Learning Semantic Script Knowledge with Event Embeddings

Modi, A.; Titov, I.

Publication date
2014

Document Version
Accepted author manuscript

Published in
Workshop proceedings: papers accepted to the International Conference on Learning Representations (ICLR) 2014

Citation for published version (APA):
Learning Semantic Script Knowledge with Event Embeddings

Ashutosh Modi
Saarland University, Saarbrücken, Germany

Ivan Titov
University of Amsterdam, Amsterdam, the Netherlands

Abstract

Induction of common sense knowledge about prototypical sequences of events has recently received much attention (e.g., (Chambers & Jurafsky, 2008; Regneri et al., 2010)). Instead of inducing this knowledge in the form of graphs, as in much of the previous work, in our method, distributed representations of event realizations are computed based on distributed representations of predicates and their arguments, and then these representations are used to predict prototypical event orderings. The parameters of the compositional process for computing the event representations and the ranking component of the model are jointly estimated from texts. We show that this approach results in a substantial boost in ordering performance with respect to previous methods.

1. Introduction

It is generally believed that natural language understanding systems would benefit from incorporating common-sense knowledge about prototypical sequences of events and their participants. Early work focused on structured representations of this knowledge (called scripts (Schank & Abelson, 1977)) and manual construction of script knowledge bases. However, these approaches do not scale to complex domains (Mueller, 1998; Gordon, 2001). More recently, automatic induction of script knowledge from text have started to attract attention: these methods exploit either natural texts (Chambers & Jurafsky, 2008; 2009) or crowdsourced data (Regneri et al., 2010), and, consequently, do not require expensive expert annotation. Given a text corpus, they extract structured representations (i.e. graphs), for example chains (Chambers & Jurafsky, 2008) or more general directed acyclic graphs (Regneri et al., 2010). These graphs are scenario-specific, nodes in them correspond to events (and associated with sets of potential event mentions) and arcs encode the temporal precedence relation. These graphs can then be used to inform NLP applications (e.g., question answering) by providing information whether one event is likely to precede or succeed another.

In this work we advocate constructing a statistical model which is capable of “answering” at least some of the questions these graphs can be used to answer, but doing this without explicitly representing the knowledge as a graph. In our method, the distributed representations (i.e. vectors of real numbers) of event realizations are computed based on distributed representations of predicates and their arguments, and then the event representations are used in a ranker to predict the expected ordering of events. Both the parameters of the compositional process for computing the event representation and the ranking component of the model are estimated from data.

In order to get an intuition why the embedding approach may be attractive, consider a situation where a prototypical ordering of events the bus disembarked passengers and the bus drove away needs to be predicted. An approach based on frequency of predicate pairs (Chambers & Jurafsky, 2008), is unlikely to make a right prediction as driving usually precedes disembarking. Similarly, an approach which treats the whole predicate-argument structure as an atomic unit (Regneri et al., 2010) will probably fail as well, as such a sparse model is unlikely to be effectively learnable even from large amounts of data. However, our embedding method would be expected to capture relevant features of the verb frames, namely, the transitive use for the predicate disembark and the effect of the particle away, and these features will then be used by the ranking component to make the correct prediction.

In previous work on learning inference rules (Berant et al., 2011), it has been shown that enforcing transitivity constraints on the inference rules results in significantly improved performance. The same is true for the event order-
Learning Semantic Script Knowledge with Event Embeddings

2. Model

In this section we describe the model we use for computing event representations as well as the ranking component of our model.

2.1. Event Representation

Learning and exploiting distributed word representations (i.e. vectors of real values, also known as embeddings) have been shown to be beneficial in many NLP applications (Bengio et al., 2001; Turian et al., 2010; Collobert et al., 2011). These representations encode semantic and syntactic properties of a word, and are normally learned in the language modeling setting (i.e. learned to be predictive of local word context), though they can also be specialized by learning in the context of other NLP applications such as PoS tagging or semantic role labeling (Collobert et al., 2011). More recently, the area of distributional compositional semantics have started to emerge (Baroni & Zamperelli, 2011; Socher et al., 2012), they focus on inducing representations of phrases by learning a compositional model. Such a model would compute a representation of a phrase by starting with embeddings of individual words in the phrase, often this composition process is recursive and guided by some form of syntactic structure.

In our work, we use a simple compositional model for representing semantics of a verb frame (i.e. the predicate and its arguments). The model is shown in Figure 1. Each word \(w_i\) in the vocabulary is mapped to a real vector based on the corresponding lemma (the embedding function \(C\)). The hidden layer is computed by summing linearly transformed predicate and argument embeddings and passing it through the logistic sigmoid function.

Algorithm 1 Learning Algorithm

Notation

- \(w\) : ranking weight vector
- \(E_k\) : \(k\)th sequence of events in temporal order
- \(t_k\) : array of model scores for events in \(E_k\)
- \(\gamma\) : fixed global margin for ranking

\[
\text{LearnWeights}() \\
\text{for epoch} = 1 \text{ to } T \\
\quad \text{for } k = 1 \text{ to } K \\
\quad \quad \text{[over event sequences]} \\
\quad \quad \text{for } i = 1 \text{ to } |E_k| \\
\quad \quad \quad \text{[over events in the seq]} \\
\quad \quad \quad \text{Compute embedding } x_{e_i} \text{ for event } e_i \\
\quad \quad \quad \text{Calculate score } s_{e_i} = w^T x_{e_i} \\
\quad \quad \text{end for} \\
\quad \text{Collect scores in } t_k = [s_{e_1}, \ldots, s_{e_l}, \ldots] \\
\quad \text{error} = \text{RankingError}(t_k) \\
\quad \text{back-propagate error} \\
\quad \text{update all embedding parameters and } w \\
\quad \text{end for} \\
\text{end for} \\
\text{RankingError}(t_k) \\
\quad \text{err} = 0 \\
\quad \text{for rank} = 1, \ldots, l \\
\quad \quad \text{for rankBefore} = 1, \ldots, rank \\
\quad \quad \quad \text{if } (t_k[\text{rankBefore}] - t_k[\text{rank}]) < \gamma \\
\quad \quad \quad \text{err} = \text{err} + 1 \\
\quad \quad \text{end if} \\
\quad \text{end for} \\
\quad \text{for rankAfter} = \text{rank} + 1, \ldots, l \\
\quad \quad \text{if } (t_k[\text{rank}] - t_k[\text{rankAfter}]) < \gamma \\
\quad \quad \text{err} = \text{err} + 1 \\
\quad \text{end if} \\
\text{end for} \\
\text{return err}
\]

In our approach we implicitly learn the model which satisfies transitivity constraints, without the need for any explicit global optimization on a graph.

The approach is evaluated on crowdsourced dataset of Regneri et al. (2010) and we demonstrate that using our model results in the 13.5% absolute improvement in \(F_1\) on event ordering with respect to their graph induction method (84% vs. 71%).

Figure 1. Computation of an event representation (the bus disembarked passengers).
Learning Semantic Script Knowledge with Event Embeddings

<table>
<thead>
<tr>
<th>Scenario</th>
<th>Precision (%)</th>
<th>Recall (%)</th>
<th>F1 (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>BL</td>
<td>EEverb</td>
<td>MSA</td>
</tr>
<tr>
<td>Bus</td>
<td>70.1</td>
<td>81.9</td>
<td>80.0</td>
</tr>
<tr>
<td>Coffee</td>
<td>70.1</td>
<td>73.7</td>
<td>70.0</td>
</tr>
<tr>
<td>Fastfood</td>
<td>69.9</td>
<td>81.0</td>
<td>53.0</td>
</tr>
<tr>
<td>Ret. Food</td>
<td>74.0</td>
<td>94.1</td>
<td>48.0</td>
</tr>
<tr>
<td>Iron</td>
<td>73.4</td>
<td>80.1</td>
<td>78.0</td>
</tr>
<tr>
<td>Microwave</td>
<td>72.6</td>
<td>79.2</td>
<td>47.0</td>
</tr>
<tr>
<td>Scr. Eggs</td>
<td>72.7</td>
<td>71.4</td>
<td>67.0</td>
</tr>
<tr>
<td>Shower</td>
<td>62.2</td>
<td>76.2</td>
<td>48.0</td>
</tr>
<tr>
<td>Telephone</td>
<td>67.6</td>
<td>87.8</td>
<td>83.0</td>
</tr>
<tr>
<td>Vending</td>
<td>66.4</td>
<td>78.3</td>
<td>84.0</td>
</tr>
<tr>
<td>Average</td>
<td>69.9</td>
<td>81.3</td>
<td>65.8</td>
</tr>
</tbody>
</table>

Table 1. Results on the crowdsourced data for the verb-frequency baseline (BL), the verb-only embedding model (EEverb), Regneri et al. (2010) (MSA), Frermann et al. (2014)(BS) and the full model (EE).

2When we kept the word representations fixed to the SENNA embeddings (Collobert et al., 2011).

3. Experiments

We evaluate our approach on crowdsourced data collected for script induction by Regneri et al. (2010), though, in principle, the method is applicable in arguably more general setting of Chambers & Jurafsky (2008).

3.1. Data and task

Regneri et al. (2010) collected short textual descriptions (called event sequence descriptions, ESDs) of various types of human activities (e.g., going to a restaurant, ironing clothes) using crowdsourcing (Amazon Mechanical Turk), this dataset was also complemented by descriptions provided in the OMICS corpus (Gupta & Kochenderfer, 2004).

The datasets are fairly small, containing 30 ESDs per activity type in average (we will refer to different activities as scenarios), but the collection can easily be extended given the low cost of crowdsourcing. The ESDs are written in a bullet-point style and the annotators were asked to follow the temporal order in writing. Consider an example ESD for the scenario prepare coffee:

{go to coffee maker} → {fill water in coffee maker} → {place the filter in holder} → {place coffee in filter} → {place holder in coffee maker} → {turn on coffee maker}

Though individual ESDs may seem simple, the learning task is challenging because of the limited amount of training data, variability in the used vocabulary, optionality of events (e.g., going to the coffee machine may not be mentioned in an ESD), different granularity of events and variability in the ordering (e.g., coffee may be put in a filter before placing it in a coffee maker).

2When we kept the word representations fixed to the SENNA embeddings and learned only matrices T, R and A, we obtained similar results (0.3% difference in the average F1 score).
Unlike our work, Regneri et al. (2010) relies on WordNet to provide extra signal when using the Multiple Sequence Alignment (MSA) algorithm. As in their work, each description was preprocessed to extract a predicate and heads of argument noun phrases to be used in the model.

The methods are evaluated on human annotated scenario-specific tests: the goal is to classify event pairs as appearing in a given stereotypical order or not (Regneri et al., 2010). The model was estimated as explained in Section 2.2 with the order of events in ESDs treated as gold standard. We used 4 held-out scenarios to choose model parameters, no scenario-specific tuning was performed, and the 10 test scripts were not used to perform model selection.

When testing, we predicted that the event pair \((e_1, e_2)\) is in the stereotypical order \((e_1 \prec e_2)\) if the ranking score for \(e_1\) exceeded the ranking score for \(e_2\).

3.2. Results and discussion

In our experiments, we compared our event embedding model (EE) against three baseline systems (BL, MSA) and BSMSA is the system of Regneri et al. (2010). BS is a hierarchical bayesian system of Frermann et al. (2014). BL chooses the order of events based on the preferred order of the corresponding verbs in the training set: \((e_1, e_2)\) is predicted to be in the stereotypical order if the number of times the corresponding verbs \(v_1\) and \(v_2\) appear in this order in the training ESDs exceeds the number of times they appear in the opposite order (not necessary at adjacent positions); a coin is tossed to break ties (or if \(v_1\) and \(v_2\) are the same verb).

We also compare to the version of our model which uses only verbs \((\text{EE}_{\text{verbs}})\). Note that \(\text{EE}_{\text{verbs}}\) is conceptually very similar to BL, as it essentially induces an ordering over verbs. However, this ordering can benefit from the implicit transitivity assumption used in \(\text{EE}_{\text{verbs}}\) (and EE), as we discussed in the introduction. The results are presented in Table 1.

The first observation is that the full model improves substantially over the baseline and the previous methods (MSA and BS) (13.5% and 6.5% improvement over MSA and BS respectively in F1), this improvement is largely due to an increase in the recall but the precision is not negatively affected. We also observe a substantial improvement in all metrics from using transitivity, as seen by comparing the results of BL and \(\text{EE}_{\text{verbs}}\) (11.3% improvement in F1). This simple approach already outperforms the pipelined MSA system. These results seem to support our hypothesis in the introduction that inducing graph representations from scripts may not be an optimal strategy from the practical perspective.

References


