Machine Comprehension Based on Learning to Rank

Tian Tian and **Yuezhang Li**
Carnegie Mellon University
`ttian1, yuezhanl@andrew.cmu.edu`

Abstract

Machine comprehension plays an essential role in NLP and has been widely explored with dataset like MCTest. However, this dataset is too simple and too small for learning true reasoning abilities. (Hermann et al., 2015) therefore release a large scale news article dataset and propose a deep LSTM reader system for machine comprehension. However, the training process is expensive. We therefore try feature-engineered approach with semantics on the new dataset to see how traditional machine learning technique and semantics can help with machine comprehension. Meanwhile, our proposed L2R reader system achieves good performance with efficiency and less training data.

1 Introduction

Machine comprehension, as the central goal in NLP, has been explored with a variety of methods. Based on the released dataset MCTest (Richardson et al., 2013), a lexical matching based method is proposed (Smith et al., 2015). It applies linguistic features to tackle this problem. However, this method dives deep into the characteristics of the dataset, and may not generalize well to other datasets. This problem also goes with the discourse relation model introduced in (Narasimhan and Barzilay, 2015) that explore causal, temporal and explanation relationships between two sentences, which does not scale well. Hence, (Wang and McAllester, 2015) introduced a max-margin learning frame that incorporates a feature set of frame semantics, syntax,

coreference, and word embeddings. The accuracy of 69.94% was achieved in MC500 of MCTest dataset. Meanwhile, (Sachan et al., 2015) applied similar loss function, modeling machine comprehension as textual entailment and solved the problem by constructing latent answer-entailing structure with an accuracy of 67.83%. Although such good accuracy were achieved, the dataset they worked on has some limitations in terms of data size and content. Therefore, our work is based on a much larger news article dataset created by (Hermann et al., 2015).

Different from the MCTest dataset, this news article dataset consists of cloze style questions, i.e. questions generally generated by removing a phrase from a sentence (Taylor, 1953). Since the questions can be formed from a short summary of the document with condensed form of paraphrase, the dataset is suitable for testing machine comprehension (Hermann et al., 2015).

To realize machine comprehension, we develop a learning to rank reader (L2R Reader) system by first exploring features on frequency, word distance, syntax and semantics. Then, through learning to rank, we construct a ranking model that can directly pick the answer from the candidate list of answers. Opposed to the deep LSTM (Hermann et al., 2015) that computes the answer based on context information of documents, it is more efficient and does not require much data to reach good performance. Moreover, we incorporate the semantics into the system to improve comprehension ability.

This article is organized as follows. Section 2 introduces the task and essential parts of relevant

	CNN			Daily Mail		
	train	dev	test	train	valid	test
# 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
word count		118,497			208,045	

Table 1: Dataset Statistics

datasets. Section 3 presents related work and distinguishes our work from them. Section 4 describes our model from aspects of learning to rank algorithms, features and usage of semantic information. Section 5 then evaluates our model from performance, semantics analysis and error analysis. Finally, section 6 summarizes our work and points out contributions.

2 Task and Datasets

This section gives a brief introduction of the task and the dataset recently released for this task.

2.1 Formal Task Description

This task requires answering a cloze style question based on the understanding of a context document provided with the question. Along with each question and document, it also provides the correct answer to the question and a list of candidate answers. Thus, this can be formalized as follows:

The training data consists of tuples (d, q, a, A), where d is a context document for answering the question q, a is the correct answer to question q, A denotes a set of candidate answers to the question and $a \in A$ as defined.

2.2 Datasets

The dataset (Hermann et al., 2015) we used in this task were constructed from news article from CNN and Daily Mail websites. The context document of the dataset is from the main body of the news article while the question is formed from one top sentence summarized the news article. Specifically, the question is constructed by replacing the named entity with a placeholder, e.g. "@placeholder and @entity2 welcome son @entity6" is a question defined in the dataset.

Anonymised Version
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 . " ...
producer **X** will not press charges against *ent212* , his lawyer says .
ent193

Table 2: Example of Anonymised version of a data point

Furthermore, to make context information required for answering question, we use the anonymised version as shown in Figure 2 to eliminate influence of background knowledge. Thus, we must exploit the context to answer the question and these two corpora truly measures the capacity of reading comprehension.

The basic statistics of CNN and Daily Mail of dataset are summarized in Table 1.

3 Related Work

Machine comprehension generally concentrates on MCTest (Richardson et al., 2013) and due to the limitation of data size, the state of the arts are mainly based on traditional machine learning techniques. For example, (Wang and McAllester, 2015) proposed a max-margin learning framework that combines features on syntax, coreference, frame semantics and word embeddings, which achieves significant improvement on the problem of MCTest question answering. Although recently (Trischler et al., 2016) proposed a parallel-hierarchical model based
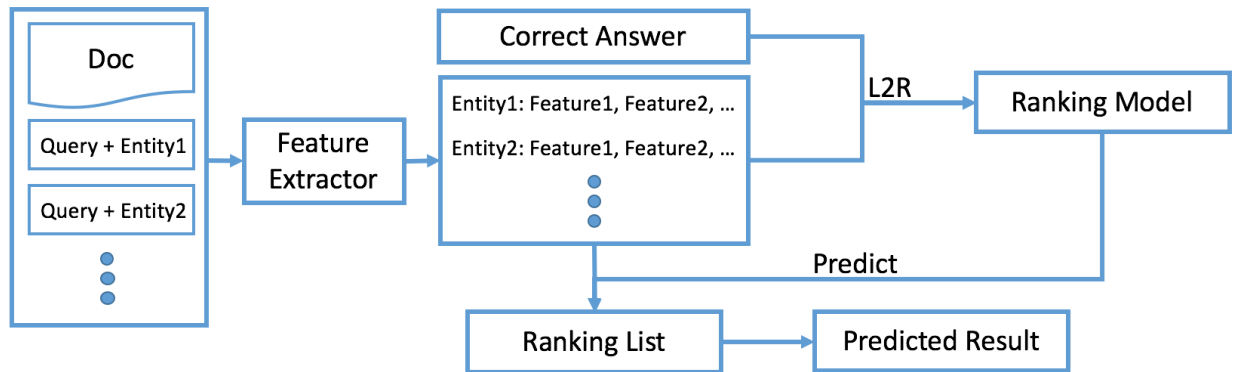
Figure 1: A high-level architecture of the L2R reader system

on neural network and this method outperforms the previous feature-engineered approaches, it has reasoning limitation that the reasoning can only be achieved by stringing important sentences together. Their experiment proves that MCTest is too simple to learn true reasoning and it is also too small for that goal.

Considering the limitations of MCTest dataset, (Hermann et al., 2015) provides a large scale supervised reading comprehension dataset collected from the CNN and Daily Mail websites. This helps with the bottleneck that large dataset is missing on machine comprehension evaluation. With this dataset, (Hermann et al., 2015) propose a deep LSTM reader that achieves an accuracy of 63.8% on CNN and 69.0% on Daily Mail. However, this deep LSTM reader is time consuming for training and no explanation can be found on why it works opposed to traditional approach. Therefore, we propose a traditional machine learning method on this new dataset to investigate what features can help with this task.

4   Model – Learning to Rank (L2R) Reader

Our model consists of learning to rank algorithms, features and semantics. As shown in Figure 1, given a document along with a list of queries that have the placeholder filled with different entities from the list of candidate answers. Then, through a feature extractor, we get several features for each entity (i.e. candidate answer). Combining with correct answer, we employ a learning to rank algorithm (i.e. L2R in Figure 1) to train a ranking model. Based on the ranking model, we generate ranking lists on new unseen dataset that has features ex-

tracted using feature extractor; from the rank list, we select the entity with highest ranking score as the predicted answer to the question.

4.1   Learning to Rank

Learning to rank is employed in Information Retrieval (IR) and Natural Language Processing (NLP). Generally, it is defined as follows: given a query, the ranking algorithm will generate a list of candidate documents with scores (Hang, 2011). In this task, we select the entity with the highest score as the answer to the question. Here, we introduce three different types of learning to rank algorithms that can help with the task.

Pointwise: In the pointwise approach, the ranking algorithm is transformed into problems including classification and regression to derive a score for every pair of document and query (Hang, 2011). This approach ignores the group structure of ranking and we do not apply it in this task.

Pairwise: In the pairewise approach, the ranking algorithm is transformed into problems of pairwise classification or pairwise regression (Hang, 2011). It also ignores the group structure of ranking. In this project, we mainly focus on the pairwise classification that employs a binary classifier in document pair ranking. We try approaches including RankNet (Burges et al., 2005), RankBoost (Freund et al., 2003), RankSVM (Herbrich et al., 1999), MART and LambdaMART (Burges, 2010)

Listwise: In the listwise approach, the ranking problem is addressed by taking the ranking list as instances in both learning and prediction process (Hang, 2011). It maintains the group structure and

we employ the following listwise approaches: List-Net (Cao et al., 2007), AdaRank (Xu and Li, 2007), Coordinate Ascent.

4.2 Features

We explore four types of features in this task. We start with the frequency of entity in both document and cloze-style question. Then we try features on word distance with different settings of window size. We further investigate features on syntactics and semantics to see how these affect the model performance.

4.2.1 Frequency

Frequency is explored based on one baseline of (Hermann et al., 2015). We simply count the number of entity that is in the candidate answer list appearing in the document and the question, then the count number works as the frequency feature of the entity. If this entity does not show in question or document, we assign a value of 0. The idea behind is that news article usually mentions important entities multiple times and cloze-style question is concerned with such entities.

4.2.2 Word Distance

We investigate word distance from three aspects – word alignment, nBOW, and word mover's distance (WMD) (Kusner et al., 2015).

Word Alignment (WA): For the word alignment, we first consider the situation shown in Figure 2. According to Figure 2, we first replace the "@place-
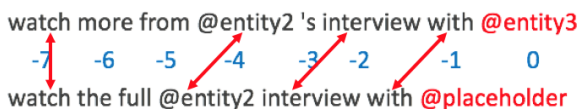

Figure 2: Example of Word Alignment

holder" with one entity in the candidate answer list and search the document to find one matched sentence that containing this entity. Then, we align these two entities and set location index as 0. Starting with this index, words located left will have negative index while words located right will have positive index. With index defined, we align same words of these two sentences and compute the difference of word indexes. As for words without alignment, a penalty is given. Finally, the score capture some

word information of question and document sentences.

Normalized Bag-Of-Words (nBOW): Furthermore, we consider using normalized bag-of-words (nBOW) vectors, $d \in R^n$, where d denotes the document vector and n is the vocabulary size. To be precise, if word i appears c_i times in the document, we let $d_i = \frac{c_i}{\sum_{j=1}^{n} c_j}$. Therefore, a sentence is transformed into a vector and we can do similarity measure accordingly.

Word Mover's distance (WMD): The WMD (Kusner et al., 2015) measures the dissimilarity between two text documents using the minimal distance that the embedded words of one document need to move to the embedded words of another document. Here, we apply WMD into two sentences after using Mikolov's word2vec (Mikolov et al., 2013) to convert words of sentences to embeddings. Different from the two methods above (WD and nBOW), the WMD can move words to semantically similar words, thereby capturing semantic information (see Figure 3). It can capture semantic similarity of two sentences with different unique words. For instance, "The President greets the press in Chicago" and "Obama speaks to the medis in Illinoirs".
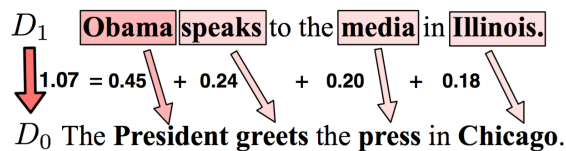

Figure 3: Example of WMD

Moreover, we try different window size to extract sentences in the document to optimize our model.

4.2.3 Syntactic Features

In this task, we consider syntactic features including part-of-speech (POS) tags and dependency parsing.

POS tags: Similar to the word alignment defined in Section 4.2.2, we consider POS tag alignment as one syntactic feature. Specifically, we transform words into POS tags using NLTK[1] and employ same technology as word alignment to measure the dissimilarity of two sentences.

---

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

Dependency Parsing: If two sentences describe same event, it is likely that they have dependency overlapping (Wang and McAllester, 2015). We thus incorporate dependency parsing to capture such information. To be precise, we use Stanford Parsing[2] to get dependencies of a sentence that are shown as several triples like (s, t, arc), e.g. (entity, has, nsubj), where s denotes source word and t is the target word. Then, this dependency-based similarity is evaluated from these three categories: (1) $s_d = s_q$ and $arc_d = arc_q$ or $t_d = t_q$ and $arc_d = arc_q$; (2) $s_d = s_q$ and $t_d = t_q$; (3) $s_d = s_q$, $t_d = t_q$ and $arc_d = arc_q$, where d denotes document, q denotes question and s_d refers to source word in document sentence while s_q represents the target word in question.

4.2.4 Semantic Features

In addition to word embeddings applied in WMD in Section 4.2.2. We adopt the SEMAFOR Frame Semantic (FS) parser (Das et al., 2014) to extract some semantic features. Figure 4 gives an example output of the SEMAFOR semantic parser. In this
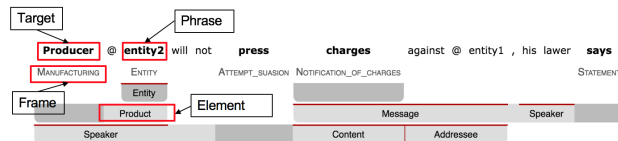

Figure 4: Example output from SEMAFOR

example, five frames are identified. For example, The word "says" is a **target**, which evokes a semantic **frame** labeled STATEMENT. Each frame has its own frame **elements**; e.g., the STATEMENT frame has frame elements of *Message* and *Speaker*. Features from these parsers have been shown to be useful for machine comprehension task (Wang and McAllester, 2015). We expect that the document sentence containing the answer will overlap with the question and correct answer in terms of targets, frames evoked and frame elements evoked. Therefore, we design the following features to capture this intuition. To be precise, after parsing a sentence, we get several triples composed of (t, f, e), where t denotes the target, f denotes the frame and e denote a set of elements. Then, the frame semantic based features are derived from the next seven categories:

(1) $t_q = t_d$; (2) $f_q = f_d$; (3) $e_q = e_d$; (4) $t_q = t_d$ and $f_q = f_d$; (5) $t_q = t_d$ and $e_q = e_d$; (6) $f_q = f_d$ and $e_q = e_d$; (7) $t_q = t_d$ and $f_q = f_d$ and $e_q = e_d$. We count the number of triples satisfying the above requirements from the document sentence and the query sentence to generate seven features.

4.3 Semantics

Semantics play a significant role in our model. This section summarizes how our model uses semantics to achieve reading comprehension. In this project, we use semantics in aspects of WMD, Frame Semantic (FS) and coreference. In WMD, we use word embeddings to capture word level semantics and FS helps us to capture sentence level semantics. The coreference is employed to identity chains of mentions within and across sentences for data pre-processing.

Specifically, word embeddings (Mikolov et al., 2013) project words into a low-dimensional space and similarity of vectors can capture some word similarity on semantics. For example, the word "Paris" is close to "Berlin" rather than "France" and vec("Paris") is closest to vec("Berlin") - vec("Germany") + vec("France"), where vec(x) denotes the word embedding of word x. Based on the word embedding given by (Mikolov et al., 2013), the WMD can align semantically similar words together using distance measure, for example, it is much cheaper to transform "Illinois" into "Chicago" than "Japan" into "Chicago". Therefore, we incorporate semantics into the model and its performance is improved.

As for frame semantics, the semantic similarity that two sentences describe same events but use different words or two sentence have different structures can be captured by frame semantics parsing. For example, two sentence "the speaker states that he is innocent." and "'I'm innocent', he says." would be parsed to have the same semantic frame of STATEMENT and frame elements of (Message, Speaker), even they use different words and have inverse sentence order.

Coreference resolution is achieved using Stanford CoreNLP[3]. We try to resolve the pronoun with the specific description and run the coreference resolu-

tion system on each document. As the coreference system will provide a chain of mentions, we take the representative mention to resolve pronoun in the document only, i.e., replacing the pronoun like "it" with the representative mention of its coreference chain.

5 Evaluation

We evaluate the L2R reader system by comparing with the baselines of (Hermann et al., 2015). Moreover, we present the system performance with different learning to rank algorithms, based on which, we select RankSVM and LamdaMART as the ranking algorithm for model training. We also evaluate the contribution of single feature to the system performance for final feature decision. Based on the final results, we analyze the effect of incorporating semantics into the system from coreference, word embeddings, and frame semantics. Following by semantic analysis, we conduct error analysis to see how to improve the system in the future work.

5.1 Experimental Results

Final Results: Table 5.1 shows the best performance of our model and some baselines. Our L2R model finally combines all the three kinds of word distance features and frame semantic features described in section 4.2. Our **L2R Reader** outperforms LSTM-based models proposed in (Hermann et al., 2015) on the CNN dataset and achieves competitive results on the Daily Mail dataset.

	CNN		Daily Mail	
	Train	Test	Train	Test
Deep LSTM[†]	55	57	63.3	62.2
Attentive Reader[†]	61.6	63	**70.5**	**69**
Impatient Reader[†]	61.8	63.8	69	68
L2R Reader	**64.3**	**65.8**	69.1	67.3

Table 3: Results of our L2R Reader on the CNN and Daily Mail dataset. Results marked with [†] are taken from previous paper.

Different L2R Algorithms: Table 4 shows the performance of different learning to rank algorithms by using the same features. Model parameters are tuned on the validation set. We found that **RankSVM** and **LambdaMART** are two best L2R algorithms on this task. RankSVM performs best

on the CNN dataset while LambdaMART performs best on the Daily Mail dataset.

Ranking Model	CNN Test	Daily Mail Test
RankSVM	**65.8**	66.7
MART	60.4	65.3
RankNet	40.9	32.8
RankBoost	32.0	28.4
AdaRank	18.0	12.7
Coordinate Asecent	59.0	54.4
LambdaMART	64.2	**67.3**
ListNet	32.7	32.3
Random Forest	63.4	65.6

Table 4: Performance of different L2R algorithms on CNN and Daily Mail dataset.

Single Feature Performance: We evaluate the performance of each single feature. Table 5 shows the results. We can see that among all single features, word alignment features perform best and frame semantic features perform worst. This can be explained by the nature of the dataset.

	CNN		Daily Mail	
	Train	Test	Train	Test
Frequency	33.2	35.0	33.4	32.4
WA	47.3	50.6	54.3	53.4
nBOW	40.3	43.5	48.7	47.8
WMD	41.2	44.0	49.3	48.5
FS	18.9	22.0	25.2	24.3

Table 5: Single feature performance on CNN and Daily Mail dataset.

Performance vs Size of Training Data: One great advantage of our model is that it only requires a small set of training data compared with neural network based models. Table 6 shows the relationship between our model performance and the number of training data. We can see that our model have a high score given only about 100 training data. Our model performance improves when giving more data, but the speed of improvement gets dramatically slow. Figure 5 shows the trend of the convergence of our model with more data.

	CNN		Dailymail	
#Training	Val	Test	Val	Test
10	35.2	41.6	45.7	43.6
20	57.4	55.1	57.2	56.2
30	56.7	60.2	61.4	60.3
40	57	60	62.3	61.3
50	60.3	63.1	63.5	62.5
100	61.5	63.9	65.4	64.6
200	62.5	64.9	67.3	65.2
500	62.8	65	67.5	66.3
1000	62.9	65.2	68.3	66.7
2000	63.2	65.2	69.0	67.3
5000	64.3	65.8	69.1	67.3

Table 6: Model performance for L2R Reader against different number of training data on the CNN and Daily Mail dataset. All the statistics are averaged over 10 times of random samples. Parameters are tuned on the validation set and the best validation model is applied to the test data.

5.2 Semantics Analysis

We also test the performance of our semantic components, which are **co-reference**, **word embeddings** and **frame semantics**.

	CNN		Daily Mail	
Models	Val	Test	Val	Test
L2R Reader	64.3	65.8	69.1	67.3
L2R+Coref	63.8	64.8	68.3	66.5
L2R-WMD	60.8	61.5	63.2	61.6
L2R-FS	61.5	62.5	65.3	63.7

Table 7: Analysis of semantic components of our model. "+"" and "-" refer to added or ablated components. Coref, WMD and FS denotes coreference system, word mover's distance and frame semantics.

Table 7 shows the results of adding co-reference, deleting word mover's distance and deleting frame semantics. From the experimental results, we can see that coreference system does not bring improvement to our system, and even harm the performance. This might because the coreference system (Stanford CoNLP) we use in our system cannot have good performance when the sentences are complicated. We can also find from the results that the **word embedding** and **frame semantics** components plays a vital role in our system, although the single feature performance of frame semantics is quite low.
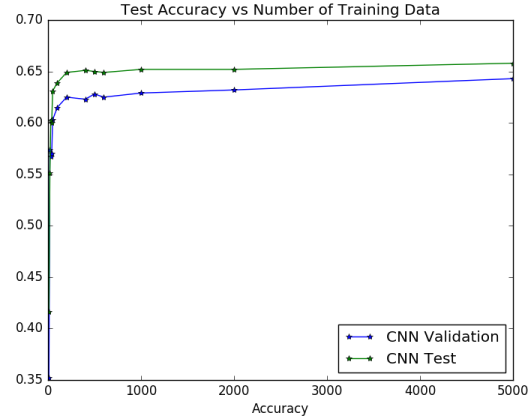

Figure 5: Model performance for L2R Reader against different number of training data on the CNN dataset.

5.3 Error Analysis

To give insight into our system's performance and reveal future research directions, we analyze the errors made by our system. We found that many queries require text summarization, event detection, background knowledge and inference. We also found an error on the CNN dataset (refer to Figure 6). We make some detailed analysis as follows.

> Doc: (@entity1) @entity0 and @entity2 , welcome to parenthood . the celebrity couple announced the arrival of their son , @entity6 , in statements to people .
> Query: @placeholder and @entity2 welcome son @entity6.
> Correct: @entity6.
> Predicted: @entity0

Figure 6: Wrong answer in the gold standard data.

Figure 6 shows an example of wrong answer in the gold standard dataset. We can see from the query that the correct answer should be entity0 rather than entity6. Our model successfully get the right answer. The error in the dataset might due to the generation process of the dataset.

Figure 7 shows an example of requiring high level text summarization. The phrases "within 10 minutes" and "kick off" do not appear in the document, but they are high level summarization of the document. Hence, our model lacks the ability of getting the correct answer in such situation.

Figure 8 shows an example of requiring event detection in filling the cloze question. The word "collapse" appears several times in the document, but describes different events. Our model fails to cap-

```
Doc: ( @entity2 ) @entity1 striker @entity0 , who has a history of
fainting , said he is now " feeling well " after collapsing during @entity6
's 3 - 2 loss at @entity7 in the @entity8 on wednesday . the worrying
incident occurred in the first half at @entity11 -- after @entity7 scored
in the seventh minute -- but the 29 - year - old left the pitch conscious
following about five minutes of treatment .

Query:@entity0 collapses within 10 minutes of kickoff at @placeholder

Correct: @entity7

Prediction: @entity11
```

Figure 7: Require high level text summarization.

```
Doc: ( @entity2 ) @entity1 striker @entity0 , who has a history of fainting ,
said he is now " feeling well " after collapsing during @entity6 's 3 - 2 loss
at @entity7 in the @entity8 on wednesday . almost exactly three years
ago at @entity11 , then @entity82 midfielder @entity81 collapsed after
suffering a cardiac arrest .

Query: the incident came three years after @placeholder collapsed at
@entity11

Correct: @entity81

Prediction: @entity7
```

Figure 8: Require event detection.

ture the difference.

```
Doc: ( @entity1 ) @entity0 , co-star of " @entity3 " — the @entity4
preschool show that ran on @entity5 during the early 1990s — has
died , aged 78 . her son , @entity6 , told @entity7 that his mother died
in @entity9 on wednesday night from a rare form of cancer first
diagnosed last october .

Query: beloved children 's performer @placeholder has died

Correct: @entity0

Prediction: @entity7
```

Figure 9: Require background knowledge and inference.

Figure 9 shows an example of requiring back-
ground knowledge and inference in filling the cloze
question. It can be inferred that "preschool show" is
performed by "children's performer". However, this
inference require some background knowledge. Our
model cannot perform well in such situation.

6 Conclusion

We explored the new cloze style reading compre-
hension task and designed a learning to rank (L2R)
reader system to provide a solution. We incorporate

semantics such as word embeddings, frame seman-
tics and coreference resolution into our system and
show that they can greatly improve our model per-
formance. We find that our model is poor at high
level text summarization, event detection and infer-
ence through error analysis. We will investigate into
how to solve these kinds of problems by using se-
mantics in the future.

References

[Burges et al.2005] Chris Burges, Tal Shaked, Erin Ren-
shaw, Ari Lazier, Matt Deeds, Nicole Hamilton, and
Greg Hullender. 2005. Learning to rank using gradi-
ent descent. In *Proceedings of the 22nd international
conference on Machine learning*, pages 89–96. ACM.

[Burges2010] Christopher JC Burges. 2010. From
ranknet to lambdarank to lambdamart: An overview.
Learning, 11:23–581.

[Cao et al.2007] Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-
Feng Tsai, and Hang Li. 2007. Learning to rank: from
pairwise approach to listwise approach. In *Proceed-
ings of the 24th international conference on Machine
learning*, pages 129–136. ACM.

[Das et al.2014] Dipanjan Das, Desai Chen, André FT
Martins, Nathan Schneider, and Noah A Smith. 2014.
Frame-semantic parsing. *Computational Linguistics*,
40(1):9–56.

[Freund et al.2003] Yoav Freund, Raj Iyer, Robert E
Schapire, and Yoram Singer. 2003. An efficient boost-
ing algorithm for combining preferences. *The Journal
of machine learning research*, 4:933–969.

[Hang2011] LI Hang. 2011. A short introduction to
learning to rank. *IEICE TRANSACTIONS on Infor-
mation and Systems*, 94(10):1854–1862.

[Herbrich et al.1999] Ralf Herbrich, Thore Graepel, and
Klaus Obermayer. 1999. Large margin rank bound-
aries for ordinal regression. *Advances in neural infor-
mation processing systems*, pages 115–132.

[Hermann et al.2015] Karl Moritz Hermann, Tomas Ko-
cisky, Edward Grefenstette, Lasse Espeholt, Will Kay,
Mustafa Suleyman, and Phil Blunsom. 2015. Teach-
ing machines to read and comprehend. In *Advances in
Neural Information Processing Systems*, pages 1684–
1692.

[Kusner et al.2015] Matt Kusner, Yu Sun, Nicholas
Kolkin, and Kilian Q Weinberger. 2015. From word
embeddings to document distances. In *Proceedings of
the 32nd International Conference on Machine Learn-
ing (ICML-15)*, pages 957–966.

[Mikolov et al.2013] Tomas Mikolov, Ilya Sutskever, Kai
Chen, Greg S Corrado, and Jeff Dean. 2013. Dis-
tributed representations of words and phrases and their

compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

[Narasimhan and Barzilay2015] Karthik Narasimhan and Regina Barzilay. 2015. Machine comprehension with discourse relations. In *53rd Annual Meeting of the Association for Computational Linguistics*.

[Richardson et al.2013] Matthew Richardson, Christopher JC Burges, and Erin Renshaw. 2013. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *EMNLP*, volume 1, page 2.

[Sachan et al.2015] Mrinmaya Sachan, Avinava Dubey, Eric P Xing, and Matthew Richardson. 2015. Learning answerentailing structures for machine comprehension. In *Proceedings of ACL.*

[Smith et al.2015] Ellery Smith, Nicola Greco, Matko Bošnjak, and Andreas Vlachos. 2015. A strong lexical matching method for the machine comprehension test.

[Taylor1953] Wilson L Taylor. 1953. Cloze procedure: a new tool for measuring readability. *Journalism and Mass Communication Quarterly*, 30(4):415.

[Trischler et al.2016] Adam Trischler, Zheng Ye, Xingdi Yuan, Jing He, Phillip Bachman, and Kaheer Suleman. 2016. A parallel-hierarchical model for machine comprehension on sparse data. *arXiv preprint arXiv:1603.08884*.

[Wang and McAllester2015] Hai Wang and Mohit Bansal Kevin Gimpel David McAllester. 2015. Machine comprehension with syntax, frames, and semantics. *Volume 2: Short Papers*, page 700.

[Xu and Li2007] Jun Xu and Hang Li. 2007. Adarank: a boosting algorithm for information retrieval. In *Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 391–398. ACM.