Machine Comprehension with Syntax, Frames, and Semantics

Hai Wang **Mohit Bansal** **Kevin Gimpel** **David McAllester**
Toyota Technological Institute at Chicago, Chicago, IL, 60637, USA
{haiwang,mbansal,kgimpel,mcallester}@ttic.edu

Abstract

We demonstrate significant improvement on the MCTest question answering task (Richardson et al., 2013) by augmenting baseline features with features based on syntax, frame semantics, coreference, and word embeddings, and combining them in a max-margin learning framework. We achieve the best results we are aware of on this dataset, outperforming concurrently-published results. These results demonstrate a significant performance gradient for the use of linguistic structure in machine comprehension.

1 Introduction

Recent question answering (QA) systems (Ferrucci et al., 2010; Berant et al., 2013; Bordes et al., 2014) have focused on open-domain factoid questions, relying on knowledge bases like Freebase (Bollacker et al., 2008) or large corpora of unstructured text. While clearly useful, this type of QA may not be the best way to evaluate natural language understanding capability. Due to the redundancy of facts expressed on the web, many questions are answerable with shallow techniques from information extraction (Yao et al., 2014).

There is also recent work on QA based on synthetic text describing events in adventure games (Weston et al., 2015; Sukhbaatar et al., 2015). Synthetic text provides a cleanroom environment for evaluating QA systems, and has spurred development of powerful neural architectures for complex reasoning. However, the formulaic semantics underlying these synthetic texts allows for the construction of perfect rule-based question answering systems, and may not reflect the patterns of natural linguistic expression.

In this paper, we focus on **machine comprehension**, which is QA in which the answer is contained within a provided passage. Several comprehension tasks have been developed, including Remedia (Hirschman et al., 1999), CBC4kids (Breck et al., 2001), and the QA4MRE textual question answering tasks in the CLEF evaluations (Peñas et al., 2011; Peñas et al., 2013; Clark et al., 2012; Bhaskar et al., 2012).

We consider the Machine Comprehension of Text dataset (MCTest; Richardson et al., 2013), a set of human-authored fictional stories with associated multiple-choice questions. Knowledge bases and web corpora are not useful for this task, and answers are typically expressed just once in each story. While simple baselines presented by Richardson et al. answer over 60% of questions correctly, many of the remaining questions require deeper analysis.

In this paper, we explore the use of dependency syntax, frame semantics, word embeddings, and coreference for improving performance on MCTest. Syntax, frame semantics, and coreference are essential for understanding who did what to whom. Word embeddings address variation in word choice between the stories and questions. Our added features achieve the best results we are aware of on this dataset, outperforming concurrently-published results (Narasimhan and Barzilay, 2015; Sachan et al., 2015).

2 Model

We use a simple latent-variable classifier trained with a max-margin criterion. Let P denote the passage, q denote the question of interest, and A denote the set of candidate answers for q, where each $a \in A$ denotes one candidate answer. We want to learn a function $h : (P, q) \rightarrow A$ that, given a passage and a question, outputs a legal $a \in A$. We use a linear model for h that uses a latent variable w to identify the sentence in the passage in which the answer can be found.

Let W denote the set of sentences within the passage, where a particular $w \in W$ denotes one sentence.

Given a feature vector $f(P, w, q, a)$ and a weight vector θ with an entry for each feature, the prediction $\hat{a}$ for a new P and q is given by:

$$\hat{a} = \arg\max_{a \in A} \max_{w \in W} \theta^{\top} f(P, w, q, a)$$

Given triples $\{\langle P^i, q^i, a^i \rangle\}_{i=1}^{n}$, we minimize an ℓ_2-regularized max-margin loss function:

$$\min_{\theta} \lambda ||\theta||^2 + \sum_{i=1}^{n} \Bigg\{ -\max_{w \in W} \theta^{\top} f(P^i, w, q^i, a^i) + \max_{a \in A} \Big\{ \max_{w' \in W} \theta^{\top} f(P^i, w', q^i, a) + \Delta(a, a^i) \Big\} \Bigg\}$$

where λ is the weight of the ℓ_2 term and $\Delta(a, a^i) = 1$ if $a \neq a^i$ and 0 otherwise. The latent variable w makes the loss function non-convex.

3 Features

We start with two features from Richardson et al. (2013). Our first feature corresponds to their sliding window similarity baseline, which measures weighted word overlap between the bag of words constructed from the question/answer and the bag of words in the window. We call this feature B. The second feature corresponds to their word distance baseline, and is the minimal distance between two word occurrences in the passage that are also contained in the question/answer pair. We call this feature D. Space does not permit a detailed description.

3.1 Frame Semantic Features

Frame semantic parsing (Das et al., 2014) is the problem of extracting frame-specific predicate-argument structures from sentences, where the frames come from an inventory such as FrameNet (Baker et al., 1998). This task can be decomposed into three subproblems: *target identification*, in which frame-evoking predicates are marked; *frame label identification*, in which the evoked frame is selected for each predicate; and *argument identification*, in which arguments to each frame are identified and labeled with a role from the frame. An example output of the SEMAFOR frame semantic parser (Das et al., 2014) is given in Figure 1.

Three frames are identified. The target words *pulled*, *all*, and *shelves* have respective frame labels CAUSE_MOTION, QUANTITY, and NATURAL_FEATURES. Each frame has its own set of arguments; e.g., the CAUSE_MOTION frame has the labeled *Agent*, *Theme*, and *Goal* arguments. Features from these parses have been shown to be useful for NLP tasks such as slot filling in spoken dialogue systems (Chen et al., 2013). We expect that the passage sentence containing the answer will overlap with the question and correct answer in terms of predicates, frames evoked, and predicted argument labels, and we design features to capture this intuition. Given the frame semantic parse for a sentence, let T be the bag of frame-evoking target words/phrases.[1] We define the bag of frame labels in the parse as F. For each target $t \in T$, there is an associated frame label denoted $F_t \in F$. Let R be the bag of phrases assigned with an argument label in the parse. We denote the bag of argument labels in the parse by L. For each phrase $r \in R$, there is an argument label denoted $L_r \in L$. We define a frame semantic parse as a tuple $\langle T, F, R, L \rangle$. We define six features based on two parsed sentences $\langle T^1, F^1, R^1, L^1 \rangle$ and $\langle T^2, F^2, R^2, L^2 \rangle$:

- f_1: # frame label matches: $|\{\langle s, t \rangle : s \in F^1, t \in F^2, s = t\}|$
- f_2: # argument label matches: $|\{\langle s, t \rangle : s \in L^1, t \in L^2, s = t\}|$.
- f_3: # target matches, ignoring frame labels: $|\{\langle s, t \rangle : s \in T^1, t \in T^2, s = t\}|$.
- f_4: # argument matches, ignoring arg. labels: $|\{\langle s, t \rangle : s \in R^1, t \in R^2, s = t\}|$.
- f_5: # target matches, using frame labels: $|\{\langle s, t \rangle : s \in T^1, t \in T^2, s = t, F_s^1 = F_t^2\}|$.
- f_6: # argument matches, using arg. labels: $|\{\langle s, t \rangle : s \in R^1, t \in R^2, s = t, L_s^1 = L_t^2\}|$.

We use two versions of each of these six features: one version for the passage sentence w and the question q, and an additional version for w and the candidate answer a.

3.2 Syntactic Features

If two sentences refer to the same event, then it is likely that they have some overlapping dependen-

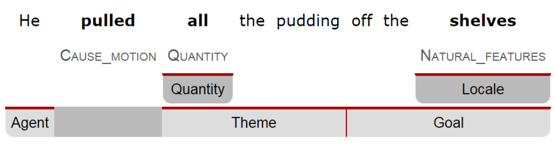

Figure 1: Example output from SEMAFOR.

[1]By *bag*, we mean here a set with possible replicates.

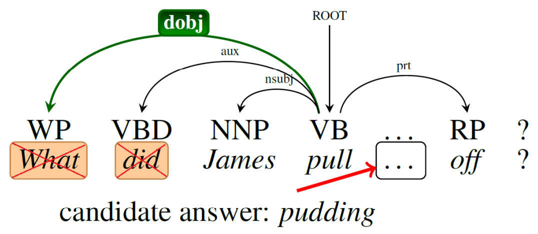

Figure 2: Transforming the question to a statement.

cies. To compare a Q/A pair to a sentence in the passage, we first use rules to transform the question into a statement and insert the candidate answer into the trace position. Our simple rule set is inspired by the rich history of QA research into modeling syntactic transformations between questions and answers (Moschitti et al., 2007; Wang et al., 2007; Heilman and Smith, 2010). Given Stanford dependency tree and part-of-speech (POS) tags for the question, let arc(u, v) be the label of the dependency between child word u and head word v, let $POS(u)$ be the POS tag of u, let c be the *wh*-word in the question, let r be the root word in the question's dependency tree, and let a be the candidate answer. We use the following rules:[2]

- c = *what*, $POS(r)$ = VB, and arc(c, r) = dobj. Insert a after word u where arc(u, r) = nsubj. Delete c and the word after c.
- c = *what*, $POS(r)$ = NN, and arc(c, r) = nsubj. Replace c by a.
- c = *where*, $POS(r)$ = VB, and arc(c, r) = advmod. Delete c and the word after c. If r has a child u such that arc(u, r) = dobj, insert a after u; else, insert a after r and delete r.
- c = *where*, r = *is*, $POS(r)$ = VBZ, and arc(c, r) = advmod. Delete c. Find r's child u such that arc(u, r) = nsubj, move r to be right after u. Insert a after r.
- c = *who*, $POS(r)$ = NN, and arc(c, r) = nsubj. Replace c by a.
- c = *who*, $POS(r) \in$ {VB, VBD}, and arc(c, r) = nsubj. Replace c by a.

We use other rules in addition to those above: change "why x?" to "the reason x is a", and change "how many x", "how much x", or "when x" to "x a".

Given each candidate answer, we attempt to transform the question to a statement using the rules above.[3] An example of the transformation is given in Figure 2. In the parse, *pull* is the root word and *What* is attached as a dobj. This matches the first rule, so we delete *did* and insert the candidate answer *pudding* after *pull*, making the final transformed sentence: *James pull pudding off*.

After this transformation of the question (and a candidate answer) to a statement, we measure its similarity to the sentence in the window using simple dependency-based similarity features. Denoting a dependency as $(u, v, \text{arc}(u, v))$, then two dependencies $(u_1, v_1, \text{arc}(u_1, v_1))$ and $(u_2, v_2, \text{arc}(u_2, v_2))$ match if and only if $u_1 = u_2$, $v_1 = v_2$, and $\text{arc}(u_1, v_1) = \text{arc}(u_2, v_2)$. One feature simply counts the number of dependency matches between the transformed question and the passage sentence. We include three additional count features that each consider a subset of dependencies from the following three categories: (1) $v = r$ and $u = a$; (2) $v = r$ but $u \neq a$; and (3) $v \neq r$. In Figure 2, the triples (*James*, *pull*, nsubj) and (*off*, *pull*, prt) belong to the second category while (*pudding*, *pull*, dobj) belongs to the first.

3.3 Word Embeddings

Word embeddings (Mikolov et al., 2013) represent each word as a low-dimensional vector where the similarity of vectors captures some aspect of semantic similarity of words. They have been used for many tasks, including semantic role labeling (Collobert et al., 2011), named entity recognition (Turian et al., 2010), parsing (Bansal et al., 2014), and for the Facebook QA tasks (Weston et al., 2015; Sukhbaatar et al., 2015). We first define the vector f_w^+ as the vector summation of all words inside sentence w and $f_w^\times$ as the element-wise multiplication of the vectors in w. To define vectors for answer a for question q, we concatenate q and a, then calculate f_{qa}^+ and $f_{qa}^\times$. For the bag-of-words feature B, instead of merely counting matches of the two bags of words, we also use $\cos(f_{qa}^+, f_w^+)$ and $\cos(f_{qa}^\times, f_w^\times)$ as features, where cos is cosine similarity. For syntactic features, where τ_w is the bag of dependencies of w and τ_{qa} is the bag of dependencies for the transformed question for candidate answer a, we use a feature function that returns the following:

$$\sum_{(u,v,\ell)\in\tau_w} \sum_{(u',v',\ell')\in\tau_{qa}} \mathbb{1}_{\ell=\ell'} \cos(u,u') \cos(v,v')$$

[2]There are existing rule-based approaches to transforming *statements* to questions (Heilman, 2011); our rules reverse this process.

[3]If no rule applies, we return 0 for all syntactic features.

where ℓ is short for arc(u, v).[4]

3.4 Coreference Resolution

Coreference resolution systems aim to identify chains of mentions (within and across sentences) that refer to the same entity. We integrate coreference information into the bag-of-words, frame semantic, and syntactic features. We run a coreference resolution system on each passage, then for these three sets of features, we replace exact string match with a check for membership in the same coreference chain.

When using features augmented by word embeddings or coreference, we create new versions of the features that use the new information, concatenating them with the original features.

4 Experiments

MCTest splits its stories into train, development, and test sets. The original MCtest DEV is too small, to choose the best feature set, we merged the train and development sets in MC160 and MC500 and split them randomly into a 250-story training set (TRAIN) and a 200-story development set (DEV). We optimize the max-margin training criteria on TRAIN and use DEV to tune the regularizer λ and choose the best feature set. We report final performance on the original two test sets (for comparability) from MCTest, named MC160 and MC500.

We use SEMAFOR (Das et al., 2010; Das et al., 2014) for frame semantic parsing and the latest Stanford dependency parser (Chen and Manning, 2014) as our dependency parser. We use the Stanford rule-based system for coreference resolution (Lee et al., 2013). We use the pre-trained 300-dimensional word embeddings downloadable from the `word2vec` site.[5] We denote the frame semantic features by F and the syntactic features by S. We use superscripts ^{w} and ^{c} to indicate the use of embeddings and coreference for a particular feature set. To minimize the loss, we use the `miniFunc` package in MATLAB with LBFGS (Nocedal, 1980; Liu and Nocedal, 1989).

The accuracy of different feature sets on DEV is given in Table 1.[6] The boldface results correspond to the best feature set combination chosen by evaluating on DEV. In this case, the feature dimensionality is 29, which includes 4 bag-of-words features, 1 distance feature, 12 frame semantic features, and with the remaining being syntactic features. After choosing the best feature set on DEV, we then evaluate our system on TEST.

Negations: in preliminary experiments, we found that our system suffered with negation questions, so we developed a simple heuristic to deal with them. We identify a question as negation if it contains "not" or "n't" and does not begin with "how" or "why". If a question is identified as negation, we then negate the final score for each candidate answer.

Features	DEV Accuracy (%)
B + D + F	64.18
B + D + F + S	66.24
B^{wc} + D + F^c + S^{wc}	**69.87**

Table 1: Accuracy on DEV.

The final test results are shown in Table 2. We first compare to results from prior work (Richardson et al., 2013). Their first result uses a sliding window with the bag-of-words feature B described in Sec. 3; this system is called "Baseline 1" (B1). They then add the distance feature D, also described in Sec. 3. The combined system, which uses B and D, is called "Baseline 2" (B2). Their third result adds a rich textual entailment system to B2; it is referred to as B2+RTE.[7] We also compare to concurrently-published results (Narasimhan and Barzilay, 2015; Sachan et al., 2015).

We report accuracies for all questions as well as separately for the two types: those that are answerable with a single sentence from the passage ("Single") and those that require multiple sentences ("Multiple"). We see gains in accuracy of 6% absolute compared to the B2+RTE baseline and also outperform concurrently-published results (Narasimhan and Barzilay, 2015; Sachan et al., 2015). Even though our system only explicitly uses a single sentence from the passage when choosing an answer, we improve baseline accuracy for both single-sentence and multiple-sentence questions. [8]

[4]Similar to the original syntactic features (see end of Section 3.2), we also have 3 additional features for the three subset categories.

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

[6]All accuracies are computed with tie-breaking partial credit (similar to previous work), i.e., if we have the same score for all four candidate answers, then we get partial credit of 0.25 for this question.

[7]These three results are obtained from files at `http://research.microsoft.com/en-us/um/redmond/projects/mctest/results.html`.

[8]However, we inspected these question annotations and

System		MC160			MC500		
		Single (112)	Multiple (128)	All	Single (272)	Multiple (328)	All
	B1	64.73	56.64	60.41	58.21	56.17	57.09
Richardson et al. (2013)	B2	75.89	60.15	67.50	64.00	57.46	60.43
	B2+RTE	76.78	62.50	69.16	68.01	59.45	63.33
Narasimhan and Barzilay (2015)		82.36	65.23	73.23	68.38	59.90	63.75
Sachan et al. (2015)		-	-	-	67.65	**67.99**	67.83
our system		**84.22**	**67.85**	**75.27**	**72.05**	67.94	**69.94**

Table 2: Accuracy comparison of published results on test sets.

Features	DEV Accuracy (%)
full (B^{wc}+D+F^{c}+S^{wc})	69.87
– B^{wc} (D + F^{c}+S^{wc})	58.46
– D (B^{wc}+F^{c}+S^{wc})	65.89
– B^{wc}, – D (F^{c}+S^{wc})	54.19
– embeddings (B^{c}+D+F^{c}+S^{c})	68.28
– coreference (B^{w}+D+F+S^{w})	68.43
– frame semantics (B^{wc}+D+S^{wc})	67.89
– syntax (B^{wc}+D+F^{c})	67.64
– negation (B^{wc}+D+F^{c}+S^{wc})	68.72

Table 3: Ablation study of feature types on the dev set.

We also measure the contribution of each feature set by deleting it from the full feature set. These ablation results are shown in Table 3. We find that frame semantic and syntax features contribute almost equally, and using word embeddings contributes slightly more than coreference information. If we delete the bag-of-words and distance features, then accuracy drops significantly, which suggests that in MCTest, simple surface-level similarity features suffice to answer a large portion of questions.

5 Analysis

Successes To show the effects of different features, we show cases where the full system gives the correct prediction (marked with ∗) but ablating the named features causes the incorrect answer (marked with †) to be predicted:

Ex. 1: effect of embeddings: we find the soft similarity between 'noodle' and 'spaghetti'.

***clue**: Marsha's favorite dinner was spaghetti.*
***q**: What is Marsha's noodle made out of? ∗**A)** Spaghetti; †**C)** mom;*

Ex. 2: coreference resolves *She* to *Hannah Harvey*.

Hannah Harvey was a ten year old. She lived in New York.
***q**: Where does Hannah Harvey live? ∗**A)** New York; †**C)** Kenya;*

Ex. 4: effect of syntax: by inserting answer **C**, the transformed statement is: *Todd say there's no place like home when he got home from the city.*

When his mom asked him about his trip to the city Todd said, "There's no place like home."
***q**: What did Todd say when he got home from the city? †**B)** There were so many people in cars; ∗**C)** There's no place like home;*

Errors To give insight into our system's performance and reveal future research directions, we also analyzed the errors made by our system. We found that many required inferential reasoning, counting, set enumeration, multiple sentences, time manipulation, and comparisons. Some randomly sampled examples are given below, with the correct answer starred (∗):

Ex. 1: requires inference across multiple sentences:

*One day Fritz got a splinter in his foot. Stephen did not believe him. Fritz showed him the picture. Then Stephen believed him. **q**: What made Stephen believe Fritz? ∗**A)** the picture of the splinter in his foot; †**C)** the picture of the cereal with milk;*

Ex. 2: requires temporal reasoning and world knowledge:

*Ashley woke up bright and early on Friday morning. Her birthday was only a day away. **q**: What day of the week was Ashley's birthday? ∗**A)** Saturday; †**C)** Friday;*

Ex. 3: requires comparative reasoning:

*Tommy has an old bicycle now. He is getting too big for it. **q**: What's wrong with Tommy's old bicycle? ∗**B)** it's too small; †**C)** it's old;*

6 Conclusion

We proposed several novel features for machine comprehension, including those based on frame semantics, dependency syntax, word embeddings, and coreference resolution. Empirical results demonstrate substantial improvements over several strong baselines, achieving new state-of-the-art results on MCTest. Our error analysis suggests that deeper linguistic analysis and inferential reasoning can yield further improvements on this task.

Acknowledgments

We thank Dekang Lin, Nathan Schneider and the anonymous reviewers for helpful comments.

occasionally found them to be noisy, which may cloud these comparisons.

References

Collin F. Baker, Charles J. Fillmore, and John B. Lowe. 1998. The Berkeley FrameNet project. In *Proceedings of the 17th International Conference on Computational Linguistics-Volume 1*, pages 86–90. Association for Computational Linguistics.

Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2014. Tailoring continuous word representations for dependency parsing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 809–815, Baltimore, Maryland, June. Association for Computational Linguistics.

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Proceedings of EMNLP*.

Pinaki Bhaskar, Partha Pakray, Somnath Banerjee, Samadrita Banerjee, Sivaji Bandyopadhyay, and Alexander F Gelbukh. 2012. Question answering system for QA4MRE@CLEF 2012. In *CLEF (Online Working Notes/Labs/Workshop)*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250. ACM.

Antoine Bordes, Sumit Chopra, and Jason Weston. 2014. Question answering with subgraph embeddings. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 615–620, Doha, Qatar, October. Association for Computational Linguistics.

Eric Breck, Marc Light, Gideon S Mann, Ellen Riloff, Brianne Brown, Pranav Anand, Mats Rooth, and Michael Thelen. 2001. Looking under the hood: Tools for diagnosing your question answering engine. In *Proceedings of the workshop on Open-domain question answering-Volume 12*, pages 1–8. Association for Computational Linguistics.

Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 740–750, Doha, Qatar, October. Association for Computational Linguistics.

Yun-Nung Chen, William Yang Wang, and Alexander I Rudnicky. 2013. Unsupervised induction and filling of semantic slots for spoken dialogue systems using frame-semantic parsing. In *Automatic Speech Recognition and Understanding (ASRU), 2013 IEEE Workshop on*, pages 120–125. IEEE.

Peter Clark, Philip Harrison, and Xuchen Yao. 2012. An entailment-based approach to the QA4MRE challenge. In *CLEF (Online Working Notes/Labs/Workshop)*.

Ronan Collobert, Jason Weston, Léon Bottou, Michael Karlen, Koray Kavukcuoglu, and Pavel Kuksa. 2011. Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537, November.

Dipanjan Das, Nathan Schneider, Desai Chen, and Noah A. Smith. 2010. Probabilistic frame-semantic parsing. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 948–956, Los Angeles, California, June. Association for Computational Linguistics.

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

David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. 2010. Building Watson: An overview of the DeepQA project. *AI magazine*, 31(3):59–79.

Michael Heilman and Noah A. Smith. 2010. Tree edit models for recognizing textual entailments, paraphrases, and answers to questions. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 1011–1019, Los Angeles, California, June. Association for Computational Linguistics.

M. Heilman. 2011. *Automatic factual question generation from text*. Ph.D. thesis, Carnegie Mellon University.

Lynette Hirschman, Marc Light, Eric Breck, and John D Burger. 1999. Deep read: A reading comprehension system. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*, pages 325–332. Association for Computational Linguistics.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Comput. Linguist.*, 39(4):885–916, December.

D. C. Liu and J. Nocedal. 1989. On the limited memory BFGS method for large scale optimization. *Math. Programming*, 45:503–528.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In C.J.C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Alessandro Moschitti, Silvia Quarteroni, Roberto Basili, and Suresh Manandhar. 2007. Exploiting syntactic and shallow semantic kernels for question answer classification. In *Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 776–783, Prague, Czech Republic, June. Association for Computational Linguistics.

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

Jorge Nocedal. 1980. Updating quasi-newton matrices with limited storage. *Mathematics of Computation*, 35:773–782.

Anselmo Peñas, Eduard H Hovy, Pamela Forner, Álvaro Rodrigo, Richard Sutcliffe, Corina Forascu, and Caroline Sporleder. 2011. Overview of QA4MRE at CLEF 2011: Question answering for machine reading evaluation. In *CLEF (Notebook Papers/Labs/Workshop)*, pages 1–20.

Anselmo Peñas, Eduard Hovy, Pamela Forner, Álvaro Rodrigo, Richard Sutcliffe, and Roser Morante. 2013. QA4MRE 2011-2013: Overview of question answering for machine reading evaluation. In *Information Access Evaluation. Multilinguality, Multimodality, and Visualization*, pages 303–320. Springer.

Matthew Richardson, Christopher J.C. Burges, and Erin Renshaw. 2013. MCTest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 193–203, Seattle, Washington, USA, October. Association for Computational Linguistics.

Mrinmaya Sachan, Avinava Dubey, Eric P. Xing, and Matthew Richardson. 2015. Learning answer-entailing structures for machine comprehension. In *53rd Annual Meeting of the Association for Computational Linguistics*.

S. Sukhbaatar, A. Szlam, J. Weston, and R. Fergus. 2015. Weakly supervised memory networks. March.

Joseph Turian, Lev Ratinov, and Yoshua Bengio. 2010. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 384–394, Stroudsburg, PA, USA. Association for Computational Linguistics.

M. Wang, N. A. Smith, and T. Mitamura. 2007. What is the Jeopardy model? a quasi-synchronous grammar for QA. In *Proc. of EMNLP-CoNLL*.

Jason Weston, Antoine Bordes, Sumit Chopra, and Tomas Mikolov. 2015. Towards ai-complete question answering: A set of prerequisite toy tasks. April.

X. Yao, J. Berant, and B. Van Durme. 2014. Freebase QA: Information extraction or semantic parsing? In *Workshop on Semantic Parsing*.