is that this model favors entities which occur frequently in the input document. However, as discussed in (Hermann et al., 2015), correct answers often appear frequently in the document for the news datasets in consideration here, so this generally improves model accuracy. The authors also present results on an ensemble of models with the same architecture which does significantly better than any single model.

The AS reader is a special case of our model with only a single layer and without the *gated-attention* mechanism described in detail below. These additions make the GA reader more expressive, and are justified by its superior performance.

2.4 Dynamic Entity Representations

The Dynamic Entity Representation (DER) network was recently introduced by (Kobayashi et al., 2016). This model builds dynamic representations of the candidate answers (entities in their case) while reading the document, by using a max-pooling function to accumulate the information collected on an entity so far. The model is motivated by the anonymization of entities in the CNN and Daily Mail datasets, and shows improved performance over previous approaches on the CNN dataset. Results on the Daily Mail and CBT datasets are not presented.

In our model we do not accumulate information over candidate answers while reading. Instead, we dynamically gate their representations between layers to "mask" away less informative features in their representation and retain the important ones.

3 Gated-Attention Reader

3.1 Motivation

Our method extends the Attention Sum (AS) Reader, and performs multiple hops over the input document similar to the Memory Networks architecture (Sukhbaatar et al., 2015). We use the same *pointer sum attention* mechanism as the AS reader in the output layer to obtain a distribution over candidate answers. End-to-end memory networks have been shown to perform well on synthetic QA tasks which require reasoning over multiple input sentences, and a key reason for this is its multi-layered architecture which performs several passes over the context (Sukhbaatar et al., 2015). Based on these findings we also incorporate multiple layers in our model.

Our key contribution, however, is in a *gated-attention mechanism*, where after the first layer, subsequent hops over the document are gated by the query representation using an element-wise combination at layer *inputs* (Figure 1). This is in contrast to conventional attention mechanisms which apply a weighted averaging of layer *outputs* with the weights determined by a dot-product between query and document, e.g., attention mechanisms in all architectures discussed in section 2.

Compositional operators for merging vector space representations of text have been studied extensively in the literature without a clear consensus on the best one. Depending on the task at hand, in some

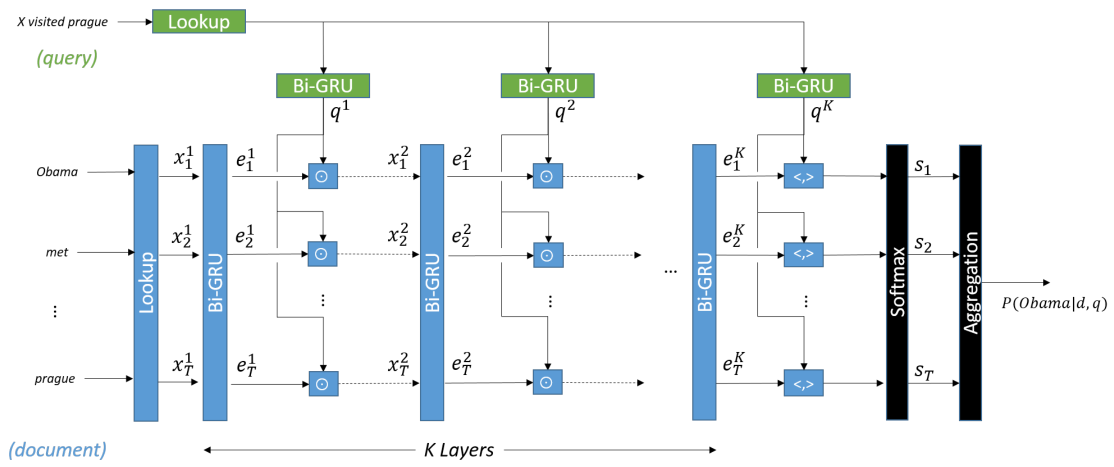


Figure 1: Gated-Attention Reader. Dashed lines represent dropout connections. See section 3.2 for detailed descriptions.

cases addition seems to perform better (for example when composing phrase representations (Mitchell and Lapata, 2008)), in others multiplication does better (for example when modeling entity relations (Yang et al., 2014)). In our experiments, we explored both addition and multiplication as well as concatenation for the element-wise gating operation discussed above, and multiplication performed best. Previously, (Kiros et al., 2014) also used multiplicative interactions between text attributes (such as author information) and word embeddings to derive context-dependent text representations. For the GA reader, the query representation can also be viewed as an attribute conditioned on which we want to learn the relevant document representation.

3.2 Model Details

Figure 1 illustrates the Gated-Attention (GA) reader. Each box labeled *Bi-GRU* denotes a bidirectional Gated Recurrent Unit network (Cho et al., 2014). The GA reader reads input documents and queries over K horizontal layers. At each layer, we combine the final forward and backward GRU hidden states to get the layer-specific query vector:

$$q^i = h^f_{|Q|}(i) || h^b_0(i), \qquad i = 1, 2, ...K, \tag{2}$$

where $||$ stands for concatenation, $|Q|$ is the query length, and $h^f_t(i)$ and $h^b_t(i)$ are forward and backward GRU hidden states at time t in layer i, respectively. GRUs which encode the query are shown in green in the figure.

For the document, input at the first Bi-GRU layer consists of word embeddings for the words in the document $x^1_t = L(w_t)$, where $L \in \mathrm{R}^{|V| \times d_L}$ is the word-lookup table, and w_t are indices into the word vocabulary. To compute the input to subsequent Bi-GRU layers, we use the *Gated-Attention* mechanism by applying an element-wise multiplication between the query embedding q^{i-1} and the outputs e^{i-1}_t from the previous layer:

$$x^i_t = q^{i-1} \odot e^{i-1}_t, \quad t = 1, 2, ...|d|. \quad i = 2, ...K. \tag{3}$$

Here $|d|$ is the document length and $\odot$ represents element-wise multiplication (a.k.a. Hadamard product). Layer outputs e^i_t are the contextual embeddings of document tokens formed by concatenating intermediate forward and backward GRU hidden states:

$$e^i_t = h^f_t(i) || h^b_t(i), \quad t = 1, 2, ...|d|. \quad i = 1, ...K. \tag{4}$$

GRUs encoding the document are shown in blue in Figure 1. To obtain the probability that a particular token in the document answers the query we take an inner-product between outputs of the last layer q^K and e^K_t, and pass through a soft-max layer:

$$s_t = \frac{\exp\{\langle q^K, e^K_t \rangle\}}{\sum_{t'} \exp\{\langle q^K, e^K_{t'} \rangle\}}, \quad t = 1, 2, ...|d|, \tag{5}$$

where $\langle , \rangle$ denotes dot-product between two vectors. Finally, similar to the AS reader, we aggregate the probabilities for tokens which appear multiple times

Table 1: Dataset statistics.

Dataset	#Train	#Val	#Test	#Vocab
CNN	380,298	3,924	3,198	118,497
Daily Mail	879,450	64,835	53,182	208,045
CBT-NE	108,719	2,000	2,500	53,063
CBT-CN	120,769	2,000	2,500	53,185

in a document before selecting the maximum as the predicted answer (here $I(a, d)$ is the set of positions where token a appears in the document d):

$$P(a|d, q) = \sum_{i \in I(a,d)} s_i \tag{6}$$

$$a^* = \text{argmax}_{a \in A} \; P(a|d, q). \tag{7}$$

We then use cross-entropy loss between the predicted probabilities and true answers for training. In the special case where $K = 1$, the above GA reader reduces to the Attention-Sum Reader.

4 Experiments and Results

4.1 Datasets

We evaluate the GA reader on four datasets. The first two, CNN and Daily Mail News Stories[1], come from (Hermann et al., 2015) and consist of News articles from the popular CNN and Daily Mail websites. A query over each article was formed by removing an entity from the short summary which follows the article. Further, entities within each article were anonymized to make the task purely a comprehension one. N-gram statistics, for instance, computed over the entire corpus are no longer useful in such an anonymized corpus.

The other two datasets are formed from two different subsets of the Children's Book Test (CBT)[2] (Hill et al., 2015). Documents consist of 20 contiguous sentences from the body of a popular children's book, and the query is formed by deleting a token from the 21st sentence. We only focus on subsets where the deleted token is either a common noun (CN) or named entity (NE) since simple language models already give human-level performance on the other types (cf. (Hill et al., 2015)).

[1] https://github.com/deepmind/rc-data

[2] http://www.thespermwhale.com/jaseweston/babi/CBTest.tgz

Table 2: Optimum hyperparameters for each dataset.

Dataset	K	d	p
CNN	3	256	0.2
Daily Mail	2	256	0.2
CBT-NE	2	128	0.4
CBT-CN	2	128	0.4

Statistics of the four datasets used in our experiments are summarized in Table 1.

4.2 Implementation Details

Our model was implemented using the Theano[3] and Lasagne[4] Python libraries. We used Stochastic Gradient Descent with ADAM updates for training, which combines classical momentum and adaptive gradients (Kingma and Ba, 2014). The learning rate was set to 0.0005 and the batch size at each iteration to 32 for all models & datasets. We also used gradient clipping with a threshold of 10 to stabilize the GRU training (Pascanu et al., 2012). Word embeddings are of size 128 for all models & datasets, initialized with vectors obtained by running the `word2vec` toolkit[5] on all documents and queries in the training set. We empirically found that this approach led to faster convergence. All other parameters were initialized to their default values as specified in the Lasagne library.

Three parameters were tuned on the validation set for each dataset—the number of layers K, GRU hidden state sizes (both query and document) d, and the dropout rate p. We experimented with $K = \{2, 3\}$, $d = \{256, 384\}$ and $p = \{0.1, 0.2, 0.3, 0.4, 0.5\}$. Memory constraints prevented us from experimenting with higher K. Optimal values for each dataset are shown in Table 2, though it should be noted that several different settings give similar performance.

4.3 Performance Comparison

Table 3 shows the performance of all methods discussed above on CNN, Daily Mail and CBT datasets. Following the setting of previous works, we compare both the single best models and their ensembles. For the GA reader the best M trained

[3] http://deeplearning.net/software/theano/

[4] https://lasagne.readthedocs.io/en/latest/

[5] https://code.google.com/archive/p/word2vec/

Table 3: Validation/Test accuracy ($\%$) of all models on CNN, Daily Mail and CBT datasets. Results marked with † are cf previously published works. Best single/ensemble models are in red/blue.

Model	**CNN**		**Daily Mail**		**CBT-NE**		**CBT-CN**	
	Val	Test	Val	Test	Val	Test	Val	Test
Humans (query) †	–	–	–	–	–	52.0	–	64.4
Humans (context + query) †	–	–	–	–	–	81.6	–	81.6
LSTMs (context + query) †	–	–	–	–	51.2	41.8	62.6	56.0
Deep LSTM Reader †	55.0	57.0	63.3	62.2	–	–	–	–
Attentive Reader †	61.6	63.0	70.5	69.0	–	–	–	–
Impatient Reader †	61.8	63.8	69.0	68.0	–	–	–	–
MemNets (single model) †	63.4	66.8	–	–	70.4	66.6	64.2	63.0
MemNets (ensemble) †	66.2	69.4	–	–	–	–	–	–
AS Reader (single model) †	68.6	69.5	75.0	73.9	73.8	68.6	68.8	63.4
AS Reader (ensemble) †	73.9	75.4	78.7	77.7	76.2	71.0	71.1	68.9
DER Network †	71.3	72.9	–	–	–	–	–	–
GA Reader (single model)	73.0	**73.8**	76.7	**75.7**	74.9	**69.0**	69.0	**63.9**
GA Reader (ensemble)	76.4	**77.4**	79.1	**78.1**	75.5	**71.9**	72.1	**69.4**

models on each dataset were selected as ensemble members, where M was tuned on the validation set.

The single best GA readers outperform all previously published single models on all four datasets. Meanwhile, combining together multiple GA readers gives the new state-of-the-art ensemble results on all four datasets. Improvement for the single models is greater for Daily Mail and CNN datasets as compared to the CBT datasets, for which the GA reader was prone to overfitting. The CBT datasets are considerably smaller than the other two, thus this is not surprising given the higher expressive power of GA reader. Improvement for the ensembles is significant for the CNN and CBT-NE datasets, but mild for Daily Mail and CBT-CN datasets.

4.4 Visualizing Attention

Figure 2 presents two example questions from the CNN test set with an overlaid heat-map showing the attention at the output layer of the GA reader. The attention is computed as the value of s_t in equation (5) for each token in the document, and is only visualized when $s_t > 0.05$. Both questions require the system to understand paraphrases as well as reason over multiple evidence sentences to arrive at the correct answer. The GA reader manages to attend to the right tokens in both cases.

5 Detailed Analysis

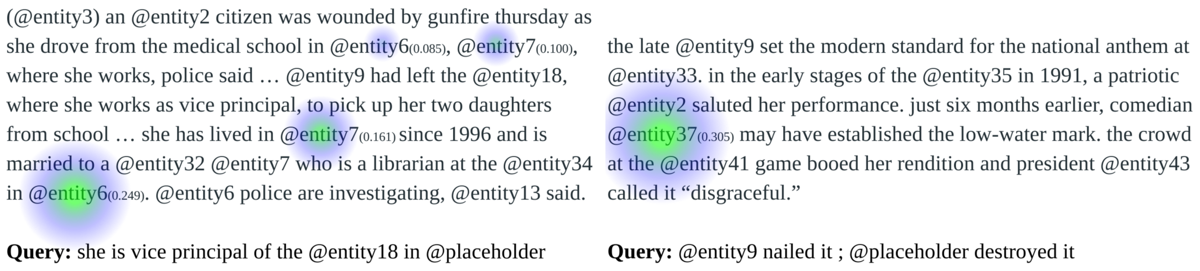
5.1 Linguistic Feature Annotation

To gain insights about the neural readers' behavior at a more detailed level, we randomly sampled 100 questions from the test set of CNN and manually annotated their linguistic features[6]. For each question, features that may potentially correlate with the neural readers' performance were extracted, as outlined in Table 4. In this paper, we primarily focus on two types of linguistic features:

- *Plain Features* that are automatically annotated by scripts, such as document/query length, answer frequency, etc. Most of them are straightforward meta-information about the question.

- *Semantic Features* that are manually annotated by a human expert. For example, whether logical reasoning is necessary to obtain the correct answer. Features of this type are arguably more

[6]We plan to make the annotated questions publicly available if the manuscript is accepted for publication.



(@entity3) an @entity2 citizen was wounded by gunfire thursday as she drove from the medical school in @entity6(0.085), @entity7(0.100), where she works, police said … @entity9 had left the @entity18, where she works as vice principal, to pick up her two daughters from school … she has lived in @entity7(0.161) since 1996 and is married to a @entity32 @entity7 who is a librarian at the @entity34 in @entity6(0.249). @entity6 police are investigating, @entity13 said.

Query: she is vice principal of the @entity18 in @placeholder

(a) GA reader correctly answered @entity6.

the late @entity9 set the modern standard for the national anthem at @entity33. in the early stages of the @entity35 in 1991, a patriotic @entity2 saluted her performance. just six months earlier, comedian @entity37(0.305) may have established the low-water mark. the crowd at the @entity41 game booed her rendition and president @entity43 called it “disgraceful.”

Query: @entity9 nailed it ; @placeholder destroyed it

(b) GA reader correctly answered @entity37.

Figure 2: GA Reader’s final-layer attention (before aggregation) over tokens in the questions.

Plain Features		
Feature Name	Value	Description
Document Length (DL)	$\mathbb{R}_+$	Total number of words in the context in logarithmic scale, namely $\log(\lvert d\rvert)$.
Query Length (QL)	$\mathbb{N}_+$	Total number of words in the query, namely $\lvert q\rvert$.
Answer Frequency (AF)	$\mathbb{N}_+$	Count of correct answers in d.
Answer First Location (AFL)	$[0, 1]$	Distance from the 1st answer occurrence to the beginning of d, normalized by $\lvert d\rvert$.
Answer Last Location (ALL)	$[0, 1]$	Distance from the last answer occurrence to the end of d, normalized by $\lvert d\rvert$.
Semantic Features		
Feature Name	Value	Description
Evidence Sentence (ES)	$\mathbb{N}_+$	Number of evidence sentences in d needed in order to correctly answer the query.
N-gram (NG)	$\{0, 1\}$	If there is an N-gram in d that overlaps with the correct answer and its context in q.
Paraphrase (P)	$\{0, 1\}$	Whether the query is rephrased in the document.
Logical Reasoning (LR)	$\{0, 1\}$	If multiple independent facts need to be logically combined to obtain the answer.
Temporal Reasoning (TR)	$\{0, 1\}$	Special case of Logical Reasoning, where the temporal order of events matters.

Table 4: Annotated linguistic features for the 100 CNN questions.

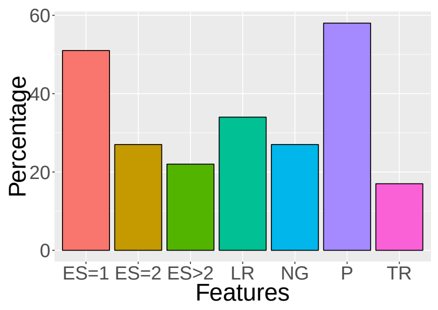


Figure 3: Corpus statistics of manually annotated semantic features. The y-axis stands for the percentage of questions among the 100 samples conditioned on each feature.

precise indicators about the intrinsic semantic nature of any given question.

Figure 3 summarizes the corpus statistics regarding semantic features. Interestingly, the figure shows that the majority of the questions can be answered based on N-grams or paraphrases with less than two evidence sentences. This suggests relying on accuracy as the sole metric for model evaluation could be misleading, as a high accuracy can be achieved by pure “memorizing” regardless of the model’s capability of capturing real semantics.

5.2 Conditional Performance

In this subsection, we investigate the performance of several representative models conditioned on a subset of questions with certain semantic features. As we discussed above, breaking down overall accuracy into multiple conditional performances will enable us to get a more comprehensive view about the strength and weakness of different architectures.

Four models are chosen for comparison: (1) the word distance benchmark (WD), a simple yet strong baseline relying on word distance measurements; (2) the Deep LSTM Reader, a representative LSTM-based neural architecture (Hermann et al., 2015); (3) the Attention Sum (AS) Reader, which achieves

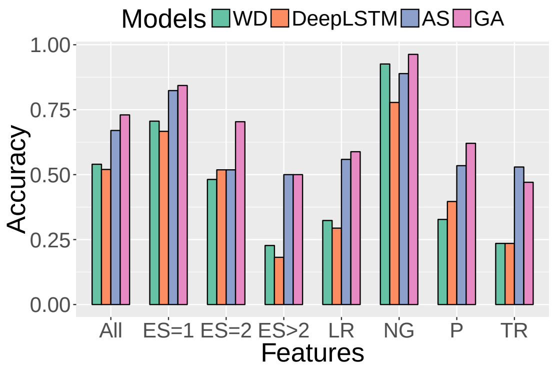


Figure 4: Conditional performance of the word distance model, Deep LSTM, AS and GA readers over the 100 CNN questions with respect to different semantic features. "All" refers to the overall performance on the 100 questions without conditions.

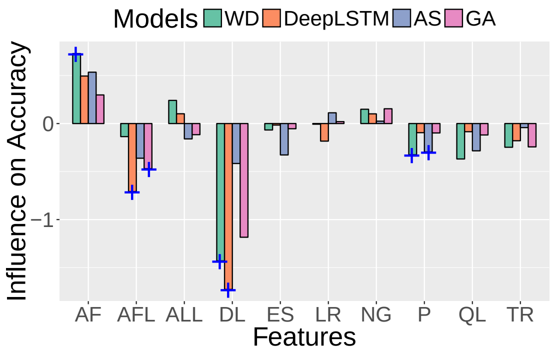


Figure 5: Regression coefficients of various linguistic features with respect to the performance of different models. A positive coefficient indicates a positive influence on the corresponding model's test-set accuracy and vice versa. Coefficients that are statistically significant ($\alpha = 5\%$) are marked with "+".

the state-of-the-art performance over several text comprehension tasks (Kadlec et al., 2016); (4) GA reader, our proposed model.

Results are reported in Figure 4, where the conditional performance of GA reader dominates other baselines except for temporal reasoning, showing a balanced strength in handling questions across the spectrum. While all models show weakness in reasoning with multiple facts ($\texttt{ES} \geq 3$), we notice GA reader wins with a good margin in terms of combining two facts ($\texttt{ES} = 2$), suggesting its advantages in preliminary logical reasoning. The results also empirically justify the merits of attention sum, as both AS and GA readers significantly outperform WD and vanilla deep LSTM over all question types.

5.3 Sensitivity Analysis

With the linguistic features in Table 4, each of the 100 CNN sample questions can be summarized as an 11-dimensional vector x (10 features plus an intercept). Let $y = 1$ if the question has been correctly answered and $y = 0$ otherwise. One can therefore quantitatively characterize the sensitivity of the neural readers' performance w.r.t. different linguistic features by fitting a linear regression model where x and y are treated as explanatory and response variables, respectively [7]. The resulting regression coefficients and p-values can effectively summarize which features are influential to the model performance.

[7]Before regression, all features were normalized to the range of $[0, 1]$ to make their scales comparable.

Figure 5 shows the regression coefficients associated with different linguistic features in Table 4. Besides the known effect of DL and AF that have been discussed in the literature, we see the involvement of logical reasoning (LR), paraphrases (P) and multiple evidence sentences (ES) consistently lead to decreased performance for all models, though multi-layered architectures (DeepLSTM, GA) are empirically more robust than single-layer (AS) or shallow (WD) ones in those aspects. Interestingly, Figure 5 also shows AFL is playing a crucial role. One possible explanation is that questions whose answers appear early tend to be relatively easier in nature. Meanwhile, it is not hard to verify that with high probability AFL will be negatively correlated with AF, a quantity known to have significance influence on the performance of most models.

6 Conclusion

We presented the Gated-Attention reader for answering cloze-style questions over context documents. The model features a novel multiplicative gating mechanism, and further combines the merits of both *point-sum attention* and the multi-layered architectures of Memory Networks. Our model design was justified by its state-of-the-art performance over several large-scale benchmark datasets under both single-best and ensemble settings, and backed up by a detailed performance analysis with respect to questions with various linguistic features.

Acknowledgments

The authors would like to thank Eduard Hovy and Teruko Mitamura for useful discussions and feedback.

References

[Bahdanau et al.2014] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

[Cho et al.2014] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using rnn encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078*.

[Hermann et al.2015] 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.

[Hill et al.2015] 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*.

[Kadlec et al.2016] Rudolf Kadlec, Martin Schmid, Ondrej Bajgar, and Jan Kleindienst. 2016. Text understanding with the attention sum reader network. *arXiv preprint arXiv:1603.01547*.

[Kingma and Ba2014] Diederik Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

[Kiros et al.2014] Ryan Kiros, Richard Zemel, and Ruslan R Salakhutdinov. 2014. A multiplicative model for learning distributed text-based attribute representations. In *Advances in Neural Information Processing Systems*, pages 2348–2356.

[Kobayashi et al.2016] Sosuke Kobayashi, Ran Tian, Naoaki Okazaki, and Kentaro Inui. 2016. Dynamic entity representations with max-pooling improves machine reading. In *NAACL-HLT*.

[Mitchell and Lapata2008] Jeff Mitchell and Mirella Lapata. 2008. Vector-based models of semantic composition. In *ACL*, pages 236–244.

[Pascanu et al.2012] Razvan Pascanu, Tomas Mikolov, and Yoshua Bengio. 2012. On the difficulty of training recurrent neural networks. *arXiv preprint arXiv:1211.5063*.

[Sukhbaatar et al.2015] Sainbayar Sukhbaatar, Jason Weston, Rob Fergus, et al. 2015. End-to-end memory networks. In *Advances in Neural Information Processing Systems*, pages 2431–2439.

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

[Yang et al.2014] Bishan Yang, Wen-tau Yih, Xiaodong He, Jianfeng Gao, and Li Deng. 2014. Learning multi-relational semantics using neural-embedding models. *arXiv preprint arXiv:1411.4072*.