Abstract

Machine comprehension tests the system’s ability to understand a piece of text through a reading comprehension task. For this task, we propose an approach using the Abstract Meaning Representation (AMR) formalism. We construct meaning representation graphs for the given text and for each question-answer pair by merging the AMRs of comprising sentences using cross-sentential phenomena such as coreference and rhetorical structures. Then, we reduce machine comprehension to a graph containment problem. We posit that there is a latent mapping of the question-answer meaning representation graph onto the text meaning representation graph that explains the answer. We present a unified max-margin framework that learns to find this mapping (given a corpus of texts and question-answer pairs), and uses what it learns to answer questions on novel texts. We show that this approach leads to state of the art results on the task.

1 Introduction

Learning to efficiently represent and reason with natural language is a fundamental yet long-standing goal in NLP. This has led to a series of efforts in broad-coverage semantic representation (or “sembanking”). Recently, AMR, a new semantic representation in standard neo-Davidsonian (Davidson, 1969; Parsons, 1990) framework has been proposed. AMRs are rooted, labeled graphs which incorporate PropBank style semantic roles, within-sentence coreference, named entities and the notion of types, modality, negation, quantification, etc. in one framework.

In this paper, we describe an approach to use AMR for the task of machine comprehension. Machine comprehension (Richardson et al., 2013) evaluates a machine’s understanding by posing a series of multiple choice reading comprehension tests. The tests are unique as the answer to each question can be found only in its associated texts, requiring us to go beyond simple lexical solutions. Our approach models machine comprehension as an extension to textual entailment, learning to output an answer that is best entailed by the passage. It works in two stages. First, we construct a meaning representation graph for the entire passage (§2.1) from the AMR graphs of comprising sentences. To do this, we account for cross-sentence linguistic phenomena such as entity and...
event coreference, and rhetorical structures. A similar meaning representation graph is also constructed for each question-answer pair. Once we have these graphs, the comprehension task henceforth can be reduced to a graph containment problem. We posit that there is a latent subgraph of the text meaning representation graph (called snippet graph) and a latent alignment of the question-answer graph onto this snippet graph that entails the answer (see Figure 1 for an example). Then, we propose a unified max-margin approach (§2.2) that jointly learns the latent structure (subgraph selection and alignment) and the QA model. We evaluate our approach on the MCTest dataset and achieve competitive or better results than a number of previous proposals for this task.

2 The Approach

2.1 The Meaning Representation Graphs

We construct the meaning representation graph using individual sentences AMR graphs and merging identical concepts (using entity and event coreference). First, for each sentence AMR, we merge nodes corresponding to multi-word expressions and nodes headed by a date entity (“date-entity”), or a named entity (“name”) or a person entity (“person”). For example, the hypothesis meaning representation graph in Figure 1 was achieved by merging the AMR parse shown in Figure 2.

Next, we select the subset of sentence AMRs corresponding to sentences needed to answer the question. This step uses cross-sentential phenomena such as rhetorical structure\(^1\) and entities/event coreference. The coreferent entities/event mentions are further merged into one node resulting in a graph called the relevant text snippet graph. A similar process is also performed with the hypothesis sentences (generated by combining the question and answer candidate) as shown in Figure 1.

2.2 Max-Margin Solution

For each question \(q_i \in Q\), let \(t_i\) be the corresponding passage text and \(A_i = \{a_{i1}, \ldots, a_{im}\}\) be the set of candidate answers to the question. Our solution casts the machine comprehension task as a textual entailment task by converting each question-answer candidate pair \((q_i, a_{ij})\) into a hypothesis statement \(h_{ij}\). We use the question matching/rewriting rules described in Cucerzan and Agichtein (2005) to get the hypothesis statements. For each question \(q_i\), the machine comprehension task reduces to picking the hypothesis \(h_i\) that has the highest likelihood of being entailed by the text \(t_i\) among the set of hypotheses \(h_i = \{h_{i1}, \ldots, h_{im}\}\) generated for the question \(q_i\). Let \(h^*_i \in h_i\) be the hypothesis corresponding to the correct answer.

As described, we use subgraph matching to help us model the inference. We assume that the selection of sentences to generate the relevant text snippet graph and the mapping of the hypothesis meaning representation graph onto the passage meaning representation graph is latent and infer it jointly along with the answer. We treat it as a structured prediction problem of ranking the hypothesis set \(h_i\) such that the correct hypothesis \(h^*_i\) is at the top of this ranking. We learn a scoring function \(S_w(t, h, z)\) with parameter \(w\) such that the score of the correct hypothesis \(h^*_i\) and corresponding best latent structure \(z^*_i\) is higher than the score of the other hypotheses and corresponding best latent structures. In a max-margin fashion, we want that \(S_w(t, h^*_i, z^*_i) > S(t, h_{ij}, z_{ij}) + 1 - \xi_i\) for all \(h_{ij} \in h_i \setminus h^*_i\) for some slack \(\xi_i\). Writing the relaxed max margin formulation:

\[
\min_{\|w\|} \frac{1}{2} \|w\|^2 + C \max_{h_i, z_i} \max_{h^*_i, z^*_i} \sum_{t, h^*_i, h_{ij}, z^*_i} \sum_{h^*_i, h_{ij}, z^*_i} S_w(t, h^*_i, z^*_i) + \Delta(h^*_i, h_{ij}) - C \sum_i S_w(t, h^*_i, z^*_i)
\]

We use 0-1 cost, i.e. \(\Delta(h^*_i, h_{ij}) = 1(h^*_i \neq h_{ij})\). If the scoring function is convex then this objective is in concave-convex form and hence can be solved by the concave-convex programming procedure (CCCP) (Yuille and Rangarajan, 2003). We assume the scoring function to be linear: \(S_w(t, h, z) = w^T \psi(t, h, z)\). Here,
\(\psi(t, h, z)\) is a feature map discussed later. The CCP algorithm essentially alternates between solving for \(z^*_t\), \(z_{ij} \forall i, j\) s.t. \(h_{ij} \in \text{h}_i \setminus \text{h}_j^*\) and \(w\) to achieve a local minima. In the absence of information regarding the latent structure \(z\) we pick the structure that gives the best score for a given hypothesis i.e. \(\arg\max_z S_w(t, h, z)\).

### 2.3 Scoring Function and Inference

Now, we define the scoring function \(S_w(t, h, z)\). Let the hypothesis meaning representation graph be \(G' = (V', E')\). Our latent structure \(z\) decomposes into the selection \((z_s)\) of relevant sentences that lead to the text snippet graph \(G\), and the mapping \((z_m)\) of every node and edge in \(G'\) onto \(G\). We define the score such that it factorizes over the nodes and edges in \(G'\). The weight vector \(w\) also has three components \(w_s, w_r,\) and \(w_e\) corresponding to the relevant sentences selection, node matches and edge matches respectively. An edge in the graph is represented as a triple \((\{v^1, r, v^2\}\) consisting of the endpoint vertices and relation \(r\).

\[
S_w(t, h, z) = w_s^T f(G', G, t, h, z_s) + \sum_{v' \in V'} w_r^T f(v', z_m(v')) + \sum_{e' \in E'} w_e^T f(e', z_m(e'))
\]

Here, \(t\) is the text corresponding to the hypothesis \(h\), and \(f\) are parts of the feature map \(\psi\) to be described later. \(z(v')\) maps a node \(v' \in V'\) to a node in \(V\). Similarly, \(z(e')\) maps an edge \(e' \in E'\) to an edge in \(E\).

Next, we describe the inference procedure i.e. how to select the structure that gives the best score for a given hypothesis. The inference is performed in two steps: The first step selects the relevant sentences from the text. This is done by simply maximizing the first part of the score: \(z_s = \arg\max_z w_s^T f(G', G, t, h, z_s)\). Here, we only consider subsets of 1, 2 and 3 sentences as most questions can be answered by 3 sentences in the passage. The second step is formulated as an integer linear program by rewriting the scoring function. The ILP objective is:

\[
\sum_{v' \in V'} \sum_{v \in V} z_{v', v} w_s^T f(v', v) + \sum_{e' \in E'} \sum_{e \in E} z_{e', e} w_e^T f(e', e)
\]

Here, with some abuse of notation, \(z_{v', v}\) and \(z_{e', e}\) are binary integers such that \(z_{v', v} = 1\) iff \(z\) maps \(v'\) onto \(v\) else \(z_{v', v} = 0\). Similarly, \(z_{e', e} = 1\) iff \(z\) maps \(e'\) onto \(e\) else \(z_{e', e} = 0\). Additionally, we have the following constrains to our ILP:

- Each node \(v' \in V'\) (or each edge \(e' \in E'\)) is mapped to exactly one node \(v \in V\) (or one edge \(e \in E\)). Hence: \(\sum_{v \in V} z_{v', v} = 1 \ \forall v'\) and \(\sum_{e \in E} z_{e', e} = 1 \ \forall e'\)
- If an edge \(e' \in E'\) is mapped to an edge \(e \in E\), then vertices \((v^1, v^2)\) that form the end points of \(e'\) must also be aligned to vertices \((v'^1, v'^2)\) that form the end points of \(e\). Here, we note that AMR parses also have inverse relations such as “arg0-of”. Hence, we resolve this with a slight modification. If neither or both relations (corresponding to edges \(e'\) and \(e\)) are inverse relations (case 1), we enforce that \(v^1 \) align with \(v'^1\) and \(v^2\) with \(v'^2\). If exactly one of the relations is an inverse relation (case 2), we enforce that \(v'^1\) align with \(v^2\) and \(v'^2\) align with \(v^1\). Hence, we introduce the following constraints:

\[
\begin{align*}
&z_{v'e} \leq z_{v^1, v^1} \\
&z_{v'e} \leq z_{v^2, v^2}
\end{align*}
\]

\[
\begin{align*}
&z_{e'e} \leq z_{v^1, v^1} \\
&z_{e'e} \leq z_{v^2, v^2}
\end{align*}
\]

These databases contain paraphrase rules of the form string\(_1\) \(\rightarrow\) string\(_2\). ParaPara rules were extracted through bilingual pivoting and DIRT rules were extracted using the distributional hypothesis. Whenever we
have a substring in the text snippet that can be transformed into another using any of these two databases, we keep match features for the substring with a higher score (according to the current w) and ignore the other substring. Finally, we also have features corresponding to the RST (Mann and Thompson, 1988) links to enable inference across sentences. RST tells us that sentences with discourse relations are related to each other and can help us answer certain kinds of questions (Jansen et al., 2014). For example, the “cause” relation between sentences in the text can often give cues that can help us answer “why” or “how” questions. Hence, we have additional features - conjunction of the rhetorical structure label from a RST parser and the question word as well.

The second part corresponds to node matches. Here, we have features for (a) Surface-form match (Edit-distance), and (b) Semantic word match (cosine similarity using Senna word vectors (Collobert et al., 2011) and “Antonymy” “Class-Inclusion” or ‘Is-A’ relations using Wordnet).

The third part corresponds to edge matches. Let the edges be \( e = (v^1, r, v^2) \) and \( e' = (v'^1, r', v'^2) \) for notational convenience. Here, we introduce two features based on the relations - indicator that the two relations are the same or inverse of each other, indicator that the two relations are in the same relation category – categories as described in Banerjee et al. (2013). Then, we introduce a number of features based on distributional representation of the node pairs. We compute three vertex vector compositions (sum, difference and product) for the nodes for each edge proposed in recent representation learning literature in NLP (Mitchell and Lapata, 2008) Mikolov et al., 2013 i.e. \( v^1 \circ v^2 \) and \( v'^1 \circ v'^2 \) for \( \circ \in \{+, -, \times\} \). Then, we compute the cosine similarities of the resulting compositions producing three features. Finally we introduce features based on the structured distributional semantic representation (Erk and Padó, 2008 Baroni and Lenci, 2010; Goyal et al., 2013) which takes the relations into account while performing the composition. Here, we use a large text corpora (in our experiments, the English Wikipedia) and construct a representation matrix \( M^{(r)} \subset V \times V \) for every relation \( r \) (\( V \) is the vocabulary) where, the \( ij^{th} \) element \( M^{(r)}_{ij} \) has the value \( \log(1+x) \) where \( x \) is the frequency for the \( i^{th} \) and \( j^{th} \) vocabulary items being in relation \( r \) in the corpora. This allows us to compose the node and relation representations and compare them. Here we compute the cosine similarity of the compositions \( (v^1)^TM^{(r)}v^2 \) and \( (v'^1)^TM^{(r)}v'^2 \), the compositions \( M^{(r)}v^2 \) and \( M^{(r)}v'^2 \) and their respective sums \( (v^1)^TM^{(r)} + M^{(r)}v^2 \) and \( (v'^1)^TM^{(r)} + M^{(r)}v'^2 \) to get three more features.

### 2.5 Negation and Multi-task Learning

Next, we borrow two ideas from Sachan et al. (2015) namely, negation and multi-task learning, treating different question types in the machine comprehension setup as different tasks.

Handling negation is important for our model as facts align well with their negated versions. We use a simple heuristic. During training, if we detect negation (using a set of simple rules that test for presence of negation words (“not”, “n’t”, etc.)), we flip the corresponding constraint, now requiring that the correct hypothesis to be ranked below all the incorrect ones. During test phase if we detect negation, we predict the answer corresponding to the hypothesis with the lowest score.

QA systems often include a question classification component that divides the questions into semantic categories based on the type of the question or answers expected. This allows the model to learn question type specific parameters when needed. We experiment with three task classifications proposed by Sachan et al. (2015). First is QCClassification, which classifies the question, based on the question word (what, why, what, etc.). Next is the QAClassification scheme, which classifies questions into different semantic classes based on the possible semantic types of the answers sought. The third scheme, TaskClassification classifies the questions into one of 20 subtasks for Machine Comprehension proposed in Weston et al. (2015). We point the reader to Sachan et al. (2015) for details on the multi-task model.

### 3 Experiments

#### Datasets:

We use MCTest-500 dataset (Richardson et al., 2013), a freely available set of 500 stories (300 train, 50 dev and 150 test) and associated questions to evaluate our model. Each story in MCTest has four multiple-choice questions, each with four answer choices. Each question has exactly one correct answer. Each question is also annotated as ‘single’ or ‘multiple’. The questions annotated ‘single’ require just one sentence in the passage to answer them. For ‘multiple’ questions
it should not be possible to find the answer to the question with just one sentence of the passage. In a sense, ‘multiple’ questions are harder than ‘single’ questions as they require more complex inference. We will present the results breakdown for ‘single’ or ‘multiple’ category questions as well.

**Baselines:** We compare our approach to the following baselines: (1-3) The first three baselines are taken from Richardson et al. (2013). SW and SW+D use a sliding window and match a bag of words constructed from the question and the candidate answer to the text. RTE uses textual entailment by selecting the hypothesis that has the highest likelihood of being entailed by the passage. (4) LEX++, taken from Smith et al. (2015) is another lexical matching method that takes into account multiple context windows, question types and coreference. (5) JACANA uses an off the shelf aligner and aligns the hypothesis statement with the passage. (6-7) LSTM and QANTA, taken from Sachan et al. (2015), use neural networks (LSTMs and Recursive NNs, respectively). (8) ATTENTION, taken from Yin et al. (2016), uses an attention-based convolutional neural network. (9) DISCOURSE, taken from Narasimhan and Barzilay (2015), proposes a discourse based model. (10-14) LSSVM, LSSVM+Negation, LSSVM+Negation (MultiTask), taken from Sachan et al. (2015) are all discourse aware latent structural svm models. LSSVM+Negation accounts for negation. LSSVM+Negation+MTL further incorporates multi-task learning based on question types. Here, we have three variants of multitask learners based on the three question classification strategies. (15) Finally, SYN+FRM+SEM, taken from Wang et al. (2015) proposes a framework with features based on syntax, frame semantics, coreference and word embeddings.

**Results:** We compare our AMR subgraph containment approach, where we consider our modifications for negation and multi-task learning as well in Table 1. We can observe that our models have a comparable performance to all the baselines including the neural network approaches and all previous approaches proposed for this task. Further, when we incorporate multi-task learning, our approach achieves the state of the art. Also, our approaches have a considerable improvement over the baselines for ‘multiple’ questions. This shows the benefit of our latent structure that allows us to combine evidence from multiple sentences. The negation heuristic helps significantly, especially for ‘single’ questions (majority of negation cases in the MCTest dataset are for the “single” questions). The multi-task method which performs a classification based on the subtasks for machine comprehension defined in Weston et al. (2015) does better than QAClassification that learns the question answer classification. QAClassification in turn performs better than QClassification that learns the question classification only.

These results, together, provide validation for our approach of subgraph matching over meaning representation graphs, and the incorporation of negation and multi-task learning.

### Table 1: Comparison of variations of our method against several baselines on the MCTest-500 dataset. The table shows accuracy on the test set of MCTest-500. All differences between the baselines (except SYN+FRM+SEM) and our approaches, and the improvements due to negation and multi-task learning are significant ($p < 0.05$) using the two-tailed paired T-test.

<table>
<thead>
<tr>
<th>AMR</th>
<th>Single</th>
<th>Multiple</th>
<th>All</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subgraph</td>
<td>67.28</td>
<td>63.24</td>
<td>66.16</td>
</tr>
<tr>
<td>Subgraph+Negation</td>
<td>69.48</td>
<td>66.46</td>
<td>67.83</td>
</tr>
<tr>
<td>QClassification</td>
<td>70.59</td>
<td>67.99</td>
<td>69.17</td>
</tr>
<tr>
<td>QAClassification</td>
<td>71.32</td>
<td>68.29</td>
<td>69.67</td>
</tr>
<tr>
<td>TaskClassification</td>
<td>72.05</td>
<td>68.90</td>
<td>70.33</td>
</tr>
</tbody>
</table>

| SW                                 | 54.56  | 54.04    | 54.28|
| SW+D                               | 62.99  | 58.00    | 60.26|
| RTE                                | 69.85  | 42.71    | 55.01|
| LEX++                              | 69.12  | 63.34    | 65.96|
| JACANA Aligner                     | 58.82  | 54.88    | 56.67|
| LSTM                               | 62.13  | 58.84    | 60.33|
| QANTA                              | 61.23  | 59.45    | 61.10|
| ATTENTION                          | 54.20  | 51.70    | 52.90|
| DISCOURSE                          | 68.38  | 59.90    | 63.75|
| LSSVM                              | 61.12  | 66.67    | 64.15|
| LSSVM+Negation                     | 63.24  | 66.15    | 64.83|
| QClassification                    | 64.34  | 66.46    | 65.50|
| QAClassification                   | 66.18  | 67.37    | 66.83|
| TaskClassification                 | 67.65  | 67.99    | 67.83|
| SYN+FRM+SEM                       | 72.05  | 67.94    | 69.94|

3We tune the SVM parameter $C$ on the dev set. We use Stanford CoreNLP, HILDA parser (Feng and Hirst, 2014) and JAMR (Flanigan et al., 2014) for preprocessing.

### 4 Conclusion

We proposed a solution for reading comprehension tests using AMR. Our solution builds intermediate meaning representations for passage and question-answers. Then it poses the comprehension task as a subgraph matching task by learning latent alignments from one meaning representation to another. Our approach achieves competitive or better performance than other approaches proposed for this task. Incorporation of negation and multi-task learning leads to further improvements establishing it as the new state-of-the-art.
References


