UNIVERSITY OF AMSTERDAM

UvA-DARE (Digital Academic Repository)

Inducing Neural Models of Script Knowledge

Modi, A.; Titov, I.

DOI
10.3115/v1/W14-1606
Publication date
2014
Document Version
Final published version
Published in
CoNNL-2014 : Eighteenth Conference on Computational Natural Language Learning

License
CC BY-NC-SA

Link to publication

Citation for published version (APA):
Modi, A., & Titov, I. (2014). Inducing Neural Models of Script Knowledge. In R. Morante, & SW. Yih (Eds.), *CoNNL-2014 : Eighteenth Conference on Computational Natural Language Learning: proceedings of the conference : June 26-27, 2014, Baltimore, Maryland, USA* (pp. 49-57). The Association for Computational Linguistics. https://doi.org/10.3115/v1/W14-1606

General rights
It is not permitted to download or to forward/distribute the text or part of it without the consent of the author(s) and/or copyright holder(s), other than for strictly personal, individual use, unless the work is under an open content license (like Creative Commons).

Disclaimer/Complaints regulations
If you believe that digital publication of certain material infringes any of your rights or (privacy) interests, please let the Library know, stating your reasons. In case of a legitimate complaint, the Library will make the material inaccessible and/or remove it from the website. Please Ask the Library: https://uba.uva.nl/en/contact, or a letter to: Library of the University of Amsterdam, Secretariat, Singel 425, 1012 WP Amsterdam, The Netherlands. You will be contacted as soon as possible.

UvA-DARE is a service provided by the library of the University of Amsterdam (https://dare.uva.nl)

Download date:05 Jun 2023

Inducing Neural Models of Script Knowledge

Ashutosh Modi
MMCI,
Saarland University, Germany
amodi@mmci.uni-saarland.de

Ivan Titov
ILLC,
University of Amsterdam, Netherlands
titov@uva.nl

Abstract

Induction of common sense knowledge about prototypical sequence of events has recently received much attention (e.g., Chambers and 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. We show that this approach results in a substantial boost in performance on the event ordering task with respect to the previous approaches, both on natural and crowdsourced texts.

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 and 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 and 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 and 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. Note that these graphs encode common-sense knowledge about prototypical ordering of events rather than temporal order of events as described in a given text.

Though representing the script knowledge as graphs is attractive from the human interpretability perspective, it may not be optimal from the application point of view. More specifically, these representations (1) require a model designer to choose an appropriate granularity of event mentions (e.g., whether nodes in the graph should be associated with verbs, or also their arguments); (2) do not provide a mechanism for deciding which scenario applies in a given discourse context and (3) often do not associate confidence levels with information encoded in the graph (e.g., the precedence relation in Regneri et al. (2010)).

Instead of constructing a graph and using it to provide information (e.g., prototypical event ordering) to NLP applications, in this work we advocate for constructing a statistical model which is capable to "answer" 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 prototypical ordering of events. Both the parameters of the compositional process for computing the event representation and the rank-

Proceedings of the Eighteenth Conference on Computational Language Learning, pages 49–57,
Baltimore, Maryland USA, June 26-27 2014. ©2014 Association for Computational Linguistics

ing component of the model are estimated from texts (either relying on unambiguous discourse clues or natural ordering in text). In this way we build on recent research on compositional distributional semantics (Baroni and Zamparelli, 2011; Socher et al., 2012), though our approach specifically focuses on embedding predicate-argument structures rather than arbitrary phrases, and learning these representation to be especially informative for prototypical event ordering.

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 and Jurafsky, 2008) (henceforth CJ08), 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 unlabeled 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 likely to be true for the event ordering 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 learn the model which satisfies transitivity constraints, without the need for any explicit global optimization on a graph. This results in a significant boost of performance when using embeddings of just predicates (i.e. ignoring arguments) with respect to using frequencies of ordered verb pairs, as in CJ08 (76% vs. 61% on the natural data).

Our model is solely focusing on the ordering task, and admittedly does not represent all the information encoded by a script graph structure. For example, it cannot be directly used to predict a missing event given a set of events (the narrative cloze task (Chambers and Jurafsky, 2009)). Nevertheless, we believe that the framework (a probabilistic model using event embeddings as its component) can be extended to represent other aspects of script knowledge by modifying the learning objective, but we leave this for future work. In this paper, we show how our model can be used to predict if two event mentions are likely paraphrases of the same event.

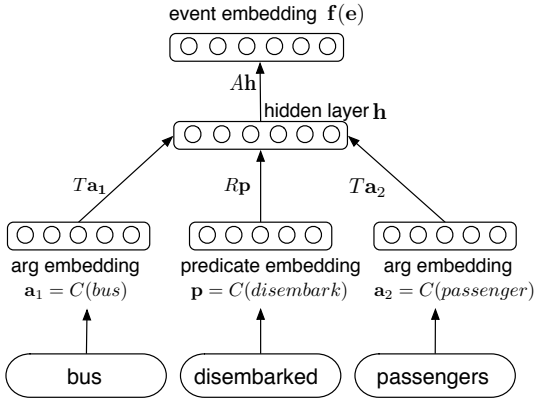

Figure 1: Computation of an event representation for a predicate with two arguments (*the bus disembarked passengers*), an arbitrary number of arguments is supported by our approach.

The approach is evaluated in two set-ups. First, we consider the crowdsourced dataset of Regneri et al. (2010) and demonstrate that using our model results in the 13.5% absolute improvement in $F1$ on event ordering with respect to their graph induction method (84.1% vs. 70.6%). Secondly, we derive an event ordering dataset from the Gigaword corpus, where we also show that the embedding method beats the frequency-based baseline (i.e. reimplementation of the scoring component of CJ08) by 22.8% in accuracy (83.5% vs. 60.7%).

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 and 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.

In our work, we use a simple compositional model for representing semantics of a verb frame e (i.e. the predicate and its arguments). We will refer to such verb frames as events. The model is shown in Figure 1. Each word c_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[1] embeddings and passing it through the logistic sigmoid function. We use different transformation matrices for arguments and predicates, T and R, respectively. The event representation $\boldsymbol{f}(e)$ is then obtained by applying another linear transform (matrix A) followed by another application of the sigmoid function. Another point to note in here is that, as in previous work on script induction, we use lemmas for predicates and specifically filter out any tense markers as our goal is to induce common-sense knowledge about an event rather than properties predictive of temporal order in a specific discourse context.

We leave exploration of more complex and linguistically-motivated models for future work.[2] These event representations are learned in the context of event ranking: the transformation parameters as well as representations of words are forced to be predictive of the temporal order of events. In our experiments, we also consider initialization of predicate and arguments with the SENNA word embeddings (Collobert et al., 2011).

[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*.

[2]In this study, we apply our model in two very different settings, learning from crowdsourced and natural texts. Crowdsourced collections are relatively small and require not over-expressive models.

2.2 Learning to Order

The task of learning stereotyped order of events naturally corresponds to the standard ranking setting. 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 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.

At training time, we assume that each training example k is a list of events $e_1^{(k)}, \ldots, e_{n^{(k)}}^{(k)}$ provided in the stereotypical order (i.e. $e_i^{(k)} \prec e_j^{(k)}$ if $i < j$), $n^{(k)}$ is the length of the list k. We minimize the L_2-regularized ranking hinge loss:

$$\sum_k \sum_{i<j\leq n^{(k)}} \max(0, 1 - \mathbf{w}^T \boldsymbol{f}(e_i^{(k)}; \Theta) + \mathbf{w}^T \boldsymbol{f}(e_j^{(k)}; \Theta)) + \alpha(\|\mathbf{w}\|_2 + \|\Theta\|_2),$$

where $\boldsymbol{f}(e; \Theta)$ is the embedding computed for event e, Θ are all embedding parameters corresponding to elements of the matrices $\{R, C, T, A\}$. We use stochastic gradient descent, gradients w.r.t. Θ are computed using back propagation.

3 Experiments

We evaluate our approach in two different set-ups. First, we induce the model from the crowdsourced data specifically collected for script induction by Regneri et al. (2010), secondly, we consider an arguably more challenging set-up of learning the model from news data (Gigaword (Parker et al., 2011)), in the latter case we use a learning scenario inspired by Chambers and Jurafsky (2008).[3]

[3]Details about downloading the data and models are at: http://www.coli.uni-saarland.de/projects/smile/docs/nmReadme.txt

	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
Return	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
Microw.	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
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
Phone	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).

3.1 Learning from Crowdsourced Data

3.1.1 Data and task

Regneri et al. (2010) collected 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 and Kochenderfer, 2004). The datasets are fairly small, containing 30 ESDs per activity type in average (we will refer to different activities as *scenarios*), but in principle the collection can easily be extended given the low cost of crowdsourcing. The ESDs list events forming the scenario and are written in a bullet-point style. The annotators were asked to follow the prototypical event order in writing. As an example, consider a 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*}

Regneri et al. also automatically extracted predicates and heads of arguments for each event, as needed for their MSA system and our compositional model.

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 the filter before placing it in the coffee maker). 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 stereotypical order or not (Regneri et al., 2010).[4]

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. The selected model used the dimensionality of 10 for event and word embeddings. The initial learning rate and the regularization parameter were set to 0.005 and 1.0, respectively and both parameters were reduced by the factor of 1.2 every epoch the error function went up. We used 2000 epochs of stochastic gradient descent. Dropout (Hinton et al., 2012) with the rate of 20% was used for the hidden layers in all our experiments. 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.

3.1.2 Results and discussion

We evaluated our event embedding model (*EE*) against baseline systems (*BL* , *MSA* and *BS*). MSA is the system of Regneri et al. (2010). BS is a hierarchical Bayesian model by 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

[4]The event pairs are not coming from the same ESDs making the task harder as the events may not be in any temporal relation.

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). This frequency counting method was previously used in CJ08.[5]

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 method (MSA) in F1 (13.5% improvement over MSA and 6.5% improvement over BS). Note also that this improvement is consistent across scenarios: EE outperforms MSA and BS on 9 scenarios out of 10 and 8 out of 10 scenarios in case of BS. Unlike MSA and BS, no external knowledge (i.e. WordNet) was exploited in our method.

We also observe a substantial improvement in all metrics from using transitivity, as seen by comparing the results of BL and EE_{verb} (11% improvement in F1). This simple approach already substantially 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.

We performed additional experiments using the SENNA embeddings (Collobert et al., 2011). Instead of randomly initializing arguments and predicate embeddings (vectors), we initialized them with pre-trained SENNA embeddings. We have not observed any significant boost in performance from using the initialization (average F1 of 84.0% for EE). We attribute the lack of significant improvement to the following three factors. First of all, the SENNA embeddings tend to place antonyms / opposites near each other (e.g., come and go, or end and start). However, 'opposite' predicates appear in very different positions in scripts. Additionally, the SENNA embeddings have dimensionality of 50 which appears to be too high for small crowd-sourced datasets, as it forces us to use larger matrices T and R. Moreover, the SENNA embeddings are estimated from Wikipedia, and the activities in our crowdsourced domain are perhaps underrepresented there.

[5]They scored permutations of several events by summing the logarithmed differences of the frequencies of ordered verb pairs. However, when applied to event pairs, their approach would yield exactly the same prediction rule as BL.

3.1.3 Paraphrasing

Regneri et al. (2010) additionally measure paraphrasing performance of the MSA system by comparing it to human annotation they obtained: a system needs to predict if a pair of event mentions are paraphrases or not. The dataset contains 527 event pairs for the 10 test scenarios. Each pair consists of events from the same scenario. The dataset is fairly balanced containing from 47 to 60 examples per scenario.

This task does not directly map to any statistical inference problem with our model. Instead we use an approach inspired by the interval algebra of Allen (1983).

Our ranking model maps event mentions to positions on the time line (see Figure 2). However, it would be more natural to assume that events are intervals rather than points. In principle, these intervals can be overlapping to encode a rich set of temporal relations (see (Allen, 1983)). However, we make a simplifying assumption that the intervals do not overlap and every real number belongs to an interval. In other words, our goal is to induce a segmentation of the line: event mentions corresponding to the same interval are then regarded as paraphrases.

One natural constraint on this segmentation is the following: if two event mentions are from the same training ESD, they cannot be assigned to the same interval (as events in ESD are not supposed to be paraphrases). In Figure 2 arcs link event mentions from the same ESD. We look for a segmentation which produces the minimal number of segments and satisfy the above constraint for event mentions appearing in training data.

Though inducing intervals given a set of temporal constraints is known to be NP-hard in general (see, e.g., (Golumbic and Shamir, 1993)), for our constraints a simple greedy algorithm finds an optimal solution. We trace the line from the left maintaining a set of event mentions in the current unfinished interval and create a boundary when the constraint is violated; we repeat the process until we processed all mentions. In Figure 2, we would create the first boundary between *arrive in a restaurant* and *order beverages*: *order bev-*

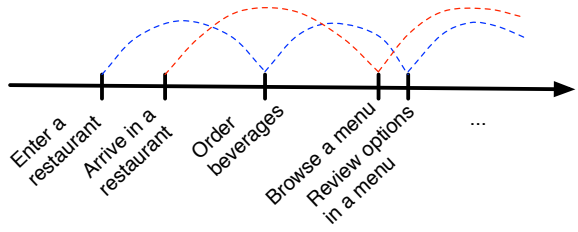

Figure 2: Events on the time line, dotted arcs link events from the same ESD.

Scenario	F1 (%)			
	APBL	MSA	BS	EE
Take bus	53.7	74.0	47.0	63.5
Make coffee	42.1	65.0	52.0	63.5
Order fastfood	37.0	59.0	80.0	62.6
Return food back	64.8	71.0	67.0	81.1
Iron clothes	43.3	67.0	60.0	56.7
Microwave cooking	43.2	75.0	82.0	57.8
Scrambled eggs	57.6	69.0	76.0	53.0
Take shower	42.1	78.0	67.0	55.7
Answer telephone	71.0	89.0	81.0	79.4
Vending machine	56.1	69.0	77.0	69.3
Average	51.1	**71.6**	68.9	64.5

Table 2: Paraphrasing results on the crowdsourced data for Regneri et al. (2010) (MSA), Frermann et al. (2014)(BS) and the all-paraphrase baseline (APBL) and using intervals induced from our model (EE).

erages and *enter a restaurant* are from the same ESD and continuing the interval would violate the constraint. It is not hard to see that this results in an optimal segmentation. First, the segmentation satisfies the constraint by construction. Secondly, the number of segments is minimal as the arcs which caused boundary creation are non-overlapping, each of these arcs needs to be cut and our algorithm cuts each arc exactly once.

This algorithm prefers to introduce a boundary as late as possible. For example, it would introduce a boundary between *browse a menu* and *review options in a menu* even though the corresponding points are very close on the line. We modify the algorithm by moving the boundaries left as long as this move does not result in new constraint violations and increases margin at boundaries. In our example, the boundary would be moved to be between *order beverages* and *browse a menu*, as desired.

The resulting performance is reported in Table 2. We report results of our method, as well as results for MSA, BS and a simple all-paraphrase baseline which predict that all mention pairs in a test set are paraphrases (APBL).[6] We can see that interval induction technique results in a lower F1 than that of MSA or BS. This might be partially due to not using external knowledge (WordNet) in our method.

We performed extra analyses on the development scenario *doorbell*. The analyses revealed that the interval induction approach is not very robust to noise: removing a single noisy ESD results in a dramatic change in the interval structure induced and in a significant increase of F1. Consequently, soft versions of the constraint would be beneficial. Alternatively, event embeddings (i.e. continuous vectors) can be clustered directly. We leave this investigation for future work.

[6]The results for the random baseline are lower: F1 of 40.6% in average.

3.2 Learning from Natural Text

In the second set of experiments we consider a more challenging problem, inducing knowledge about the stereotyped ordering of events from natural texts. In this work, we are largely inspired by the scenario of CJ08. The overall strategy is the following: we process the Gigaword corpus with a high precision rule-based temporal classifier relying on explicit clues (e.g., "then", "after") to get ordered pairs of events and then we train our model on these pairs (note that clues used by the classifier are removed from the examples, so the model has to rely on verbs and their arguments). Conceptually, the difference between our approach and CJ08 is in using a different temporal classifier, not enforcing that event pairs have the same protagonist, and learning an event embedding model instead of scoring event sequences based on verb-pair frequencies.

We also evaluate our system on examples extracted using the same temporal classifier (but validated manually) which allows us to use much larger tests set, and, consequently, provide more detailed and reliable error analysis.

3.2.1 Data and task

The Gigaword corpus consists of news data from different news agencies and newspapers. For testing and development we took the AFP (Agence France-Presse) section, as it appeared most different from the rest when comparing sets of extracted event pairs (other sections correspond mostly to US agencies). The AFP section was not used for

	Accuracy (%)
BL	60.7
CJ08	60.1
EE_{verb}	75.9
EE	**83.5**

Table 3: Results on the Gigaword data for the verb-frequency baseline (BL), the verb-only embedding model (EE_{verb}), the full model (EE) and CJ08 rules.

training. This selection strategy was chosen to create a negative bias for our model which is more expressive than the baseline methods and, consequently, better at memorizing examples.

As a rule-based temporal classifier, we used high precision "happens-before" rules from the VerbOcean system (Chklovski and Pantel, 2004). Consider "*to* ⟨*verb-x*⟩ *and then* ⟨*verb-y*⟩" as one example of such rule. We used predicted collapsed Stanford dependencies (de Marneffe et al., 2006) to extract arguments of the verbs, and used only a subset of dependents of a verb.[7] This preprocessing ensured that (1) clues which form part of a pattern are not observable by our model both at train and test time; (2) there is no systematic difference between both events (e.g., for collapsed dependencies, the noun subject is attached to both verbs even if the verbs are conjoined); (3) no information about the order of events in text is available to the models. Applying these rules resulted in 22,446 event pairs for training, and we split additional 1,015 pairs from the AFP section into 812 for final testing and 203 for development. We manually validated random 50 examples and all 50 of them followed the correct temporal order, so we chose not to hand correct the test set.

We largely followed the same training and evaluation regime as for the crowdsourced data. We set the regularization parameter and the learning rate to 0.01 and $5.e-4$ respectively. The model was trained for 600 epochs. The embedding sizes were 30 and 50 dimensions for words and events, respectively.

3.2.2 Results and discussion

In our experiments, as before, we use BL as a baseline, and EE_{verb} as a verb-only simplified version of our approach. We used another baseline consisting of the verb pair ordering counts provided by Chambers and Jurafsky (2008).[8] We refer this baseline as CJ08. Note also that BL can be regarded as a reimplementation of CJ08 but with a different temporal classifier. We report results in Table 3.

The observations are largerly the same as before: (1) the full model substantially outperforms all other approaches (p-level < 0.001 with the permutation test); (2) enforcing transitivity is very helpful (75.9 % for EE_{verb} vs. 60.1% for BL). Surprisingly CJ08 rules produce as good results as BL, suggesting that maybe our learning set-ups are not that different.

However, an interesting question is in which situations using a more expressive model, EE, is beneficial. If these accuracy gains have to do with memorizing the data, it may not generalize well to other domains or datasets. In order to test this hypothesis we divided the test examples in three frequency bands according to the frequency of the corresponding verb pairs in the training set (total, in both orders). There are 513, 249 and 50 event pairs in the bands corresponding to unseen pairs of verbs, frequency ≤ 10 and frequency > 10, respectively. These counts emphasize that correct predictions on unseen pairs are crucial and these are exactly where BL would be equivalent to a random guess. Also, this suggest, even before looking into the results, that memorization is irrelevant. The results for BL, CJ08, EE_{verb} and EE are shown in Figure 3.

One observation is that most gains for EE and EE_{verb} are due to an improvement on unseen pairs. This is fairly natural, as both transitivity and information about arguments are the only sources of information available. In this context it is important to note that some of the verbs are *light*, in the sense that they have little semantic content of their own (e.g., *take*, *get*) and the event semantics can only be derived from analyzing their arguments (e.g., *take an exam* vs. *take a detour*). On the high frequency verb pairs all systems perform equally well, except for CJ08 as it was estimated from somewhat different data.

In order to understand how transitivity works, we considered a few unseen predicate pairs where the EE_{verb} model was correctly predicting their order. For many of these pairs there were no infer-

[7] The list of dependencies not considered: *aux*, *auxpass*, *attr*, *appos*, *cc*, *conj*, *complm*, *cop*, *dep*, *det*, *punct*, *mwe*.

[8] These verb pair frequency counts are available at www.usna.edu/Users/cs/nchamber/data/schemas/acl09/verb-pair-orders.gz

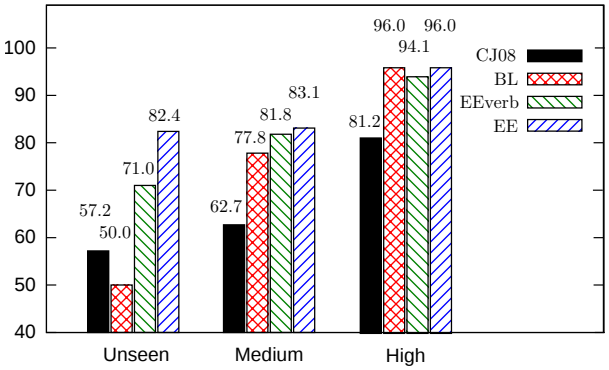

Figure 3: Results for different frequency bands: unseen, medium frequency (between 1 and 10) and high frequency ($>$ 10) verb pairs.

ence chains of length 2 (e.g., chain of length 2 was found for the pair *accept* $\prec$ *carry*: *accept* $\prec$ *get* and *get* $\prec$ *carry* but not many other pairs). This observation suggest that our model captures some non-trivial transitivity rules.

4 Related Work

Additionally to the work on script induction discussed above (Chambers and Jurafsky, 2008, 2009; Regneri et al., 2010), other methods for unsupervised learning of event semantics have been proposed. These methods include unsupervised *frame* induction techniques (O'Connor, 2012; Modi et al., 2012). Frames encode situations (or objects) along with their participants and properties (Fillmore, 1976). Events in these unsupervised approaches are represented with categorical latent variables, and they are induced relying primarily on the selectional preferences' signal. The very recent work of Cheung et al. (2013) can be regarded as their extension but Cheung et al. also model transitions between events with Markov models. However, neither of these approaches considers (or directly optimizes) the discriminative objective of learning to order events, and neither of them uses distributed representations to encode semantic properties of events.

As we pointed out before, our embedding approach is similar (or, in fact, a simplification of) the phrase embedding methods studied in the recent work on distributional compositional semantics (Baroni and Zamparelli, 2011; Socher et al., 2012). However, they have not specifically looked into representing script information. Approaches which study embeddings of relations in knowledge bases (e.g., Riedel et al. (2013)) bear some similarity to the methods proposed in this work but they are mostly limited to binary relations and deal with predicting missing relations rather than with temporal reasoning of any kind.

Identification of temporal relations within a text is a challenging problem and an active area of research (see, e.g., the TempEval task (UzZaman et al., 2013)). Many rule-based and supervised approaches have been proposed in the past. However, integration of common sense knowledge induced from large-scale unannotated resources still remains a challenge. We believe that our approach will provide a powerful signal complementary to information exploited by most existing methods.

5 Conclusions

We have developed a statistical model for representing common sense knowledge about prototypical event orderings. Our model induces distributed representations of events by composing predicate and argument representations. These representations capture properties relevant to predicting stereotyped orderings of events. We learn these representations and the ordering component from unannotated data. We evaluated our model in two different settings: from crowdsourced data and natural news texts. In both set-ups our method outperformed baselines and previously proposed systems by a large margin. This boost in performance is primarily caused by exploiting transitivity of temporal relations and capturing information encoded by predicate arguments.

The primary area of future work is to exploit our method in applications such as question answering. Another obvious applications is discovery of temporal relations within documents (UzZaman et al., 2013) where common sense knowledge implicit in script information, induced from large unannotated corpora, should be highly beneficial. Our current model uses a fairly naive semantic composition component, we plan to extend it with more powerful recursive embedding methods which should be especially beneficial when considering very large text collections.

6 Acknowledgements

Thanks to Lea Frermann, Michaela Regneri and Manfred Pinkal for suggestions and help with the data. This work is partially supported by the MMCI Cluster of Excellence at the Saarland University.

References

James F Allen. 1983. Maintaining knowledge about temporal intervals. *Communications of the ACM*, 26(11):832–843.

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

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

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

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

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

Jackie Chi Kit Cheung, Hoifung Poon, and Lucy Vanderwende. 2013. Probabilistic frame induction. In *Proceedings of NAACL*.

Timothy Chklovski and Patrick Pantel. 2004. Verbocean: Mining the web for fine-grained semantic verb relations. In *Proceedings of EMNLP*.

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

Marie-Catherine de Marneffe, Bill MacCartney, and Christopher D. Manning. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*.

Charles Fillmore. 1976. Frame semantics and the nature of language. *Annals of the New York Academy of Sciences*, 280(1):20–32.

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

Martin Charles Golumbic and Ron Shamir. 1993. Complexity and algorithms for reasoning about time: A graph-theoretic approach. *Journal of ACM*, 40(5):1108–1133.

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

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

Geoffrey E. Hinton, Nitish Srivastava, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2012. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv: CoRR, abs/1207.0580*.

Ashutosh Modi, Ivan Titov, and Alexandre Klementiev. 2012. Unsupervised induction of frame-semantic representations. In *Proceedings of the NAACL-HLT Workshop on Inducing Linguistic Structure*. Montreal, Canada.

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

Brendan O'Connor. 2012. Learning frames from text with an unsupervised latent variable model. *CMU Technical Report*.

Robert Parker, David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2011. English gigaword fifth edition. *Linguistic Data Consortium*.

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

Sebastian Riedel, Limin Yao, Andrew McCallum, and Benjamin Marlin. 2013. Relation extraction with matrix factorization and universal schemas. *TACL*.

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

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

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

Naushad UzZaman, Hector Llorens, Leon Derczynski, James Allen, Marc Verhagen, and James Pustejovsky. 2013. Semeval-2013 task 1: Tempeval-3: Evaluating time expressions, events, and temporal relations. In *Proceedings of SemEval*.