Event Ordering with a Generalized Model for Sieve Prediction Ranking

Bill McDowell
The Pennsylvania State University
forkunited@gmail.com

Nathanael Chambers
United States Naval Academy
nchamber@usna.edu

Alexander G. Ororbia II
The Pennsylvania State University
ago109@psu.edu

David Reitter
The Pennsylvania State University
reitter@psu.edu

Abstract

This paper improves on several aspects of a sieve-based event ordering architecture, CAEVO (Chambers et al., 2014), which creates globally consistent temporal relations between events and time expressions. First, we examine the usage of word embeddings and semantic role features. With the incorporation of these new features, we demonstrate a 5% relative F1 gain over our replicated version of CAEVO. Second, we reformulate the architecture's sieve-based inference algorithm as a *prediction reranking* method that approximately optimizes a scoring function computed using classifier precisions. Within this prediction reranking framework, we propose an alternative scoring function, showing an 8.8% relative gain over the original CAEVO. We further include an in-depth analysis of one of the main datasets that is used to evaluate temporal classifiers, and we show that in spite of the density of this corpus, there is still a danger of overfitting. While this paper focuses on temporal ordering, its results are applicable to other areas that use sieve-based architectures.

1 Introduction

Narratives that describe a series of events rarely do so in order. Basic rules of journalism dictate that important information leads a news report, and accordingly, algorithms that re-order events chronologically need to combine a wealth of contextual, rhetorical, and commonsense information.

Most research on event ordering aims to produce only partial orderings of event mentions and time expressions (Bethard and Martin, 2007; Cheng et al., 2007; UzZaman and Allen, 2010; Llorens et al., 2010; Bethard, 2013). In the past, labeled corpora used for training and evaluation contained only small subsets of pairs of events and times. Some of these corpora, like *Timebank*, have annotations restricted to salient, easily labeled pairs within the same document. Other more recent data sets contain annotations that form timelines of events that involve common entities (Pustejovsky et al., 2003; Minard et al., 2015). Due to the lack of consistency of annotations across event pairs, it is difficult to use these corpora in accurately measuring the practical performance of event ordering algorithms.

Richer datasets are becoming available that provide more complete event orderings which include logically implied relations that are less evident from local text features. In particular, the *TimeBank-Dense* corpus provides a significantly more dense and complete set of annotations, allowing for the evaluation of methods that make use of broad contextual information across many event pairs (Cassidy et al., 2014). One method that has been developed to leverage such information is CAEVO—a sieve-based architecture that made the first effort toward dense event ordering (Chambers et al., 2014). This method maintains transitivity constraints across independent predictions from several specialized classifiers. More specifically, the architecture runs a series of "sieve" classifiers with their predictions ranked in order by precision using a held-out dataset. The higher precision classifiers are ranked more highly in the series, and predictions are expanded by transitivity rules (e.g. if event e_1 is before e_2, and e_2 is before e_3, then e_1 is before e_3) after each individual classifier generates its predictions. The high density of the constructed prediction graph allows the transitivity rules to generate accurate predictions for links that would otherwise be difficult to predict from the text.

This paper proposes improvements to CAEVO with respect to (1) feature engineering within machine learned sieves, (2) generalization of the sieve-based architecture to facilitate higher performing sieve prediction rerankings, and (3) the leveraging of unlabeled data. First, for our feature engineering improvements, we are motivated by the fact that TimeBank-Dense contains a relatively small training sample, and so we extend the feature sets for the architecture's machine learned classifiers to include features that encode lexical information about events in relatively low dimensional spaces based on word embeddings (Mikolov et al., 2013) and semantic role labeling (SRL) annotations (Gildea and Jurafsky, 2002). Second, our generalization of the sieve-based architecture allows us to experiment with alternative methods for establishing the precedence ranking of sieve predictions. Furthermore, we identify an approximate upper bound on any ranking method's performance. Lastly, in our experiments with unlabeled data, we analyze the effect of changing the density of the architecture's prediction graph. Our hypothesis is that increasing the number of predictions on unlabeled data will increase performance on labeled data through the application of CAEVO's transitivity constraints.

Our extensions produce new state-of-the-art results on the original test split of TimeBank-Dense (8.8% F1 increase). Beyond this, we describe alternative evaluations on other splits of the data in order to analyze the effect of the common small sizes of temporal corpora like TimeBank-Dense. This analysis is critical for future work in temporal ordering, and sheds further light on previous work's results.

2 Related Work

Early work on event ordering focused on developing machine-learned classifiers that label the temporal relations between small subsets of pairs of events within documents using lexical and syntactical features (Bethard and Martin, 2007; Cheng et al., 2007; UzZaman and Allen, 2010; Llorens et al., 2010; Bethard, 2013). Later work leveraged information across pairwise predictions by imposing transitivity constraints using techniques like integer linear programming and Markov logic networks (Bramsen et al., 2006; Chambers and Jurafsky, 2008; Tatu and Srikanth, 2008; Yoshikawa et al., 2009). CAEVO followed these and other hybrid rule-based approaches (D'Souza and Ng, 2013), but with the transitivity constraints yielding larger gains in performance for the more complete temporal graph constructed on the TimeBank-Dense corpus (Cassidy et al., 2014; Chambers et al., 2014).

The *TimeBank-Dense* corpus provides a significantly more dense and complete set of annotations compared to previous corpora.[1] TimeBank-Dense extends a subset of the original TimeBank corpus with annotations for (almost) *all* event-time, time-time, and event-time pairs across consecutive sentences, as well as relations to the document creation time. This dense corpus facilitated the evaluation of CAEVO—a sieve-based architecture which maintains transitivity constraints across independent predictions from several specialized classifiers.

Recent work has focused on the construction of timelines of related events, using SRL annotations to determine which events are related through common actors (Laparra et al., 2015). In addition, other work has outperformed the original CAEVO with a 2.2% relative F1 gain on TimeBank-Dense using word embedding features within a stacked ensemble of event-event, event-time, and event-creation-time logistic regression classifiers (Mirza and Tonelli, 2016). We draw inspiration from this recent work by incorporating SRL and word embedding features into the machine-learned CAEVO sieves.

The CAEVO architecture is itself inspired by the sieve-based architectures that have been successfully applied to event and entity coreference as well as spatial relation extraction tasks (Lee et al., 2012, 2013; D'Souza and Ng, 2015). Years since CAEVO's introduction, a coreference sieve architecture still achieves top performance (Lee et al., 2017). The key idea behind these architectures is to combine information from several classifiers by assigning precedence to predictions according to the reliability of the classifier from which they originate. A precision-ranked series of

[1]The new corpus is the result of several TempEval competitions (Verhagen et al., 2007, 2010; UzZaman et al., 2012) which prompted efforts to develop more complete event ordering annotations (Bramsen et al., 2006; Kolomiyets et al., 2012; Do et al., 2012).

"sieve" classifiers generate predictions, and predictions from the more reliable sieves earlier in the series inform the predictions of the less reliable sieves later in the series. Generally, the predictions from a highly-ranked sieve can inform a low-ranked sieve in several ways, but within CAEVO, predictions from early classifiers are coupled via transitive inference rules to generate an expanding set of predictions that override output from less reliable classifiers later on in the series. In the next section, we describe a more generic view of this architecture which will motivate alternative methods for assigning precedence to predictions from the collection of classifiers.

3 Generalizing Sieve Architectures

Sieve architectures are used in many areas such as entity coreference, relation extraction, and temporal ordering. A core contribution of this paper is a generalization of how these models score and rank their decisions. Like other sieve models, CAEVO uses a precision-ranked series of classifiers (i.e. "sieves") coupled with transitivity constraints to provide a solution to the event ordering task. However, prior work has not investigated alternatives to the coarse-grained precision-based rankings provided by the architecture. This section gives a generalized formal view on this precision-ranked setup, and Section 4.3.1 describes our experiments with new alternatives to traditional sieve architectures that are available within our generalized view.

Informally speaking, a sieve architecture applies a sequence of classifiers that each make their own independent labeling decisions, and the architecture resolves conflicts between these decisions by assigning precedence to those which have higher estimated precision. The architecture estimates the precision for each sieve on a development set of data, and associates all predictions from a given sieve with this precision estimate. These precision scores determine an overall ranking to predictions within the final system. When labeling a new test document, the architecture chooses predictions from all higher precision sieves over predictions from lower precision sieves. But this common ranking of predictions is coarse-grained, so this paper proposes other ways of ordering the classifier predictions. Figure 1 illustrates the difference between prediction rankings from traditional sieve architectures and our

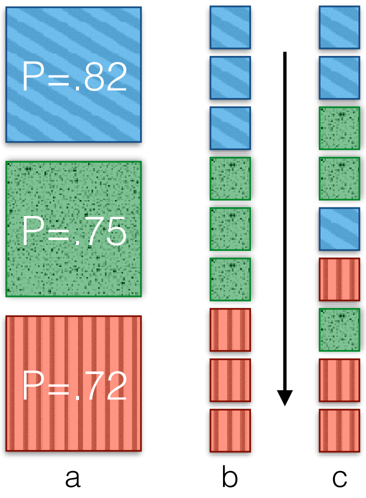

Figure 1: Sieve classifier decisions as ranked in a sieve architecture: (a) three sieves with their precisions, (b) each sieve's decisions ranked as in a traditional system, (c) a potential ranking influenced by precision, but not strictly bound to it.

alternatives. The middle column shows the strict prediction ordering given by traditional sieve systems, but the fuzzy ordering in the right column is possible within our proposed alternative architectures. Section 4.3.1 explores this in depth.

We now formally define a typical sieve architecture (in terms of the temporal ordering domain). Consider the set of event mentions E, time expressions T, and temporal relation types $L = \{\text{BEFORE}, \text{AFTER}, \text{INCLUDES}, \text{INCLUDED}, \text{SIMULTANEOUS}, \text{VAGUE}\}$. We desire an architecture that encodes functions $f_{ee} : E \times E \rightarrow L$, $f_{et} : E \times T \rightarrow L$, and $f_{tt} : T \times T \rightarrow L$ which accurately classify relations between event-event, event-time, and time-time pairs, respectively. The gold-standard annotations within our corpora are logically consistent, so we can assume that the true event orderings induced by the functions f_{ee}, f_{et}, and f_{tt} conform to the transitivity constraints given in Table 1.

Algorithm 1 depicts a generalized view of the CAEVO architecture which encodes approximations to the desired f_{ee}, f_{et}, and f_{tt} labeling functions (and this view also applies to other typical sieve systems). The algorithm combines predictions from a set of sieve classifiers $\hat{F}$ that provide partial approximations to f_{ee}, f_{et}, and f_{tt} within restricted syntactic contexts. As described by Chambers et al. (2014), $\hat{F}$ contains both rule based and machine-learned classifiers. In this pa-

Constraints
BEFORE(o_1, o_2), BEFORE(o_2, o_3) $\longrightarrow$ BEFORE(o_1, o_3)
BEFORE(o_1, o_2), INCLUDES(o_2, o_3) $\longrightarrow$ BEFORE(o_1, o_3)
BEFORE(o_1, o_2), SIMULTAN(o_2, o_3) $\longrightarrow$ BEFORE(o_1, o_3)
INCLUDED(o_1, o_2), BEFORE(o_2, o_3) $\longrightarrow$ BEFORE(o_1, o_3)
INCLUDED(o_1, o_2), INCLUDED(o_2, o_3) $\longrightarrow$ INCLUDED(o_1, o_3)
INCLUDED(o_1, o_2), SIMULTAN(o_2, o_3) $\longrightarrow$ INCLUDED(o_1, o_3)
INCLUDED(o_1, o_2), AFTER(o_2, o_3) $\longrightarrow$ AFTER(o_1, o_3)
INCLUDES(o_1, o_2), INCLUDES(o_2, o_3) $\longrightarrow$ INCLUDES(o_1, o_3)
INCLUDES(o_1, o_2), SIMULTAN(o_2, o_3) $\longrightarrow$ INCLUDES(o_1, o_3)
SIMULTAN(o_1, o_2), BEFORE(o_2, o_3) $\longrightarrow$ BEFORE(o_1, o_3)
SIMULTAN(o_1, o_2), INCLUDED(o_2, o_3) $\longrightarrow$ INCLUDED(o_1, o_3)
SIMULTAN(o_1, o_2), INCLUDES(o_2, o_3) $\longrightarrow$ INCLUDES(o_1, o_3)
SIMULTAN(o_1, o_2), SIMULTAN(o_2, o_3) $\longrightarrow$ SIMULTAN(o_1, o_3)
SIMULTAN(o_1, o_2), AFTER(o_2, o_3) $\longrightarrow$ AFTER(o_1, o_3)
AFTER(o_1, o_2), INCLUDES(o_2, o_3) $\longrightarrow$ AFTER(o_1, o_3)
AFTER(o_1, o_2), SIMULTAN(o_2, o_3) $\longrightarrow$ AFTER(o_1, o_3)
AFTER(o_1, o_2), AFTER(o_2, o_3) $\longrightarrow$ AFTER(o_1, o_3)
BEFORE(o_1, o_2) $\longrightarrow$ AFTER(o_2, o_1)
AFTER(o_1, o_2) $\longrightarrow$ BEFORE(o_2, o_1)
INCLUDES(o_1, o_2) $\longrightarrow$ INCLUDED(o_2, o_1)
INCLUDED(o_1, o_2) $\longrightarrow$ INCLUDES(o_2, o_1)
SIMULTAN(o_1, o_2) $\longrightarrow$ SIMULTAN(o_2, o_1)
VAGUE(o_1, o_2) $\longrightarrow$ VAGUE(o_2, o_1)

Table 1: Transitivity and symmetry constraints in C from Equation 1 and Algorithm 1. In this list, every constraint applies to events and/or times o_1, o_2 and o_3. We abbreviate "SIMULTANEOUS" with "SIMULTAN" due to space constraints.

per, our experiments focus on the machine learned sieves that give within-sentence event-event predictions (EEWS), within-sentence event-time predictions (ETWS), within-syntactic dominance relation event-event predictions (EED), and event to document creation time relations (EDCT).

Given a set of unlabeled data points $D \subseteq (E \cup T) \times (E \cup T)$, Algorithm 1 uses the sieves in $\hat{F}$ to construct a set of predictions $\hat{F}_D = \{(d, \hat{f}(d), \hat{f}) \mid d \in D, \hat{f} \in \hat{F}\}$ where each prediction is indexed with its associated sieve $\hat{f}$. The algorithm then sorts and partitions $\hat{F}_D$ according to a prediction scoring function $s : (D \times L \times \hat{F}) \rightarrow \mathbb{R}$. Finally, the returned set of predictions R is constructed by iteratively adding predictions from $\hat{F}_D$ in descending order (with respect to s) while applying constraints C. C consists of the transitive rules (depicted in Table 1) along with the constraint that prior predictions in R cannot be overwritten by later predictions. The rules C are applied at each iteration by extending the current predictions with those implied by transitivity.

Algorithm 1 Sieve Inference

```
1:  function SIEVEINFERENCE
2:      Input $\hat{F}$ := learned and rule-based sieves
3:      Input $D$ := data to classify
4:      Input $s$ := prediction scoring function
5:      Input $C$ := constraint application function
6:      $\hat{F_D} \leftarrow \{(d, \hat{f}(d), \hat{f}) \mid d \in D, \hat{f} \in \hat{F}\}$
7:      $P \leftarrow \hat{F}_D$ sorted and partitioned by $s$
8:      $R \leftarrow \{\}$
9:      for $i := 1$ to $|P|$ do
10:         $R \leftarrow C(P_i \cup R)$
        return $R$
```

One of the weaknesses of CAEVO (and other sieve-based systems) addressed in this paper is that its scoring function, $s(d, \hat{f}(d), \hat{f})$, is simply the precision of $\hat{f}$ as measured on held-out data. All predictions made by $\hat{f}$ must have the same ranking score (see Figure 1 again). This coarse-ranking is likely to be sub-optimal relative to rankings based on other scoring functions.

We can motivate improvements to Algorithm 1 by viewing it as a greedy approximation to the optimization problem which chooses a set of scored predictions according to:

$$R = \underset{S \subseteq \hat{F}_D}{\arg\max} \left(\sum_{p \in S} s(p) \right) \text{ subject to } C \tag{1}$$

Given the view that CAEVO is providing a solution to the objective in Equation 1 using Algorithm 1, it is straightforward to see possible directions for improvement. Namely, the architecture can be improved through changes to the sieves $\hat{F}$, the scoring methods s, the constraints C, the data D, and the underlying greedy approximation algorithm. Intuitively, if we want Equation 1 to give a highly accurate set of predictions, then we should pick an $\hat{F}$ to contain more accurate classifiers [2], an s which ranks correct predictions above incorrect predictions, and a large set D which enables C to propagate precise labels from easy-to-classify data samples onto hard-to-classify data samples. Notably, CAEVO's rigid choice of scoring function s to be the precision of $\hat{f}$ only allows s to give a coarse-grained scoring, which can score incorrect predictions higher than correct predictions.

[2]This includes possibility that one or more $\hat{f} \in \hat{F}$ could be parameterized by more complex function approximators, such as neural networks (LeCun et al., 2015).

Furthermore, CAEVO's sieve-inference in Algorithm 1 is greedy, and other methods like integer linear programming (ILP) which provide better solutions to Equation 1 might yield more accurate predictions. Lastly, CAEVO limits D to only contain *labeled* evaluation data without taking advantage of constraints imposed across *unlabeled* event pairs. These observations motivate several of the extensions we describe and experiment with below.

4 Models and Experiments

In our experiments, we replicate CAEVO, add new features to the sieves, modify the scoring function, and include larger amounts of related unlabeled data to further constrain the predictions. Unless otherwise noted, results are computed using the original train-dev-test split of the TimeBank-Dense and original CAEVO experiments (Chambers et al., 2014; Cassidy et al., 2014).

4.1 Replication

We replicate the CAEVO architecture within a more generic framework with the aim of substantiating and extending the CAEVO results from Chambers et al. (2014). The replication process allows us to validate the robustness of the originally published results while determining their sensitivity to various parameter settings.

We reconstruct features within an alternative feature engineering pipeline, and ensure that the feature matrices match those from the original system. During this process, we observed two issues in the original system. First, features based on gold-standard event "tense", "aspect", and "class" were not included in the machine-learned models that produced the reported results even though they were described in the original paper (they appear to have been inadvertently configured off). In light of this, we leave these features out of our replicated architecture, but add them into the revised architecture in our feature engineering experiments. Second, the EEWS, EED, and ETWS sieves in CAEVO used a minimum feature occurrence cutoff of 2 across training data whereas EDCT used a cutoff of 1. We experiment with different settings of these values in the next section. Other minor bugs and the details of replication are described in the appendix.

The **R** column of Table 2 gives micro-averaged accuracies[3] for the four machine-learned sieves and the full replicated architecture. These accuracies reproduce the original CAEVO results up to less than 1% discrepancy in accuracy due to version differences between the machine-learning libraries and minor bugs in the original system. [4]

4.2 Rich Feature Engineering

We extend our CAEVO replication with additional knowledge of event attributes, word embeddings, and SRL labels for each of the machine learned sieves.

4.2.1 Event Attributes

As noted above, the original CAEVO paper had reported the use of gold-standard TimeML tense, aspect, class, polarity, and modality event attribute features, but close inspection of the architecture suggests that these features had been left out when computing the final results. We experimented with adding features computed from these attributes into each of the machine-learned classifiers. For each event in a given event-event or event-time pair, we extend the feature vector with indicators for possible values of each event attribute. Also, for each event-event pair, we extend the feature vector with indicators of whether the event attributes are equal for the source and target (e.g. equal tense), as well as features representing the conjunction of each attribute across source and target (e.g. for the tense attribute, one of the indicators is *PAST-FUTURE*, which is for a pair containing a past tensed event and a future tensed event).

The F1 scores computed on the TimeBank-Dense test-set with the additional event attribute features are given in the **Ev** column of Table 2. Each machine-learned sieve increases in F1, but the overall architecture *decreases* slightly. This highlights the non-monotonic relationship between the performance of individual sieves and the performance of the overall architecture.

4.2.2 Semantic Role Labeling

We compute additional features from annotations generated using the *mate-tools* SRL system (Björkelund et al., 2009). Specifically, for a given pair, we compute features representing SRL predicates of the events as well as their conjunction.

[3]The micro-averaged accuracies are equivalent to micro-averaged F1 scores computed on data for which some label is output by a classifier.

[4]The results for the rule-based sieves are not shown, but they match the original system exactly.

Sieve	R	Ev	SRL	W2V	R+	F+
EDCT	.524	.547	.511	.524	.553	.553
ETWS	.414	.450	.414	.450	.443	.480
EEWS	.442	.466	.424	.450	.450	.456
EED	.428	.500	.435	.473	.488	.466
Full	**.502**	**.495**	**.493**	**.504**	**.520**	**.527**

Table 2: Micro-averaged accuracies on the TimeBank-Dense test-set for machine-learned sieves and the fully replicated architecture with various feature extensions. Results are given for our baseline CAEVO replication (**R**) and extensions with gold-standard event attribute features (**Ev**), SRL features (**SRL**), word embedding features (**W2V**), all new features (**R+**), and all new features with new feature count cutoffs (**F+**).

Also, we compute the shortest path between a pair within the undirected graph formed by the SRL predicates and arguments (i.e. where there are nodes for predicate spans and argument spans, and there is an edge between two spans if one is the argument of the other). As shown in Table 2 under the **SRL** column, these features only give a minor improvement in micro-averaged F1 for the EED sieve, but hurt performance of the other sieves and the architecture on the TimeBank-Dense test set. However, we believe this may be due to overfitting, as we observe 3% and 4% gains for the ETWS and EED sieves with these features on the development set.

4.2.3 Word Embeddings

Given that recent work has shown improvements using word embeddings (using log-linear neural language models such as the Skip-Gram architecture), we extend feature vectors with the word vectors representing events and their similarity. Following Mirza and Tonelli (2016), we use the three million 300-dimensional *word2vec* vectors [5] pre-trained on part of the Google News dataset (Mikolov et al., 2013). For each token span corresponding to either an event mention or time expression in a given pair datum, we extend the feature vector with normalized sums of word vectors computed from tokens of the span. In addition, we include the cosine similarity between the vectors for the events in a pair, as well as a vector representing the normalized difference between the pair's vectors. Micro-averaged F1 scores on the TimeBank-Dense test set with these word embedding features are given in the **W2V** column of Table 2. The ETWS and EED sieves show improvements of more than 3%, and the EEWS shows a gain of about 1%. However, the F1 score for the overall architecture remains nearly the same.

4.2.4 Full Extension

We extend the machine learned sieves with the full set of event attributes, SRL, and word embedding features as described above. As shown under the **R+** column of Table 2, this yields a 2% gain in micro-averaged F1 for the overall architecture as well as gains for each individual sieve. Also, Section 4.1 mentioned that the feature count cutoffs in **R** and **R+** are set to 1 for EDCT and 2 for all remaining machine-learned sieves. For simplicity, we set the cutoff to 1 across all sieves in **F+**, yielding a 4% improvement in ETWS and minor gains in EEWS and the full system over **R+**. Overall, our feature engineering efforts give **F+** a 5% relative gain (2.5% absolute) over the replicated CAEVO architecture (**R**).

4.3 Modifying Sieve Inference

This section proposes new inference methods for sieve architectures by varying the scoring function s and adding unlabeled data to D from Algorithm 1 and Equation 1 in Section 3. This is a core contribution that can benefit not just temporal ordering, but also other sieve systems applied to other NLP tasks.

4.3.1 Alternative Scoring Methods

In the original CAEVO architecture's implementation of Algorithm 1 from Section 3, the score $s(d, \hat{f}(d), \hat{f})$ is computed as the precision of the sieve $\hat{f}$ on the development set. This greedy scorer s gives a coarse-grain ranking of sieve predictions, assigning equal precedence to all predictions from a given sieve $\hat{f}$. Intuitively, if we want to produce a higher accuracy architecture, then we should adjust the scoring function s to score all correct predictions more highly than all incorrect predictions[6]. CAEVO's use of $\hat{f}$ precision in computing s is a coarse-grained heuristic in line with this goal, but there are better choices.

Ideal Scorer In the best case, the **F+*** column of Table 3 shows the micro-averaged F1 (equivalent to accuracy) when s scores a prediction as

[5]Pre-trained word vectors can be retrieved from https://code.google.com/archive/p/word2vec/.

[6]Note that while such a choice of s should produce good performance, this performance is not necessarily optimal under the transitivity constraints.

Data	V	CAEVO (R)	F+	F+L	F+S	F+LU	F+*
Dev	.378	.481	.485	**.490**	.481	.491	.585
Test	.403	.502	.527	**.546**	.521	.541	.642

Table 3: Micro-averaged F1 scores on the original TimeBank-Dense train-dev-test split for several versions of the sieve architecture. Results are given for the VAGUE majority baseline (**V**), the CAEVO replication (**R**), the architecture with the extended feature set (**F+**), and varying inference methods under the extended feature set. The varying inference methods include an alternative prediction scoring function s computed by precision of each sieve on each relation label (**F+L**), s computed by precision multiplied by classifier probability estimates (**F+S**), s computed by precision on each label with extra unlabeled data (**F+LU**), and s computed to produce near-optimal ordering (**F+***).

$s(d, \hat{f}(d), \hat{f}) = 1$ if $\hat{f}(d)$ is the correct label for datum d, and 0 otherwise. This near-optimal choice of s in **F+*** gives a 10% gain over **F+**, suggesting a large room for improvement by reranking sieve predictions rather than improving the accuracy of the individual sieves. This suggests that architecture performance will increase by improving the estimates of the prediction *confidence* encoded by s, rather than improving the predictions themselves.

New Scorers We thus consider several alternatives for s. First, we attempted estimating s by training a reranking logistic regression model to predict whether $\hat{f}(d)$ is the correct label for d within prediction $(d, \hat{f}(d), \hat{f})$. This approach did not improve performance over other simpler approaches (possibly due to the small size of reranking training data), and so we only report results for the simpler approaches. In one approach, motivated by the observation that precision varies across relation labels, we compute $s(d, \hat{f}(d), \hat{f})$ as the precision of $\hat{f}$ for predictions with label $\hat{f}(d)$ on the dev data. This sieve-label precision approach improves F1 over **F+** as shown in the **F+L** column of Table 3. In a second approach, we compute $s(d, \hat{f}(d), \hat{f})$ as the precision of $\hat{f}$ multiplied by the probability assigned to $\hat{f}(d)$ by the logistic regression model employed by $\hat{f}$. According to the **F+S** column of Table 3, this approach does not show improvement over **F+**.

4.3.2 Leveraging Unlabeled Data

CAEVO uses Algorithm 1 to draw inferences about a data set D. In the original implementation, this set contained only the gold-standard labeled evaluation pairs within two sentence windows. However, if D were expanded with other unlabeled data points outside of two sentence windows (for which it is easy to predict labels with high precision), the transitivity constraints in C might generate further high precision predictions on the labeled data. Interestingly, this gives the architecture the property that making a larger number of predictions on a logically connected set of data can lead to higher overall performance on subsets of that data. Given this observation, we apply the **F+L** version of the architecture to all pairs of events and times within a document. The resulting F1 scores given this expansion of D with the unlabeled TimeBank-Dense pairs are shown under column **F+LU** of Table 3. Unfortunately, these scores show no improvement over the scores under column **F+L** which suggests that the architecture did not draw high precision inferences from the unlabeled data to labeled data. This may be due to the lack of sieves tuned specifically to make between-sentence unlabeled data predictions, or it may be due to an inherent difficulty in making these predictions over the labeled within-sentence and consecutive sentence predictions.

5 Deep Dive into the Data

One of the difficulties facing the temporal ordering community is sparse data. This has been an issue since the original TimeBank Corpus, and the TimeBank-Dense Corpus had data expansion as one of its core goals. However, we argue that data sparsity is still a problem, and previous work tends not to explore different test sets, potentially misidentifying positive and negative results specific to particular splits of the data. This issue seems especially relevant due to the small size of the TimeBank-Dense data (only 5 documents in dev and 9 documents in test for the original split (Chambers et al., 2014)).

The underlying question is whether new results present a significant improvement upon older ones. We consider multiple cross-validation splits of the data to get some sense about the answer to this question. We chose this approach over null-hypothesis significance testing due to limitations induced by small sample size in conjunction with the dependencies between predictions arising through the transitivity constraints. These two issues render it difficult to make a hard determination of significance in a way that does not violate hypothesis testing assumptions. Instead, our cross-validation splits give a weak qualitative sense of the generalizability of our methods.

Our cross validation setup consisted of four splits of the 36 documents with 5 documents per test set and 4 documents per development set for each split. Table 4 shows the micro-averaged F1 scores of the architectures described in the previous sections on each of these splits. Unsurprisingly, the results show that some architecture scores were boosted while others lessened on these alternative splits. Notably, the added features in **F+** still make consistent gains over the CAEVO replication **R**, and the ideal scorer **F+*** makes consistent large gains (as high as 18% on Fold 3, and a low of 6% on Fold 1). However, **F+L** performs well on the development sets but does not improve performance on the test sets. We hypothesize that **F+L** overfits to the development sets due to their small size and the small number of predictions available to compute sieve-label precisions. The consistently large gains of **F+L** on the dev sets suggest that the method will achieve high performance as long as the precision estimates in s are accurate, but the method requires more development data than **F+** to compute accurate estimates without overfitting due to the large number of sparsely distributed sieve-label combinations.

This extra analysis helps to highlight the potential for overfitting. We hope this encourages future work to bear this in mind, and to also present results across multiple tests. Extra evaluations like those in this section are often unexciting, but **we argue that it is of utmost importance that they are conducted**. This paper could have ended at the previous section's top test set results, but we hope the reader sees extra value in the deep analysis of one's results.

6 Discussion

In the above experiments, we successfully replicated the CAEVO system from Chambers et al. (2014), and then proposed a generalization of the sieve-based architecture that enabled several new extensions and improvements. With the injection of new features, we improve the overall system with a **5%** relative gain in F1. Furthermore, our generalized version of CAEVO's sieve architecture allowed us to score and rank predictions based on both label and sieve precision, raising the F1 results to an **8.8%** relative improvement over our replicated CAEVO (under **F+L** in Table 3). We consider these results a new state-of-the-art on the TimeBank-Dense corpus. In addition, the large gains using a near-optimal scoring function (under **F+*** in Table 3) suggest that future work might make substantial progress by building further alternative prediction scoring methods.

We also perform an in-depth analysis of our improvements on alternative splits of the data. Through this analysis, we find that our feature engineering results are robust. More interestingly, while the **F+L** scoring method gives increased F1 on the original TimeBank-Dense split and all cross-validation dev sets, it does not yield improved performance on our alternative cross-validation test sets. This analysis suggests significant improvement for **F+L** over the original architecture, but with possible overfitting label-specific precision estimates on our small amount of development data.

Finally, we presented the first experiments that leveraged unlabeled data. These experiments gave negative results, but we believe future research might see improvements through inference over unlabeled data by (1) improving the precision of unlabeled data predictions (through the incorporation of precise between-sentence prediction sieves), (2) increasing the density of the unlabeled data (e.g. by including easy-to-predict cross-document links between related events), (3) increasing the number of constraints across the data through the incorporation of sieves for additional tasks like event and entity coreference (coref), or (4) increasing the size of the data for more reliable evaluation and training. With respect to (4), we hypothesize that although Timebank-Dense contains more temporal relations than other temporal corpora, it is still small in size. Our hope for future work is to extend the data set with more dense annotations, but spread across a larger number of document contexts, such that different scoring and inference methods may be robustly trained and evaluated.

Split	Data	V	CAEVO (R)	F+	F+L	F+S	F+LU	F+*
Fold 0	Dev	.385	.574	.571	**.596**	.593	.596	.676
	Test	.400	.503	**.535**	.534	.530	.534	.632
Fold 1	Dev	.435	.519	.537	**.592**	.543	.592	.663
	Test	.312	.443	.503	.501	**.506**	.501	.558
Fold 2	Dev	.450	.522	.542	**.555**	.521	.516	.684
	Test	.462	.528	.540	.507	**.541**	.506	.706
Fold 3	Dev	.436	.536	.562	**.575**	.535	.573	.664
	Test	.470	.484	.497	**.500**	**.500**	.497	.682

Table 4: Micro-averaged F1 scores on four cross-fold validation train-dev-test splits of TimeBank-Dense for the sieve architectures defined in Table 3. The new features in **F+** make conistent gains across folds, and the ideal scorer **F+*** demonstrates consistently large room for improvement using alternative scoring methods. The **F+L** model still performs well on the dev sets, but it gives no performance gains on test sets. This suggests the danger of overfitting the scoring functions s within sieve architectures, as the sieve-label precision scores use in **F+L** were computed over a small, four document dev set in each fold.

In sum, we present a new state-of-the-art event ordering model. Furthermore, we propose a generalized approach to classifier ranking that is applicable to all sieve architectures (not just temporal ordering). Instead of producing coarse-grained ranking of classifier predictions, our proposal selects more fine-grained, higher performing prediction rerankings. In addition, we show temporal ordering gains using SRL and word embedding features. The code for our event-ordering architectures and experiments is publicly available[7]. We hope that this work will encourage further efforts in dense event ordering research.

Acknowledgments

We thank Jesse Dodge and Noah Smith for advising and support with preliminary experiments that gave several negative results prior to the work in this paper, helping to prune likely dead-end research directions. We also thank the RTW group at CMU for allowing us to use their library repositories. The work of authors BM and DR was funded under NSF grant SES-1528409. NC was partially supported by a grant from the Office of Naval Research.

References

Steven Bethard. 2013. Cleartk-timeml: A minimalist approach to tempeval 2013. In *Second Joint Conference on Lexical and Computational Semantics (* SEM)*. volume 2, pages 10–14.

Steven Bethard and James H Martin. 2007. Cu-tmp: Temporal relation classification using syntactic and semantic features. In *Proceedings of the 4th International Workshop on Semantic Evaluations*. Association for Computational Linguistics, pages 129–132.

Anders Björkelund, Love Hafdell, and Pierre Nugues. 2009. Multilingual semantic role labeling. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning: Shared Task*. Association for Computational Linguistics, pages 43–48.

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*. Association for Computational Linguistics, pages 189–198.

Taylor Cassidy, Bill McDowell, Nathanel Chambers, and Steven Bethard. 2014. An annotation framework for dense event ordering. Technical report, DTIC Document.

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*. Association for Computational Linguistics, pages 698–706.

Yuchang Cheng, Masayuki Asahara, and Yuji Matsumoto. 2007. Naist. japan: Temporal relation identification using dependency parsed tree. In *Proceedings of the 4th International Workshop on Seman-*

[7] https://github.com/forkunited/CAEVO-plus

tic Evaluations. Association for Computational Linguistics, pages 245–248.

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*. Association for Computational Linguistics, pages 677–687.

Jennifer D'Souza and Vincent Ng. 2013. Classifying temporal relations with rich linguistic knowledge. In *HLT-NAACL*. pages 918–927.

Jennifer D'Souza and Vincent Ng. 2015. Sieve-based spatial relation extraction with expanding parse trees. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. pages 758–768.

Daniel Gildea and Daniel Jurafsky. 2002. Automatic labeling of semantic roles. *Computational linguistics* 28(3):245–288.

Oleksandr Kolomiyets, Steven Bethard, and Marie-Francine Moens. 2012. Extracting narrative timelines as temporal dependency structures. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers-Volume 1*. Association for Computational Linguistics, pages 88–97.

Egoitz Laparra, Itziar Aldabe, and German Rigau. 2015. Document level time-anchoring for timeline extraction. *Volume 2: Short Papers* page 358.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *Nature* 521(7553):436–444.

Heeyoung Lee, Angel Chang, Yves Peirsman, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. 2013. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *Computational Linguistics* 39(4):885–916.

Heeyoung Lee, Marta Recasens, Angel Chang, Mihai Surdeanu, and Dan Jurafsky. 2012. Joint entity and event coreference resolution across documents. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*. Association for Computational Linguistics, pages 489–500.

Heeyoung Lee, Mihai Surdeanu, and Dan Jurafsky. 2017. A scaffolding approach to coreference resolution integrating statistical and rule-based models. *Natural Language Engineering* pages 1–30.

Hector Llorens, Estela Saquete, and Borja Navarro. 2010. Tipsem (english and spanish): Evaluating crfs and semantic roles in tempeval-2. In *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, pages 284–291.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* .

Anne-Lyse Minard, Manuela Speranza, Eneko Agirre, Itziar Aldabe, Marieke van Erp, Bernardo Magnini, German Rigau, Ruben Urizar, and Fondazione Bruno Kessler. 2015. Semeval-2015 task 4: Timeline: Cross-document event ordering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. pages 778–786.

Paramita Mirza and Sara Tonelli. 2016. On the contribution of word embeddings to temporal relation classification. In *The 26th International Conference on Computational Linguistics*. pages 2818–2828.

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.

Marta Tatu and Munirathnam Srikanth. 2008. Experiments with reasoning for temporal relations between events. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*. Association for Computational Linguistics, pages 857–864.

Naushad UzZaman and James F Allen. 2010. Trips and trios system for tempeval-2: Extracting temporal information from text. In *Proceedings of the 5th International Workshop on Semantic Evaluation*. Association for Computational Linguistics, pages 276–283.

Naushad UzZaman, Hector Llorens, James Allen, Leon Derczynski, Marc Verhagen, and James Pustejovsky. 2012. Tempeval-3: Evaluating events, time expressions, and temporal relations. *arXiv preprint arXiv:1206.5333* .

Marc Verhagen, Robert Gaizauskas, Frank Schilder, Mark Hepple, Graham Katz, and James Pustejovsky. 2007. Semeval-2007 task 15: Tempeval temporal relation identification. In *Proceedings of the 4th International Workshop on Semantic Evaluations*. Association for Computational Linguistics, pages 75–80.

Marc Verhagen, Roser Sauri, Tommaso Caselli, and James Pustejovsky. 2010. Semeval-2010 task 13: Tempeval-2. In *Proceedings of the 5th international workshop on semantic evaluation*. Association for Computational Linguistics, pages 57–62.

Katsumasa Yoshikawa, Sebastian Riedel, Masayuki Asahara, and Yuji Matsumoto. 2009. Jointly identifying temporal relations with markov logic. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1-Volume 1*. Association for Computational Linguistics, pages 405–413.

A Replication Details

While replicating CAEVO, we did not find any major issues that significantly change the results reported in the original paper. However, we found the following minor bugs: (1) bias features are included in the EEWS, EED, and ETWS machine-learned sieves but not in EDCT, (2) the computation of dependency path features does not always compute the shortest paths, and (3) code that computes token paths is specified to only compute for paths with length less than 4, but does not do this correctly. In our replicated version, we remove each of these bugs.

We also noticed the following quirks in the original system:

- Features based on gold-standard event tense, aspect, and class were not included in the machine-learned models that produced the reported results even though they were described in the original CAEVO paper.
- The EEWS, EED, and ETWS sieves were trained using feature matrices with a minimum feature occurrence count of 2 across training data whereas EDCT has a minimum feature occurrence count of 1. We know of no motivation for setting this parameter differently for the EDCT sieve, but resetting it to 2 within EDCT drops its performance to below the "All Vague" baseline sieve, resulting it from it being effectively removed from the system, and yielding a 5% drop in performance. The sensitivity of the overall system's performance to this parameter setting highlights the importance of using enough data to acquire accurate precision estimates to determine the prediction scoring.
- The EED sieve had a lower precision estimate than EEWS, but EEWS makes predictions on a superset of the event pairs for which EED makes predictions. This means that EED has no functional relevance with respect to the performance of the original architecture.

The replication process also revealed the sensitivity of the results to the details of feature engineering and feature selection. Overall, the process confirmed that minor flaws and oddities will likely remain in complicated architectures like CAEVO after they have been documented, and it can be worthwhile to repeatedly inspect and replicate these systems to ensure that they function as specified.