Embedding time expressions for deep temporal ordering models

Tanya Goyal and **Greg Durrett**
Department of Computer Science
The University of Texas at Austin
tanyagoyal@utexas.edu, gdurrett@cs.utexas.edu

Abstract

Data-driven models have demonstrated state-of-the-art performance in inferring the temporal ordering of events in text. However, these models often overlook explicit temporal signals, such as dates and time windows. Rule-based methods can be used to identify the temporal links between these time expressions (timexes), but they fail to capture timexes' interactions with events and are hard to integrate with the distributed representations of neural net models. In this paper, we introduce a framework to infuse temporal awareness into such models by learning a pre-trained model to embed timexes. We generate synthetic data consisting of pairs of timexes, then train a character LSTM to learn embeddings and classify the timexes' temporal relation. We evaluate the utility of these embeddings in the context of a strong neural model for event temporal ordering, and show a small increase in performance on the MATRES dataset and more substantial gains on an automatically collected dataset with more frequent event-timex interactions.[1]

1 Introduction

Understanding the temporal ordering of events in a document is an important component of document understanding and plays an integral role in tasks such as timeline creation (Do et al., 2012), temporal question answering (Llorens et al., 2015) and causality inference (Mostafazadeh et al., 2016; Ning et al., 2018a). Inferring temporal event order is challenging as it often disagrees with the narrative order in text. Past work on temporal relation extraction has exploited cues such as global constraints on the temporal graph structure (Bramsen et al., 2006; Chambers and Jurafsky, 2008; Ning et al., 2017), world knowledge (Ning et al., 2018b), grouping of events (Tourille et al., 2017), or fusing these cues more effectively with deep models (Meng et al., 2017; Cheng and Miyao, 2017). One key component of temporal understanding is time expressions (timexes) that help anchor events to the time axis, but few recent systems effectively use the knowledge derivable from time expressions in their models. They either give timexes no special treatment (Ning et al., 2017) or rely on rule-based post-processing modules to remove inconsistencies with explicit timexes (Chambers et al., 2014; Meng et al., 2017).

In this work, we address this shortcoming by introducing a framework for including rich representations of timexes in neural models. These models implicitly capture some information via word embeddings (Mikolov et al., 2013; Pennington et al., 2014) or contextualized embeddings such as ELMo (Peters et al., 2018). However, these embeddings do not encode the full richness of temporal information needed for this task. For example, these systems fail to infer the correct event relation in the following sentence: *He visited France in 1992 and went to Germany in 1963.* partially because the dates *1992* and *1963* do not have temporally-informed embeddings.

We devise a method for embedding timexes that more explicitly reflects their temporal status. Specifically, we sample pairs of time expressions from synthetic data, train character LSTM models to encode these time expressions and classify their temporal ordering. Due to the amount and type of data they are trained on, these time embeddings will naturally capture the temporal ordering of events in standard text and generalize to things like unseen timex values.

We incorporate these embeddings into neural models for temporal relation extraction. When used in an improved version of the model from Cheng and Miyao (2017), we show a small im-

[1]Data and code are available at https://github.com/tagoyal/Temporal-event-ordering

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 4400–4406
Florence, Italy, July 28 - August 2, 2019.

provement in performance on the benchmark MATRES dataset (Ning et al., 2018c). Additionally, to evaluate the full potential of the proposed approach, we construct another dataset with more frequent event-timex interactions using distant supervision. On this dataset, our proposed approach substantially outperforms the ELMo-equipped baseline model.

2 Methodology

We improve upon the model architecture proposed by Cheng and Miyao (2017) for temporal relation extraction, which involves classifying the temporal relation between a given pair of events e_1 and e_2. Our proposed architecture is outlined in Figure 1. The input to the system consists of two sentences, $s_1 = \{x^1_1, x^1_2, ...x^1_n\}$ and $s_2 = \{x^2_1, x^2_2, ...x^2_m\}$ containing e_1 and e_2 respectively. Note that s_1 and s_2 may correspond to the same sentence.

Input Encoding For each token x_k in each sentence, we obtain a distributed representation $\tilde{x}_k = [v_w; v_p; v_t]$. Here, v_w is the word embedding obtained from GloVe or contextualized word embeddings from ELMo, v_p is a randomly initialized and trainable embedding of the part-of-speech tag, and v_t corresponds to the timex embedding derived for time expressions (explained in Section 2.1).

Contextual Encoding A biLSTM is used to obtain contextualized embeddings h_k for each token x_k in the two sentences, as shown in Figure 1. The parameters are shared between these lower biLSTMs for the two sentences. Prior work (Cheng and Miyao, 2017) does not include these lower biLSTMs and only leverages the dependency encoding, explained next.

Dependency Encoding We use the Stanford Dependency Parser (Manning et al., 2014) to extract the dependency paths for both events to their lowest common ancestor. For inter-sentence event pairs, paths are extracted to the root of each sentence. Each vector along the dependency path is fed into an upper biLSTM to produce output h_{upper}. Formally, for sentence s_1,

$$h^1_{upper} = \text{biLSTM}([h_k \text{ for } k \in \text{dep-path}(e_1)])$$

Parameters are shared between the upper biLSTMs for the two sentences.

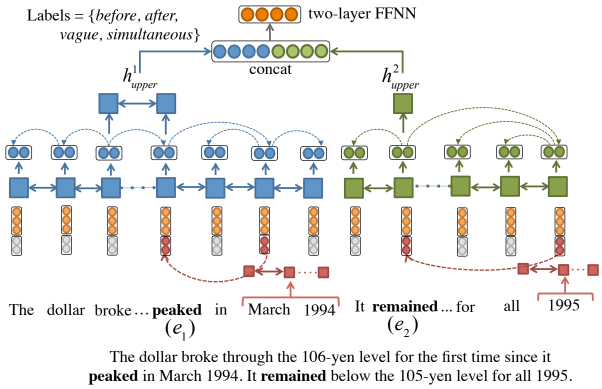

Figure 1: Temporal relation extraction model. Here, *peaked* and *remained* are the two events under consideration. The sentences are passed through the lower LSTM, then the outputs corresponding to the events' dependency paths are fed to the upper LSTMs, which produce input to feedforward and classification layers. Time expressions are embedded with a character-level model and broadcasted to events that they modify.

Output We concatenate the outputs of the upper biLSTMs' embeddings for the two events to obtain $z = [h^1_{upper}; h^2_{upper}]$. We apply multiple feedforward layers with ReLU non-linearity, followed by a softmax layer to obtain output probabilities for the four labels *before, after, vague* and *simultaneous*,[2] denoting the temporal relation between the event pair (e_1, e_2). The network is trained using the cross entropy loss.

2.1 Time Embeddings

Next, we outline our approach for constructing the timex embeddings v_t, which are concatenated to word and POS embeddings to generate the input encoding (as discussed in the previous section).

Training Data To obtain time embeddings, we first constructed a grammar of time expressions in the dataset. We identified two main classes of timexes: explicit datetimes expressed in recognizable timex format (e.g. *Sept. 12*, *1993*, *August 2013*, *1998*, *10-12-2014*, *9th January*, etc.) and natural language time indicators (e.g. *two months ago*, *5 weeks ago*, *next year*, etc.). We designed generic templates that covered both these categories of timexes, e.g. [*mm dd, yy*].[3] By randomly sampling values for the slots, we can generate valid time expressions based on this tem-

[2]These are the labels used in the MATRES dataset (Ning et al., 2018b), but our classifier could in principle generalize to other label schemes as well.

[3]See the appendix for more examples.

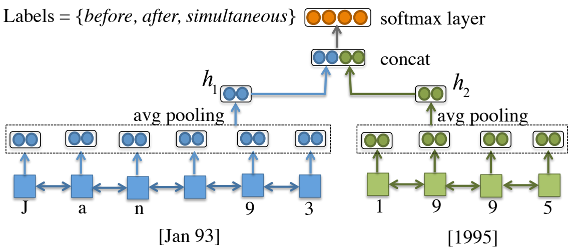

Figure 2: Timex model. The output of character biLSTMs is used to as input to classification. These vectors serve as time embeddings in the downstream tasks.

Model	w/ linear	w/ biLSTM
GloVe embedding	81.3	88.7
ELMo embedding	88.3	97.6
Char embedding (Ours)	–	97.3

Table 1: Performance on the synthetic timex dataset, classifying a pair of timexes as *before*, *after*, or *simultaneous*. Including a biLSTM layer (as depicted in Figure 2) leads to higher performance than just pooling and a linear layer. Character-level modeling (from ELMo or our learned embeddings) is important for high performance.

plate. We used pairs of such randomly generated timexes to construct training data for our timex model. Since we generate time expression pairs from a pre-defined grammar and set of templates, it is straightforward to obtain the temporal order between the pairs of timexes.

Model Architecture The model architecture for the timex model is outlined in Figure 2. The input to the system are two time expressions, t_1 and t_2. We use character biLSTMs to obtain distributed representations of both time expressions. We obtain time embeddings h_1 and h_2 for timex t_1 and t_2 by averaging the outputs of biLSTM layer. The two time embeddings are concatenated and fed through multiple feed forward layers with non-linearity. This is followed by a softmax layer that produces the output probabilities for the three label classes (*before, after* and *simultaneous*), denoting the temporal relation between the two time expressions. We train this network with the cross entropy loss.

Inclusion in Temporal Models For a given time expression, the average of the outputs of the biLSTM model (h_1) is used as the time embedding as shown in Figure 1. For other non-timex tokens, a zero vector is concatenated instead. Further, we also project the time embedding for a timex to the corresponding event it modifies according to a set of grammatical rules on the dependency parse, shown with red arrows in Figure 1.

3 Experiments

3.1 Timex Pair Ordering

First, we intrinsically evaluate the performance of the character-level timex model, outlined in Section 2.1. We generated 50000 random pairs of time expressions for training and 5000 randomly generated pairs for test. We seek to answer two questions: first, can our proposed timex model successfully capture temporal information necessary to order these timex pairs, and second, how effective are pre-trained embeddings for this task?

Table 1 shows a comparison between several models in our synthetic timex setting. Our proposed timex model achieves an accuracy of 97.3%. This high accuracy indicates that the model has effectively learned from the training data; its timex embeddings contain temporal ordering information which can be used for downstream tasks.

We also evaluate whether pre-trained embeddings such as ELMo or GloVe contain the necessary temporal information necessary for classifying the temporal order between timex pairs. We first test these with a minimal model. We construct a distributed representation of each time expression (obtained by average pooling the token level GloVe or ELMo embeddings), perform element-wise subtraction between the two embeddings, and feed the result through a linear classification layer that produces the output probabilities for the temporal label classes. The left column of Table 1 shows that while both GloVe and ELMo contain some temporal information, our proposed model's additional parameters and richer embedding scheme lead to higher performance.

We further experiments to investigate if ELMo or GloVe can additionally be used in our timex model to obtain even more powerful embeddings. We replace our model's character-level vectors and character-level biLSTM with token-level pre-trained vectors (either contextualized vectors from ELMo or non-contextual vectors from GloVe) and a token-level biLSTM. As before, the outputs of this biLSTM for the two timexes are concatenated and further fed to feedforward and softmax layers for temporal label prediction. Using ELMo embeddings in this manner does not lead to a substantial improvement over previous results, with an accuracy of 97.6% for the temporal relation classification objective on the same test set. However, the performance using GloVe embeddings drops to 88.7%. This drop in performance can partially be attributed to the word-level nature of GloVe vectors, which do not necessarily cover every year that might be seen in the dataset. We used the GloVe vectors with 840 billion tokens (largest available) to circumvent this issue and minimize the number of out of vocabulary instances, but still see low performance.

3.2 Event Temporal Ordering

Next, we investigate the effectiveness of our timex embeddings in the context of our full event temporal ordering model. We evaluate on two event temporal ordering datasets, one real and one artificially constructed.

3.2.1 Evaluation on MATRES

We evaluate on the MATRES dataset proposed in Ning et al. (2018c). This dataset is designed to be less ambiguous than TimeBank-Dense (Cassidy et al., 2014). MATRES contains temporal annotations for documents from the TimeBank (Pustejovsky et al., 2003), AQUAINT (Graff, 2002) and Platinum datasets (UzZaman et al., 2013). We follow standard practice and use TimeBank and AQUAINT (256 articles) for training and Platinum (20 articles) for testing.

Table 2 outlines the performance of the proposed approach on MATRES. We implemented the model proposed by Cheng and Miyao (2017) and compare against it. We evaluate the models using both GloVe and ELMo embeddings. Our results show substantial improvement over this baseline model. Moreover, including time embeddings as additional input to the improved models leads to a small improvement in the overall accuracy. However, we did not find the results to be statistically significant according to a bootstrap resampling test (GloVe p-value $= 0.349$, ELMo p-value $= 0.267$).[4]

Note that only a fraction of examples in the MATRES dataset contain distinct time expressions that can be compared to resolve temporal ordering. To further evaluate our approach, we investigated whether an equivalent performance improvement could be achieved through post-processing rules involving time expressions. We identified event pairs in the data for which both events had an accompanying time expression modifying the event according to the dependency parse. We can then infer the temporal relation between the event pair using rules on top of these timexes. However, we observed that such a post-processing scheme had very low coverage in the dataset and could not repair *any* errors in the development set. We therefore turn our attention to a setting with a richer set of timexes for further evaluation.[5]

Model	GloVe	ELMo
Cheng and Miyao (2017)	59.53	65.50
Ours w/o timex embed	62.83	68.45
Ours w/ timex embed	63.22	68.61

Table 2: Performance of our event temporal ordering model on the MATRES dataset. We report the mean accuracy over 3 runs of each model. Our model improves substantially over Cheng and Miyao (2017). Including timexes leads to small accuracy gains, partially due to the fact that timexes often do not occur with the dataset's hard examples.

3.2.2 Evaluation on Distant Data

In MATRES, only a fraction of the examples contain time expressions and are consequently affected by inclusion of time embeddings. Therefore, to test the full potential of the proposed approach, we additionally collect a test dataset of examples with explicit timexes that expose their temporal relation; we view the timexes as distant supervision for the event pairs. To identify such examples, we use two high precision classifiers proposed in Chambers et al. (2014): (a) an event-timex classifier that identifies the temporal relation between adjacent verb and time expressions (precision = 0.92), (b) a timex-timex classifier that identifies the temporal relation between two time expressions (precision = 0.88). These classifiers can allow us to directly infer the time relation be-

[4]Augmenting word embeddings with time embeddings increases the number of network parameters; however, additional experiments revealed that increasing the size of the GloVe embeddings in the basic temporal model did not lead to an improvement in performance. Therefore, it does not seem that extra parameters in the model contribute to the observed improvements.

[5]In prior work (Cheng and Miyao, 2017; Meng and Rumshisky, 2018), machine learning classifiers are used to infer a wider range of event-timex links, which can potentially increase the informativeness of timexes. However, many of the links they target require complex inferences to determine, and as a result those works report relatively low performance for such classifiers. Hence, we do not compare to these methods in our experiments.

	2000	3000	4000
GloVe			
Ours w/o Timex Embed	74.0	76.8	78.2
Ours w/ Masked Timex	73.9	75.5	77.1
Ours w/ Timex Embed	81.6	83.2	83.1
ELMo			
Ours w/o Timex Embed	80.1	83.8	84.3
Ours w/ Masked Timex	79.8	80.1	80.7
Ours w/ Timex Embed	82.3	84.5	84.8

Table 3: Performance of our models on the distantly-labeled event ordering data. We report overall accuracy values. In both the GloVe and ELMo settings, our timex embeddings lead to higher performance. The ELMo model gets substantially worse when timexes are masked, indicating that it is organically exploiting these better than GloVe is.

tween an event pair where each event is linked to a timex. An example event pair from the distant data thus collected is: *"Riyadh* ***suspended*** *aid to the Palestinians in 1990 when it accused Arafat of siding with Iraq after the 1990 invasion of Kuwait, but it* ***restored*** *aid in 1994."*[6] Note that the classifiers used have very low recall in general, but by running the system on Gigaword (Graff et al., 2007), we can extract a large dataset in spite of this.

Since this distant data is created using rule-based classifiers, given a large amount of training data, the baseline model can achieve high performance as it learns to infer these rules. However, our aim is to improve the performance of the event ordering model on moderately sized datasets, where the knowledge induction from timex embeddings play a larger role. Therefore, we report results on training sets of size 2000, 3000, and 4000 samples. The test set is kept constant with 1000 samples.

Table 3 outlines the performance of the temporal models on this dataset. We evaluate our models across three settings: (a) our event ordering model without including timex embeddings, (b) our event ordering model with masking of time tokens (replacing it with UNK tokens) and (c) our full model including timex embeddings. We evaluate the models using both GloVe and ELMo embeddings as input. In both settings, incorporating our timexes leads to higher performance. For GloVe, the performance of the basic temporal model is similar to that when the time expression is masked out. This demonstrates that the temporal model does not use the knowledge from time expressions when making temporal relation predictions. However, in the ELMo setting, we observed a larger drop in performance by masking out the time expressions compared to GloVe embeddings. This demonstrates that the ELMo embeddings are not agnostic to time-expressions in the sentence, although they still show improvement by inclusion of timex embeddings trained specifically with the temporal classification objective on small datasets.

[6]See the appendix for more samples from the distant data.

4 Conclusion

In this paper, we propose a framework to learn temporally-aware timex embeddings from synthetic data. Through experiments on two datasets, we show that incorporating these embeddings in deep temporal models leads to an improvement in the overall temporal classification performance.

5 Acknowledgments

This work was partially supported by NSF Grant IIS-1814522, NSF Grant SHF-1762299, a Bloomberg Data Science Grant, and an equipment grant from NVIDIA. The authors acknowledge the Texas Advanced Computing Center (TACC) at The University of Texas at Austin for providing HPC resources used to conduct this research. Also, thanks to the anonymous reviewers for their helpful comments.

References

Philip Bramsen, Pawan Deshpande, Yoong Keok Lee, and Regina Barzilay. 2006. Inducing Temporal Graphs. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 189–198, Stroudsburg, PA, USA. Association for Computational Linguistics.

Taylor Cassidy, Bill McDowell, Nathanael Chambers, and Steven Bethard. 2014. An Annotation Framework for Dense Event Ordering. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 501–506.

Nathanael Chambers, Taylor Cassidy, Bill McDowell, and Steven Bethard. 2014. Dense event ordering with a multi-pass architecture. *Transactions of the Association for Computational Linguistics*, 2:273–284.

Nathanael Chambers and Dan Jurafsky. 2008. Jointly Combining Implicit Constraints Improves Temporal Ordering. In *Proceedings of the Conference on*

Empirical Methods in Natural Language Processing, EMNLP '08, pages 698–706, Stroudsburg, PA, USA. Association for Computational Linguistics.

Fei Cheng and Yusuke Miyao. 2017. Classifying Temporal Relations by Bidirectional LSTM over Dependency Paths. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 1–6.

Quang Xuan Do, Wei Lu, and Dan Roth. 2012. Joint inference for event timeline construction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, pages 677–687. Association for Computational Linguistics.

David Graff. 2002. The AQUAINT corpus of English news text. *Linguistic Data Consortium, Philadelphia*.

David Graff, Junbo Kong, Ke Chen, and Kazuaki Maeda. 2007. English Gigaword Third Edition. Linguistic Data Consortium, Catalog Number LDC2007T07.

Hector Llorens, Nathanael Chambers, Naushad UzZaman, Nasrin Mostafazadeh, James Allen, and James Pustejovsky. 2015. SemEval-2015 Task 5: QA TEMPEVAL-Evaluating Temporal Information Understanding with Question Answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*, pages 792–800.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60.

Yuanliang Meng and Anna Rumshisky. 2018. Context-Aware Neural Model for Temporal Information Extraction. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 527–536.

Yuanliang Meng, Anna Rumshisky, and Alexey Romanov. 2017. Temporal Information Extraction for Question Answering Using Syntactic Dependencies in an LSTM-based Architecture. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 887–896.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Nasrin Mostafazadeh, Alyson Grealish, Nathanael Chambers, James Allen, and Lucy Vanderwende. 2016. CaTeRS: Causal and temporal relation scheme for semantic annotation of event structures. In *Proceedings of the Fourth Workshop on Events*, pages 51–61.

Qiang Ning, Zhili Feng, and Dan Roth. 2017. A structured learning approach to temporal relation extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1027–1037.

Qiang Ning, Zhili Feng, Hao Wu, and Dan Roth. 2018a. Joint reasoning for temporal and causal relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 2278–2288.

Qiang Ning, Hao Wu, Haoruo Peng, and Dan Roth. 2018b. Improving Temporal Relation Extraction with a Globally Acquired Statistical Resource. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 841–851.

Qiang Ning, Hao Wu, and Dan Roth. 2018c. A Multi-Axis Annotation Scheme for Event Temporal Relations. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1318–1328.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep Contextualized Word Representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237.

James Pustejovsky, Patrick Hanks, Roser Sauri, Andrew See, Robert Gaizauskas, Andrea Setzer, Dragomir Radev, Beth Sundheim, David Day, Lisa Ferro, et al. 2003. The timebank corpus. In *Corpus linguistics*, volume 2003, page 40. Lancaster, UK.

Julien Tourille, Olivier Ferret, Aurelie Neveol, and Xavier Tannier. 2017. Neural Architecture for Temporal Relation Extraction: A Bi-LSTM Approach for Detecting Narrative Containers. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 224–230.

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 *Second Joint Conference on Lexical and Computational Semantics (* SEM), Volume 2: Proceedings of the Seventh International Workshop on Semantic Evaluation (SemEval 2013)*, volume 2, pages 1–9.

Event Pair	Label
Former Singapore premier Lee Kuan Yew, who **came** to power in 1959, **stepped** down in 1990 in favour of the incumbent, prime minister Goh Chok Tong, but remains influential as a senior minister in Goh's cabinet.	Before
Relations between Sudan and Saudi Arabia **grew** tense in 1990 when Riyadh accused Khartoum of supporting Iraq after its invasion of Kuwait and **worsened** in 1992 when Sudan granted asylum to Saudi militant Osama Bin Laden.	Before
The Israeli-Syrian peace talks **launched** in 1991 are mainly focusing on Damascus' insistence that Israel withdraw its troops from the Golan Heights in exchange for peace. That territory has been **occupied** by Israeli troops since 1967.	After
Resolutions were **passed** by the UN Security Council after the first Indo-Pakistan war over Kashmir in 1948. The dispute **led** to a second war between the neighbours in 1965.	Before
Turkish mainland forces **invaded** Northern Cyprus in 1974 after a coup in Nicosia backed by the military junta then ruling Greece. A Turkish-Cypriot state was **declared** in 1983, and Ankara now has about 35,000 troops and 400 tanks stationed there.	Before
More people **watched** Formula One on television in 1995 than **watched** the world cup in 1994.	After
He was **freed** six months early in September 1993 but **re-arrested** in April 1994 after meeting with John Shattuck, the US assistant secretary of state for human rights.	Before

Table 4: Examples from the distantly-labeled event ordering data. Events are shown in bold and may be co-located in a single sentence or span two sentences. Event-timex relations are recognized with high-precision classifiers from Chambers et al. (2014).

A Appendix

A.1 Timex Templates

We use generic templates for time expressions to generate training data for the timex model. Two kinds of templates were generated: (1) explicit datetimes, and (2) natural language time indicators. Examples of each of these kinds are outlined below:

1. Explicit datetime templates: *[yyyy], ['yy], [mm dd yy], [mm yy], [mmm yyyy], [mmm dd yyyy]*, etc.

2. Natural language indicators: *[xx units later], [xx units before], [now], [past xx units]*, etc., where *xx* is filled by a numerical value and *units* refers to a time unit such as months, days, or years.

Timex pairs generated through these templates can be converted to a standardized time scale and hence easily compared. It is therefore straight forward to infer the gold label for each pair of generated timexes. For MATRES, 75% of the pairs in the training set for the timex model are sampled from explicit datetime templates, and the rest are sampled from natural language templates. This relative ratio was heuristically determined. 100% of the pairs were drawn from explicit datetime templates for the distant data.

A.2 Examples from Distant Data

Table 4 provides some examples of event pairs, and their corresponding label from the distant data. This dataset is automatically created using two high precision rule-based classifiers.