Learning Semantic Script Knowledge with Event Embeddings

Modi, A.; Titov, I.

Publication date
2014

Document Version
Author accepted manuscript

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

Learning Semantic Script Knowledge with Event Embeddings

Ashutosh Modi ASHUTOSH@COLI.UNI-SB.DE
Saarland University, Saarbrücken, Germany

Ivan Titov TITOV@UVA.NL
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-

Accepted at the workshop track of International Conference on Learning Representations (ICLR), 2014

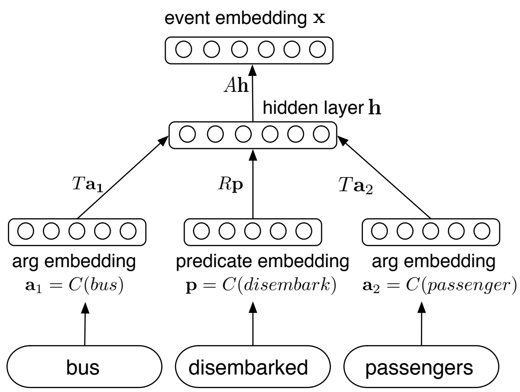

Figure 1. Computation of an event representation (*the bus disembarked passengers*).

ing task, as scripts have largely linear structure, and observing that $a \prec b$ and $b \prec c$ is likely to imply $a \prec c$. Interestingly, 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 $F1$ on event ordering with respect to their graph induction method (84% vs. 71%).

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 & Zamparelli, 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.

Algorithm 1 Learning Algorithm

Notation
$\mathbf{w}$: ranking weight vector
E_k : k^{th} sequence of events in temporal order
t_k : array of model scores for events in E_k
γ : fixed global margin for ranking

LearnWeights()
for $epoch = 1$ to T
  for $k = 1$ to K [over event sequences]
    for $i = 1$ to $|E_k|$ [over events in the seq]
      Compute embedding x_{e_i} for event e_i
      Calculate score $s_{e_i} = \mathbf{w}^T x_{e_i}$
    end for
    Collect scores in $t_k = [s_{e_1}, \ldots, s_{e_i}, \ldots]$
    $error = \textbf{RankingError}(t_k)$
    back-propagate $error$
    update all embedding parameters and $\mathbf{w}$
  end for
end for

$\textbf{RankingError}(t_k)$
$err = 0$
for $rank = 1, \ldots, l$
  for $rankBefore = 1, \ldots, rank$
    if $(t_k[rankBefore] - t_k[rank]) < \gamma$
      $err = err + 1$
    end if
  end for
  for $rankAfter = rank + 1, \ldots, l$
    if $(t_k[rank] - t_k[rankAfter]) < \gamma$
      $err = err + 1$
    end if
  end for
end for
return err

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.[1] We use different transformation matrices for arguments and predicates, T and R, respectively. The event representation x is then obtained by applying another linear transform (matrix A) followed by another application of the sigmoid function.

These event representations are learned in the context of

[1]Only syntactic heads of arguments are used in this work. If an argument is *a coffee maker*, we will use only the word *maker*.

Scenario	Precision (%)					Recall (%)					F1 (%)				
	BL	EE_{verb}	MSA	BS	EE	BL	EE_{verb}	MSA	BS	EE	BL	EE_{verb}	MSA	BS	EE
Bus	70.1	81.9	80.0	76.0	85.1	71.3	75.8	80.0	76.0	91.9	70.7	78.8	80.0	76.0	88.4
Coffee	70.1	73.7	70.0	68.0	69.5	72.6	75.1	78.0	57.0	71.0	71.3	74.4	74.0	62.0	70.2
Fastfood	69.9	81.0	53.0	97.0	90.0	65.1	79.1	81.0	65.0	87.9	67.4	80.0	64.0	78.0	88.9
Ret. Food	74.0	94.1	48.0	87.0	92.4	68.6	91.4	75.0	72.0	89.7	71.0	92.8	58.0	79.0	91.0
Iron	73.4	80.1	78.0	87.0	86.9	67.3	69.8	72.0	69.0	80.2	70.2	69.8	75.0	77.0	83.4
Microwave	72.6	79.2	47.0	91.0	82.9	63.4	62.8	83.0	74.0	90.3	67.7	70.0	60.0	82.0	86.4
Scr. Eggs	72.7	71.4	67.0	77.0	80.7	68.0	67.7	64.0	59.0	76.9	70.3	69.5	66.0	67.0	78.7
Shower	62.2	76.2	48.0	85.0	80.0	62.5	80.0	82.0	84.0	84.3	62.3	78.1	61.0	85.0	82.1
Telephone	67.6	87.8	83.0	92.0	87.5	62.8	87.9	86.0	87.0	89.0	65.1	87.8	84.0	89.0	88.2
Vending	66.4	87.3	84.0	90.0	84.2	60.6	87.6	85.0	74.0	81.9	63.3	84.9	84.0	81.0	88.2
Average	69.9	81.3	65.8	85.0	**83.9**	66.2	77.2	78.6	71.7	**84.3**	68.0	79.1	70.6	77.6	**84.1**

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

event ranking: the transformation parameters as well as representations of words are forced to be predictive of the temporal order of events. However, one important characteristic of neural network embeddings is that they can be induced in a multitasking scenario, and consequently can be learned to be predictive of different types of contexts providing a general framework for inducing different aspects of (semantic) properties of events, as well as exploiting the same representations in different applications.

2.2. Learning to Order

The task of learning stereotyped order of events naturally corresponds to the standard ranking setting. Here, we assume that we are provided with sequences of events, and our goal is to capture this order. We discuss how we obtain this learning material in the next section. We learn a linear ranker (characterized by a vector $\boldsymbol{w}$) which takes an event representation and returns a ranking score. Events are then ordered according to the score to yield the model prediction. Note that during the learning stage we estimate not only $\boldsymbol{w}$ but also the event representation parameters, i.e. matrices T, R and A, and the word embedding C. Note that by casting the event ordering task as a global ranking problem we ensure that the model implicitly exploits transitivity of the temporal relation, the property which is crucial for successful learning from finite amount of data, as we argued in the introduction and will confirm in our experiments.

We use an online ranking algorithm based on the Perceptron Rank (PRank, (Crammer & Singer, 2001)), or, more accurately, its large-margin extension. One crucial difference though is that the error is computed not only with respect to $\boldsymbol{w}$ but also propagated back through the structure of the neural network. The learning procedure is sketched in Algorithm 1. Additionally, we use a Gaussian prior on weights, regularizing both the embedding parameters and the vector $\boldsymbol{w}$. We initialize word representations using the SENNA embeddings (Collobert et al., 2011).[2]

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*$\}$ $\rightarrow$ $\{$*fill water in coffee maker*$\}$ $\rightarrow$ $\{$*place the filter in holder*$\}$ $\rightarrow$ $\{$*place coffee in filter*$\}$ $\rightarrow$ $\{$*place holder in coffee maker*$\}$ $\rightarrow$ $\{$*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 a 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).

[2] When 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).[3]

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 *BS*MSA is the system of Regneri et al. (2010). BS is a 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 (EE$_{verbs}$). Note that EE$_{verbs}$ 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 EE$_{verbs}$ (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 EE$_{verb}$ (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.

[3] The unseen event pairs are not coming from the same ESDs making the task harder: the events may not be in any temporal relation. This is also the reason for using the F1 score rather than the accuracy, both in Regneri et al. (2010) and in our work.

References

Baroni, Marco and Zamparelli, Robert. Nouns are vectors, adjectives are matrices: Representing adjective-noun constructions in semantic space. In *Proceedings of EMNLP*, 2011.

Bengio, Yoshua, Ducharme, Réjean, and Vincent, Pascal. A neural probabilistic language model. In *Proceedings of NIPS*, 2001.

Berant, Jonathan, Dagan, Ido, and Goldberger, Jacob. Global learning of typed entailment rules. In *Proceedings of ACL*, 2011.

Chambers, Nathanael and Jurafsky, Dan. Unsupervised learning of narrative schemas and their participants. In *Proceedings of ACL*, 2009.

Chambers, Nathanael and Jurafsky, Daniel. Unsupervised learning of narrative event chains. In *Proceedings of ACL*, 2008.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. Natural language processing (almost) from scratch. *Journal of Machine Learning Research*, 12:2493–2537, 2011.

Crammer, Koby and Singer, Yoram. Pranking with ranking. In *Proceedings of NIPS*, 2001.

Frermann, Lea, Titov, Ivan, and Pinkal, Manfred. A hierarchical bayesian model for unsupervised induction of script knowledge. In *EACL, Gothenberg, Sweden*, 2014.

Gordon, Andrew. Browsing image collections with representations of common-sense activities. *JAIST*, 52(11), 2001.

Gupta, Rakesh and Kochenderfer, Mykel J. Common sense data acquisition for indoor mobile robots. In *Proceedings of AAAI*, 2004.

Mueller, Erik T. *Natural Language Processing with Thought Treasure*. Signiform, 1998.

Regneri, Michaela, Koller, Alexander, and Pinkal, Manfred. Learning script knowledge with web experiments. In *Proceedings of ACL*, 2010.

Schank, R. C and Abelson, R. P. *Scripts, Plans, Goals, and Understanding*. Lawrence Erlbaum Associates, Potomac, Maryland, 1977.

Socher, Richard, Huval, Brody, Manning, Christopher D., and Ng, Andrew Y. Semantic compositionality through recursive matrix-vector spaces. In *Proceedings of EMNLP*, 2012.

Turian, Joseph, Ratinov, Lev, and Bengio, Yoshua. Word representations: A simple and general method for semi-supervised learning. In *Proceedings of ACL*, 2010.