CS 224 U: Literature review

Aju Thalappillil Scaria, Rishita Anubhai, Rose Marie Philip
{ajuts, rishita, rosep}@stanford.edu

Task definition

The web has an enormous amount of information stored as text content. While this information is accessible to anyone with an internet connection, analyzing this vast amount of data to extract meaningful information is rather difficult because of its sheer volume. Designing automated mechanisms to parse the content of the web to build automatic question answering systems has been an area of research for many years. For instance, extracting events and the associated entities from biomedical and molecular biology text has drawn a lot of attention, especially because of the introduction of BIONLP tasks targeting fine-grained information extraction. This has also been motivated by the increasing number of electronically available publications stored in databases such as PubMed.

The task of event and entity extraction encompasses several areas including natural language processing, computational linguistics and text mining. This task would in turn help other popular NLP tasks such as question answering and machine translation. Different methods that make use of syntactic and dependency parsing, optimization techniques, semantic role labeling and machine learning techniques have been used over the years to tackle this challenging problem. While some of the systems built have had significant improvement over the previously existing ones, the performance of the state-of-the-art systems is still quite distant to that of a 'perfect' extraction system. This is mostly because interpreting meaning from complex structures of natural language is a hard task. In this paper, we review nine research publications from the area of event and entity extractions and temporal ordering of events. In the first section, we summarize the content of the literature reviewed and highlight their main design characteristics and contributions. Then, we compare the different approaches taken in the papers to compare and contrast the different strategies that have been adopted, their similarities and differences.

Summaries of papers reviewed

1. Unsupervised Learning of Narrative Event Chains - Chambers and Jurafsky (2008)[Event Chains].

This paper mainly talks about unsupervised event identification and temporal event ordering. The authors follow the intuition that events related to a common protagonist (the common argument for the event verbs) should form a valid narrative event chain. They do not pay much attention to the type or role of the protagonist when grouping events with a common protagonist together. Hence, no heuristic involving the role and type of the protagonist is used to get a richer event extraction. This paper constructs a novel scoring method for predicting the most likely next event in a chain that involves two kinds of scores - local pairwise event scores and global narrative chain scores.

The second major goal in the paper was to perform temporal ordering of events. The first stage for this was to use a temporal classifier that would label the temporal attributes of the events with tense and grammatical aspect information. In the second stage, features such as event-event syntactic properties, bigram event tense, bigram grammatical aspects and sentence location of the events are used to classify the temporal relationship between two events. The paper just explores the establishment of the *before* relationship in case of temporal order identification. Finally, to compare the outcome of event identification and temporal ordering, with scripts (supervised hand labeled methods for event extraction), the paper suggests discretizing the narrative event chains. The discretization can be done by first performing agglomerative clustering of events based on the scores discussed above and then ordering these clusters by the temporal ordering method. Thus the main takeaways from this paper are intuitions behind performing event identification, temporal ordering and finally merging these two to generate script-like structures as event extractions (all by unsupervised methods).

2. Unsupervised Learning of Narrative Schemas and their Participants - Chambers and Jurafsky (2009).

This paper addresses narrative event chains in further detail. In the broader perspective, it addresses the areas of event identification and argument extraction listed above. Moving on from identifying single event chains, they talk about overlapping these single chains to form a narrative event schema. Argument extraction is also done with unsupervised semantic role labeling (without the need of a predefined class of roles or hand built domain knowledge). These two tasks also contribute to the improvement of each other since rich event extraction largely revolves around extracting correlated events by identifying coreferring arguments and linking them.

The semantic role labeling task that the paper takes up, is done in an unsupervised manner and by learning the roles automatically rather than picking them from a pool of predefined domain of roles. Once these labels are assigned, they can then construct narrative event chains, where the events have arguments with a type and role. They suggest that the types may also be picked by finding the most frequent headword in co-referential chains for the arguments.

If narrative chains are formed only based on the frequency with which two verbs share their arguments, ignoring the features of the arguments themselves, then it is likely that word sense ambiguities cause a comparatively worse event to be clustered with other events. Instead the paper recommends having the aforementioned attached types and roles for the arguments and modeling argument overlaps across all (event, argument) pairs to extract most likely next event in the narrative schema. Using this richer representation of event chains, they are combined into narrative schemas by scores depicting chain similarities and narrative similarities.

3. Template-Based Information Extraction without the Templates - Chambers and Jurafsky (2011).

The main aim of this paper is to devise a novel approach to information extraction, which uses methods to induce templates (structures which have slots, events and arguments to describe raw textual information). This links to our topic of event extraction since, template based information extraction can easily be extended to event extraction as templates and event structures are often similar. The major novelty of the paper lies in the method introduced in the paper to learn the template structure from an unlabeled corpus. The main positive of this paper over other approaches is that redundant documents about specific events are not needed to perform template induction. Also, templates with any type and any number of slots can be filled in an unsupervised fashion. After this template induction, event extraction draws from these templates.

The main task described is to extract information from a domain specific corpus by learning a rich template structure. The method for this involves

- Clustering the event patterns in the domain to estimate template topics using LDA and agglomerative clustering based on word distance and PMI over all event patterns.
- Building a new corpus specific to each cluster by retrieving documents from a larger unrelated corpus.
- Inducing semantic role labels for each of the slots in the template using the larger corpus of documents.

The paper then talks about evaluation using two metrics - overall template evaluation as well as a stricter per template evaluation. Moreover, the results have comparable precision and an F1 score that approaches existing algorithms which rely heavily of prior knowledge of the domain.

Since this approach is unsupervised, the recall is hurt. Also since event extraction here happens as an after stage of template induction, the number of parameters required are high and could be reduced in future work. Other than pointing out future prospects for these improvements, in the larger picture the paper draws out an interesting way of event extraction as a step on top of template induction, where the latter is done in an unsupervised manner for learning the templates, slots and the roles.

4. Extracting Complex Biological Events with Rich Graph-Based Feature Sets - Bjorne et al.(2009).

This paper describes a system for extracting events among genes and proteins from biomedical literature. They divide the task of event extraction into three independent steps, each considered as a machine learning problem. They claim that by separating trigger recognition from argument detection, they can use methods like named entity recognition to tag words as entities. But, they assume that all entities are recognizable as named entities and are available already. In their model, the sentence or paragraph from which events are to be extracted is represented as a graph with nodes corresponding to entities and events, and the edges corresponding to event-argument relationships. This is a natural representation as it is easy to encode event-event relations as well.

As the first step, they tackle the problem of trigger detection as a token labeling problem in which each token is assigned to an event class if it is likely to be an event otherwise, to the negative class. Multi-class SVM is used for the classification task. Token features (related to properties of individual tokens like capitalization), frequency features (for e.g., number of entities in the sentence), and dependency chain features (encoding dependencies between tokens) are used.

In the next stage, they have edge detection, modeled again as a multi-class SVM to tag each potential edge between an event and an entity; or an event and an event representing a relation between them. The edge is tagged as *theme, cause* or a negative denoting the absence of an edge. The features they use include N-grams (generated by merging attributes of 2-4 consecutive tokens), individual component features, semantic node features (obtained from just the two nodes) and frequency features. In this case, the edges between different nodes (entities and events) are predicted independent of each other. Because of this, they need a third stage in their pipeline to do semantic post processing to ensure that the semantic graph produced by the trigger and edge detection steps does not have any improper combinations. This is a rule based step which refines the graph based on event-event and event-argument types.

5. Joint Learning Improves Semantic Role Labeling - Toutanova et al..

In this paper, a joint model to improve semantic role labeling is presented. In their support, they claim that there are tight dependencies between the different entities and their arguments, because of which, building independent classifiers to handle the task of event extraction assumes a lot of independence which does not really exist in natural language. For instance, there may be hard constraints according to which arguments to events (or predicates) cannot overlap with each other, and also soft constraints by which a predicate cannot have two or more agents as arguments.

In this paper, they present a joint model for semantic role labeling using global features to achieve better performance as compared to the state of the art models. In their model, they had two sets of models - *local models* that learn to role-label nodes in the parse tree independently, and models that incorporate dependencies among the labels of multiple nodes, called *joint models*.

The local classifier handles the task of semantic role labeling as two separate tasks of identification and classification as they decompose well making it possible to solve them separately. In the identification phase, each node of the parse tree created from the sentence is tagged as an argument or not, and, in the classification phase, each argument is associated with its appropriate semantic role. In order to satisfy non-overlapping constraints of labeling the parse tree nodes, they devised a dynamic programming approach that works bottom up by picking the best labeling.

Since the local models don't capture the dependencies amongst arguments, they also have a joint model which builds on top of assignments generated by the local model. Their model also does re-ranking of the assignments generated by the local model based on weights learnt from a log-linear re-ranking model. For this, they map the features from a parse tree and the label sequence to a vector space and then maximize the log likelihood of the best assignments. Re-ranking is similar to the dynamic programming approach for satisfying non-overlapping constraints, and is required to incorporate long range dependencies.

One important factor that helps their model perform well is the usage of templates to generate features. A template helps capture dependencies between label of a node and input features of other argument nodes - for instance, templates could be used to avoid picking multiple arguments to fill the same semantic role (agent of the verb for instance). In short, jointly modeling the arguments of verbs do help.

6. Fast and Robust Joint Models for Biomedical Event Extraction - Riedel and McCallum (2008).

This paper introduces three joint models of increasing complexity designed to extract bio-medical events. They formulate the search for event structures as an optimization problem by first constructing a graph structure over tokens and then introducing binary variables for relationships.

For a sentence and a set of candidate trigger tokens of the sentence, they label each candidate with an event type, or 'None' if it is not a trigger. Hence, for a candidate trigger, there are as many binary variables as the number of possible event types + 1. Let these variables be denoted by e. For each candidate trigger, they introduce binary variables a to associate the trigger to all arguments (events or entities) of the event to which the trigger belongs to. These variable are used to add association between event-event and event-entity pairs. This representation has the shortcoming that it is not possible to differentiate between two events with the same trigger but different arguments, or, one event with several arguments. To overcome this, Riedel and McCallum (2008) augmented the graph representation by introducing another binary variable b to express relationships between arguments.

As mentioned before they have three progressive models for event extraction. The first model does joint trigger and argument extraction (by learning the assignment variables e and a) by an efficient exact inference algorithm by independently scoring trigger labels and argument roles and maximizing the sum of the scores both by using a scoring function. Model 1 can predict structures that cannot be mapped to events. For instance, it can label a token that is actually an event as an argument to another event, but, the former event may not be an active trigger or argument. This would not be ideal as we would want events to combine only with events or arguments. Model 2 enforces these constraints by using a scoring function to identify consistent trigger labels and incoming edges. Model 3 predicts the variables that denote argument-argument combinations as described earlier by using a scoring function to learn weights for the binary variable b.

7. Event Extraction as Dependency Parsing - McCloskly et al.(2011).

This paper introduces the task of event extraction from text by means of dependency parsing. Even though nested event structures are common occurrences, most previous models were incapable of handling them, as events and arguments were extracted independently. In this paper, they propose extracting events (including nested events) by taking the tree of event-argument relations and using it directly as a representation in a reranking dependency parser. The entities in this task were a part of the dataset and were provided, but the event anchors were predicted by a multi-class SVM classifier.

In the first phase, the original event representation is converted to dependency trees containing event anchors and entity mentions. Each event anchor is linked to each of its arguments and is labeled with the slot name of the argument. The labeled dependency links were generated using MSTParser. In this model, the graph may contain self-referential edges due to related events sharing the same anchor. Since their re-ranking algorithm would work only on trees, they had several preprocessing steps. They removed self-referential edges and also broke structures where one argument participates in multiple events by keeping only the dependency to the event that appears first in text. Also, all events with same types anchored on the same anchor phrase was unified.

After this phase, the resulting dependency structure is in the form of a tree. The MSTParser used features that were edge factored - that is, the features are extracted based on the features of end points of the edge and that of the edge itself. Everytime the MSTParser was run, it finds the top k highest scoring trees and then runs the reranking classifier to incorporate the global properties like event path (path from each node in the event tree upto the root), event frames (event anchors with all their arguments and argument slot names), etc to find the best scoring tree. Since their approach is not restricted by sentence boundaries, it could be extended to work on entire documents.

8. Jointly Combining Implicit Constraints Improves Temporal Ordering - Chambers and Jurafsky (2008)[Temporal ordering].

This paper discusses ways to improve existing work on ordering events in text by making use of relations between events and times, and between the times themselves. Previously, event-event ordering tasks were based on local pairwise decisions using classifiers like SVM using different features of the events. But these could sometimes introduce global inconsistencies when misclassifications occur, that are plainly obvious. For example, if event A occurs before B, B occurs before C, then A cannot occur after C. This paper tries to repair some of these event ordering mistakes by introducing two types of global constraints: transitivity and time expression normalization.

The dataset had newswire articles that are hand-tagged for events, time expressions and relations between events and times. The events are also tagged for temporal information like tense, modality, grammatical aspect etc. The first model for event-event ordering (to *before* and *after*) uses a pairwise classifier between events and a global constraint satisfaction layer (using an integer linear programming framework) that re-classifies certain examples from the first stage if it seems to violate properties like transitivity. These constraints can also help create more densely connected network of events by adding implicit relations that are not labeled. For example, if A occurs before B and B occurs before C, we can add the relation A occurs before C. However it was seen that having this additional layer did not change the overall results. This was because the hand-tagged data had large amount of unlabeled relations and global constraints cannot assist local decisions if the graph is not connected.

So, an addition was made to the model - time-time information that are deduced from logical time intervals. For example, if event A occurred 'last month' and B occurred 'yesterday', we can conclude that A occurred before B because 'last month' occurred before 'yesterday'. Having this additional feature along with the global constraints, greatly increased the size of training data set (81% increase) and also improved the performance (3.6% absolute over pairwise decisions).

9. Joint Inference for Event Timeline Construction - McCloskly et al.(2012).

This paper tries to map events into a timeline representation where each event is associated with a specific absolute time interval of occurrence rather than just inferring the relative temporal relations among the events.

The data is similar to Chambers and Jurafsky (2008)[Temporal ordering] where events in news articles and associated arguments are hand labeled. Four relations between events are considered - *before, after, overlap and no relation*. A time interval is represented in the form [start time, end time]. These intervals are sometimes explicitly mentioned in the text while at other times, it might have to be inferred relative to the document creation time of the article. The model has three steps:

1. Two local pair wise classifiers, one between event mentions and time intervals (E-T) and another between event-event mentions(E-E)
2. A combination step with event coreference (discussed later) to overwrite prediction probabilities in step 1 and
3. A joint inference module that enforces global coherency constraints on the final output of the local classifiers.

Classification in the local level is done using regularized average Perceptron over all possible pairs of event/time mentions using several features like the word, lemma, POS of events mentions, position of entities, tense, type of time interval, etc. Event coreference information is used to enhance the timeline construction performance, because all mentions of a single event overlap with each other and are associated with the same time interval. Also, all mentions of an event have same temporal relation with all mentions of another event. These two properties help avoiding misclassification in a lot of cases. The global inference model combines the local pairwise classifiers through the use of an Integer Linear Programming formulation of constraints. Both E-E and E-T tasks are optimized simultaneously.

The results showed that event coreference improved the performance of the classifier. The performance is better than most of the reported models and having time intervals instead of time points lowers the running time of the algorithm considerably.

Compare and contrast

In our literature review, we cover research papers that broadly fall into two categories. One category deals with event extraction and semantic role labeling including literature that covered biological event extraction for the BIONLP task.The other deals with temporal ordering of events.

The first series of three papers that we chose(Chambers and Jurafsky (2008)[Event Chains], Chambers and Jurafsky (2009), and Chambers and Jurafsky (2011)) broadly deals with one or more of - 'event identification', 'temporal ordering' or 'argument extraction' which they identify as the main parts of event extraction. Narrative event chains, schemas and template based event extractions are the topics that emerge out of these papers for event extraction.

Some important points which come up while comparing papers in this series are:

1. **Cardinality of arguments.** Chambers and Jurafsky (2008)[Event Chains] discusses chaining of events based on just one argument or participant. Chambers and Jurafsky (2009) builds up on that by representing other entities involved in the events as well. This information of recognizing multiple entities can be very valuable, since now it is possible to overlap various event chains and form a larger event scenario or a 'narrative event schema'. The third paper, Chambers and Jurafsky (2011), also performs unsupervised template learning where the number of slots in the templates is inferred automatically without hand labeling or supervision.
2. **Argument Types and Roles.** Chambers and Jurafsky (2008)[Event Chains] also does not pay much attention

to the type or role of the protagonist. Role information such as knowing if the protagonist is a place or a person or an object could help in clustering the events more richly as well. More than mere verb comparison based on exact same argument, role and type information given to arguments in Chambers and Jurafsky (2009) leads to a stronger approach for understanding the event chains and discarding certain candidates while finding the next most likely event. Template based event extraction in Chambers and Jurafsky (2011) also talks of inferring semantic roles for the slots in the template prior to event extraction from the templates.

3. **Approaches to Argument Extraction.** In the third paper in the series, Chambers and Jurafsky (2011), event extraction here is done one step after template induction. Although this is a very systematic approach for information extraction without specific requirements, it might be overkill to use a learning approach with so many parameters and template induction for just event extraction. On the other hand, the first two papers, as mentioned above either use an exact match for arguments (protagonist method) or use semantic role labeling on arguments, without forming anything along the lines of a template.
4. **Clustering Event Patterns.** An important similarity in papers Chambers and Jurafsky (2008)[Event Chains] and Chambers and Jurafsky (2011) is their use of agglomerative clustering for clustering events. In the first paper, event chains are discretized to be comparable with *hand labeled scripts* using agglomerative clustering on the pairwise scores. Along similar lines, Chambers and Jurafsky (2011) talks of clustering events by agglomerative clustering before the event clusters are used to fetch larger corpora and induce template slot roles. This gives us a good idea of how clustering events can be useful to discretize events and partition domains for a higher level perspective.

The work on event and entity extraction have several traits that are common with each other and some others that are different. We comment on this by dividing these differences into three high level categories as follows.

1. **Method of event and argument extraction..** Event and argument extraction is performed either independently (in which choice of an event-entity relation does not affect any others) or jointly (where events and its arguments are extracted jointly). The work by Bjorne et al.(2009) performs event extraction by a pipeline of three independent steps of identifying triggers first, followed by argument extraction and then semantic prost processing. This method is prone to cascading errors introduced in early stages of the pipeline. For instance, if a trigger is missed in the first stage, we will never be able to extract the full event that it results in. Even though this can be tackled by passing several additional candidate to the next stage, this will increase the false positive rate as highlighted by (Miwa et al. 2010c). In addition, these models cannot make use of the rich set of features by looking at how different events and entities interact with each other. Also, the different rule based post-processing steps that need to be used to clean up the event-entity combinations extracted might lead to partially correct relations to be thrown out because of errors introduced in some stage of the pipeline. In short, joint models helps to capture the dependencies better and are more robust. If we look at joint event extraction, there are two common ways in which it is done - by means of semantic role labeling or as a structure prediction problem. Toutanova et al. approaches the problem as a semantic role labeling task while Bjorne et al.(2009) and Riedel and McCallum (2008) solves it as a structure prediction task
2. **Representation of events and entities.** Trees and graphs are two popular structures used to represent events and entities. Both Bjorne et al.(2009) and Riedel and McCallum (2008) use graph structures to encode the event-entity and event-event relationships, although in different ways. In the former, event extraction is done disjointly for edge prediction. The latter uses graphs to generate binary variables as edges denoting entity-entity relationships and solves the problem of event extraction as an optimization problem over binary variables. Riedel and McCallum (2008) and McCloskly et al.(2011) use the parse tree structure quite extensively. For instance, Riedel and McCallum (2008) maps the features from the parse tree directly into vector space. While using parse structures that encode syntax information helps to include important features (relating to grouping and hierarchy of words), working with graph structures appears to be easier to approach as a learning problem.
3. **Solution.** Riedel and McCallum (2008) approaches the task as an optimization problem over binary variables, while Bjorne et al.(2009) uses multiclass SVM classifier for this task. Some others use re-ranking approaches (Toutanova et al. 2005) built on a bottom-up fashion. While, it is not obvious which of these methods works best, each seem to have its on advantages. For instance, a re-ranking dynamic programming approach works well with a tree representation, while a multiclass classifier works well with graph structures.
4. **Use of local and global features.** Most of the papers use global features for entity extraction as event arguments often have dependencies between each other. While some are in the form of post processing rules to ensure consistency, some encode it into the structure of the problem itself, which is easier to optimize.

The three papers Chambers and Jurafsky (2008)[Event Chains], Chambers and Jurafsky (2008)[Temporal ordering], and McCloskly et al.(2012) incrementally extend the work in the area of classification of temporal relations between two events. Some of the main areas common to these papers are as follows:

1. **Types of relations amongst events.** Chambers and Jurafsky (2008)[Event Chains] uses *before* against *others*, Chambers and Jurafsky (2008)[Temporal ordering] adds *after* and McCloskly et al.(2012) adds *overlap* to this set of features.

2. **Local classifier.** Chambers and Jurafsky (2008)[Event Chains] classifies the temporal relations between two events with an SVM model using local features pertaining to the events themselves. The SVM has two stages, one to label temporal attributes of events, and a second stage to classify the temporal relationship between the two events. Chambers and Jurafsky (2008)[Temporal ordering] also uses this kind of a model whereas McCloskly et al.(2012) uses regularized average Perceptron, all three papers using similar features for classification. However it is not clear if one of these is more advantageous than the other. Also, Chambers and Jurafsky (2008)[Temporal ordering] assumes the event-time relation to be given and tries to improve event-event temporal relation classification, whereas the model of McCloskly et al.(2012) jointly optimizes both tasks at the same time.

3. **Extending event relationships in graph.** Transitivity rules were applied to extend the labeling, but it was seen to not improve the results. This was assumed in Chambers and Jurafsky (2008)[Event Chains] to be because some of the documents contained inconsistent labels which created poor transitivity additions. But the second paper overcomes this problem by adding time-time information to the model, which increased the density of the graph by adding relations based on this new information as well. This turned out to be very beneficial.

4. **Timescale.** The timeline representation for ordering events in McCloskly et al.(2012) has some advantages over temporal graph representations in the former papers. The timeline model allows events to be associated with precise time intervals, which improves human interpretability of the temporal relations between events and time. It also simplifies global inference formulations as the number of variables and constraints needed in the Integer Linear Programming is more concise relative to time point based formulation. The comparison of classifiers also shows that McCloskly et al.(2012) could improve the accuracy of Chambers and Jurafsky (2008)[Temporal ordering], so using time intervals could be useful.

5. **Joint classifier.** The two later models use ILP for performing the joint inference with a set of global constraints that enforce global coherency and this is seen to be better than the greedy strategy of adding pairs of events one at a time, ordered by their confidence. This was also useful in avoiding some of the obvious mistakes made due to misclassification by the local classifiers.

Future work

In this paper, we investigated in detail some of the models that have been designed for event and argument extraction, and temporal ordering of events. Even though a lot of work has been done in these fields, we feel there is a lack of focus on extending these models to building automatic question answering systems based on the events extracted. For instance, there is no system that can take a textbook in Biology as input, extracts different high level processes involved, and is capable of answering questions that involve the different events, its prerequisites, entities involved, output, and its temporal relation to other events. This could be done by combining methods discussed in this literature review. One could start by finding events and associated attributes, to build event chains that... After finding these main events, temporal ordering of these is required to determine the order of happening of each event, so one could say which event precedes another event. It might be good to use time intervals to associate each event with a particular interval so that merging different events onto a single timeline would be easy and this would help in finding answers that link related events from different parts of a text book.

Developing such a system would find application in a multitude of areas in this age of information explosion. To start with, it could help students to look for information in text books or encyclopedias without having to go through them page by page. There is also a huge amount of data available in the web, but unless we have an efficient event extraction system, most of it will lie unusable. Of course, we cannot rely fully on the data available from the web, but again, we can use different heuristics to build confidence on the data extracted as we see more supporting evidence.

References

Nathanael Chambers and Dan Jurafsky 2008. Unsupervised learning of narrative event chains. In *Proceedings of ACL-08*.

Nathanael Chambers and Dan Jurafsky 2009. Unsupervised Learning of Narrative Schemas and their Participants In *Proceedings of ACL-09*.

Nathanael Chambers and Dan Jurafsky 2011. Template-Based Information Extraction without the Templates In *Proceedings of ACL-11*.

Jari Bjorne, Juho Heimonen, Filip Ginter, Antti Airola, Tapio Pahikkala, Tapio Salakoski 2009. Extracting Complex Biological Events with Rich Graph-Based Feature Sets. In *Proceedings of the Workshop on BioNLP: Shared Task,*.

Kristina Toutanova, Aria Haghighi, Christopher D. Manning 2005. Joint Learning Improves Semantic Role Labeling. In *Proceedings of ACL 2005*.

Sebastian Riedel Andrew McCallum 2008. Fast and Robust Joint Models for Biomedical Event Extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*.

Makoto Miwa, Rune Saetre, Jin-Dong D. Kim, and Junichi Tsujii 2010c. Event extraction with complex event classification using rich features In *Journal of bioinformatics and computational biology*.

David McClosky, Mihai Surdeanu, and Christopher D. Manning 2011. Event Extraction as Dependency Parsing In *Proceedings of ACL-11*.

Nathanael Chambers and Dan Jurafsky 2008. Jointly Combining Implicit Constraints Improves Temporal Ordering In *EMNLP-08*.

Quang Xuan Do, Wei Lu, Dan Roth 2012. Joint Inference for Event Timeline Construction In *EMNLP-12*.