Event Extraction as Dependency Parsing

David McClosky, Mihai Surdeanu, and Christopher D. Manning
Department of Computer Science
Stanford University
Stanford, CA 94305
{mcclosky,mihais,manning}@stanford.edu

Abstract

Nested event structures are a common occurrence in both open domain and domain specific extraction tasks, e.g., a "crime" event can cause a "investigation" event, which can lead to an "arrest" event. However, most current approaches address event extraction with highly local models that extract each event and argument independently. We propose a simple approach for the extraction of such structures by taking the tree of event-argument relations and using it directly as the representation in a reranking dependency parser. This provides a simple framework that captures global properties of both nested and flat event structures. We explore a rich feature space that models both the events to be parsed and context from the original supporting text. Our approach obtains competitive results in the extraction of biomedical events from the BioNLP'09 shared task with a F1 score of 53.5% in development and 48.6% in testing.

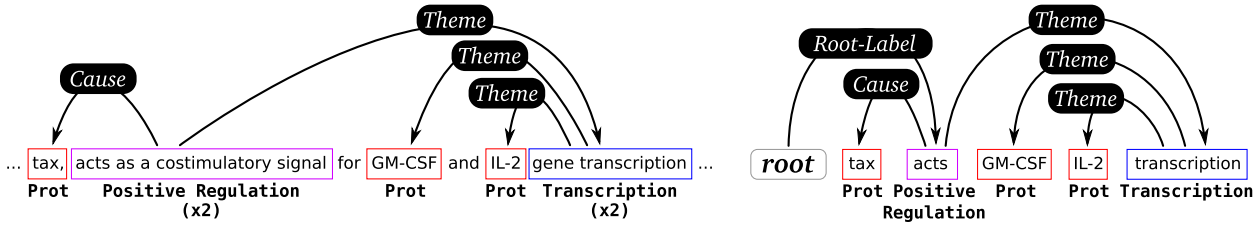
1 Introduction

Event structures in open domain texts are frequently highly complex and nested: a "crime" event can cause an "investigation" event, which can lead to an "arrest" event (Chambers and Jurafsky, 2009). The same observation holds in specific domains. For example, the BioNLP'09 shared task (Kim et al., 2009) focuses on the extraction of nested biomolecular events, where, e.g., a REGULATION event causes a TRANSCRIPTION event (see Figure 1a for a detailed example). Despite this observation, many state-of-the-art supervised event extraction models still extract events and event arguments independently, ignoring their underlying structure (Björne et al., 2009; Miwa et al., 2010b).

In this paper, we propose a new approach for supervised event extraction where we take the tree of relations and their arguments and use it directly as the representation in a dependency parser (rather than conventional syntactic relations). Our approach is conceptually simple: we first convert the original representation of events and their arguments to dependency trees by creating dependency arcs between *event anchors* (phrases that anchor events in the supporting text) and their corresponding arguments.[1] Note that after conversion, only event anchors and entities remain. Figure 1 shows a sentence and its converted form from the biomedical domain with four events: two POSITIVE REGULATION events, anchored by the phrase "acts as a costimulatory signal," and two TRANSCRIPTION events, both anchored on "gene transcription." All events take either protein entity mentions (PROT) or other events as arguments. The latter is what allows for nested event structures. Existing dependency parsing models can be adapted to produce these semantic structures instead of syntactic dependencies. We built a global reranking parser model using multiple decoders from MSTParser (McDonald et al., 2005; McDonald et al., 2005b). The main contributions of this paper are the following:

1. We demonstrate that parsing is an attractive approach for extracting events, both nested and otherwise.

[1] While our approach only works on trees, we show how we can handle directed acyclic graphs in Section 5.



(a) Original sentence with nested events

(b) After conversion to event dependencies

Figure 1: Nested events in the text fragment: "...the *HTLV-1 transactivator protein*, *tax*, **acts as a costimulatory signal** for *GM-CSF* and *IL-2* **gene transcription** ..." Throughout this paper, **bold text** indicates instances of event anchors and *italicized text* denotes entities (PROTEINs in the BioNLP'09 domain). Note that in (a) there are two copies of each type of event, which are merged to single nodes in the dependency tree (Section 3.1).

2. We propose a wide range of features for event extraction. Our analysis indicates that features which model the global event structure yield considerable performance improvements, which proves that modeling event structure jointly is beneficial.
3. We evaluate on the biomolecular event corpus from the the BioNLP'09 shared task and show that our approach obtains competitive results.

2 Related Work

The pioneering work of Miller et al. (1997) was the first, to our knowledge, to propose parsing as a framework for information extraction. They extended the syntactic annotations of the Penn Treebank corpus (Marcus et al., 1993) with entity and relation mentions specific to the MUC-7 evaluation (Chinchor et al., 1997) — e.g., EMPLOYEE OF relations that hold between person and organization named entities — and then trained a generative parsing model over this combined syntactic and semantic representation. In the same spirit, Finkel and Manning (2009) merged the syntactic annotations and the named entity annotations of the OntoNotes corpus (Hovy et al., 2006) and trained a discriminative parsing model for the joint problem of syntactic parsing and named entity recognition. However, both these works require a unified annotation of syntactic and semantic elements, which is not always feasible, and focused only on named entities and binary relations. On the other hand, our approach focuses on event structures that are nested and have an arbitrary number of arguments. We do not need a unified syntactic and semantic representation (but we can and do extract features from the underlying syntactic structure of the text).

Finkel and Manning (2009b) also proposed a parsing model for the extraction of nested named entity mentions, which, like this work, parses just the corresponding semantic annotations. In this work, we focus on more complex structures (events instead of named entities) and we explore more global features through our reranking layer.

In the biomedical domain, two recent papers proposed joint models for event extraction based on Markov logic networks (MLN) (Riedel et al., 2009; Poon and Vanderwende, 2010). Both works propose elegant frameworks where event anchors and arguments are jointly predicted for all events in the same sentence. One disadvantage of MLN models is the requirement that a human expert develop domain-specific predicates and formulas, which can be a cumbersome process because it requires thorough domain understanding. On the other hand, our approach maintains the joint modeling advantage, but our model is built over simple, domain-independent features. We also propose and analyze a richer feature space that captures more information on the global event structure in a sentence. Furthermore, since our approach is agnostic to the parsing model used, it could easily be tuned for various scenarios, e.g., models with lower inference overhead such as shift-reduce parsers.

Our work is conceptually close to the recent CoNLL shared tasks on semantic role labeling, where the predicate frames were converted to se-

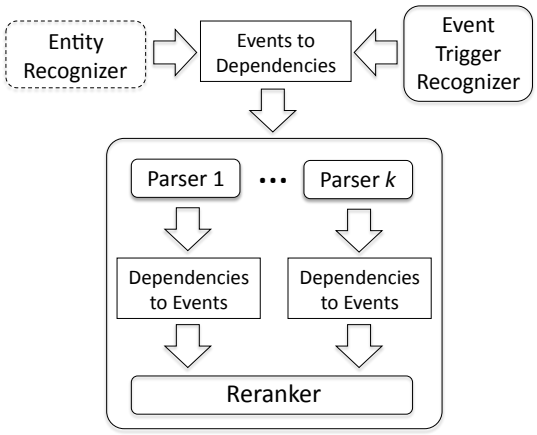


Figure 2: Overview of the approach. Rounded rectangles indicate domain-independent components; regular rectangles mark domain-specific modules; blocks in dashed lines surround components not necessary for the domain presented in this paper.

mantic dependencies between predicates and their arguments (Surdeanu et al., 2008; Hajic et al., 2009). In this representation the dependency structure is a directed acyclic graph (DAG), i.e., the same node can be an argument to multiple predicates, and there are no explicit dependencies between predicates. Due to this representation, all joint models proposed for semantic role labeling handle semantic frames independently.

3 Approach

Figure 2 summarizes our architecture. Our approach converts the original event representation to dependency trees containing both event anchors and entity mentions, and trains a battery of parsers to recognize these structures. The trees are built using event anchors predicted by a separate classifier. In this work, we do not discuss entity recognition because in the BioNLP'09 domain used for evaluation entities (PROTEINS) are given (but including entity recognition is an obvious extension of our model). Our parsers are several instances of MSTParser[2] (McDonald et al., 2005; McDonald et al., 2005b) configured with different decoders. However, our approach is agnostic to the actual parsing models used and could easily be adapted to other dependency parsers. The output from the reranking parser is converted back to the original event representation and passed to a reranker component (Collins, 2000; Charniak and Johnson, 2005), tailored to optimize the task-specific evaluation metric.

Note that although we use the biomedical event domain from the BioNLP'09 shared task to illustrate our work, the core of our approach is almost domain independent. Our only constraints are that each event mention be activated by a phrase that serves as an event anchor, and that the event-argument structures be mapped to a dependency tree. The conversion between event and dependency structures and the reranker metric are the only domain dependent components in our approach.

3.1 Converting between Event Structures and Dependencies

As in previous work, we extract event structures at sentence granularity, i.e., we ignore events which span sentences (Björne et al., 2009; Riedel et al., 2009; Poon and Vanderwende, 2010). These form approximately 5% of the events in the BioNLP'09 corpus. For each sentence, we convert the BioNLP'09 event representation to a graph (representing a labeled dependency tree) as follows. The nodes in the graph are protein entity mentions, event anchors, and a virtual ROOT node. Thus, the only words in this dependency tree are those which participate in events. We create edges in the graph in the following way. For each event anchor, we create one link to each of its arguments labeled with the slot name of the argument (for example, connecting **gene transcription** to *IL-2* with the label THEME in Figure 1b). We link the ROOT node to each entity that does not participate in an event using the ROOT-LABEL dependency label. Finally, we link the ROOT node to each top-level event anchor, (those which do not serve as arguments to other events) again using the ROOT-LABEL label. We follow the convention that the source of each dependency arc is the head while the target is the modifier.

The output of this process is a directed graph, since a phrase can easily play a role in two or more events. Furthermore, the graph may contain self-referential edges (self-loops) due to related events sharing the same anchor (example below). To guarantee that the output of this process is a tree, we must post-process the above graph with the follow-

[2] http://sourceforge.net/projects/mstparser/

ing three heuristics:

Step 1: We remove self-referential edges. An example of these can be seen in the text "the domain interacted preferentially with **underphosphorylated** *TRAF2*," there are two events anchored by the same **underphosphorylated** phrase, a NEGATIVE REGULATION and a PHOSPHORYLATION event, and the latter serves as a THEME argument for the former. Due to the shared anchor, our conversion component creates an self-referential THEME dependency. By removing these edges, 1.5% of the events in the training arguments are left without arguments, so we remove them as well.

Step 2: We break structures where one argument participates in multiple events, by keeping only the dependency to the event that appears first in text. For example, in the fragment "by enhancing its **inactivation** through **binding** to soluble *TNF-alpha receptor type II*," the protein *TNF-alpha receptor type II* is an argument in both a BINDING event (**binding**) and in a NEGATIVE REGULATION event (**inactivation**). As a consequence of this step, 4.7% of the events in training are removed.

Step 3: We unify events with the same types anchored on the same anchor phrase. For example, for the fragment "Surface **expression** of *intercellular adhesion molecule-1*, *P-selectin*, and *E-selectin*," the BioNLP'09 annotation contains three distinct GENE EXPRESSION events anchored on the same phrase (**expression**), each having one of the proteins as THEMEs. In such cases, we migrate all arguments to one of the events, and remove the empty events. 21.5% of the events in training are removed in this step (but no dependencies are lost).

Note that we do not guarantee that the resulting tree is projective. In fact, our trees are more likely to be non-projective than syntactic dependency trees of English sentences, because in our representation many nodes can be linked directly to the ROOT node. Our analysis indicates that 2.9% of the dependencies generated in the training corpus are non-projective and 7.9% of the sentences contain at least one non-projective dependency (for comparison, these numbers for the English Penn Treebank are 0.3% and 6.7%, respectively).

After parsing, we implement the inverse process, i.e., we convert the generated dependency trees to the BioNLP'09 representation. In addition to the obvious conversions, this process implements the heuristics proposed by Björne et al. (2009), which reverse step 3 above, e.g., we duplicate GENE EXPRESSION events with multiple THEME arguments. The heuristics are executed sequentially in the given order:

1. Since all non-BINDING events can have at most one THEME argument, we duplicate non-BINDING events with multiple THEME arguments by creating one separate event for each THEME.

2. Similarly, since REGULATION events accepts only one CAUSE argument, we duplicate REGULATION events with multiple CAUSE arguments, obtaining one event per CAUSE.

3. Lastly, we implement the heuristic of Björne et al. (2009) to handle the splitting of BINDING events with multiple THEME arguments. This is more complex because these events can accept one or more THEMEs. In such situations, we first group THEME arguments by the label of the first Stanford dependency (Marneffe and Manning, 2008) from the head word of the anchor to this argument. Then we create one event for each combination of THEME arguments in different groups.

3.2 Recognition of Event Anchors

For anchor detection, we used a multiclass classifier that labels each token independently.[3] Since over 92% of the anchor phrases in our evaluation domain contain a single word, we simplify the task by reducing all multi-word anchor phrases in the training corpus to their syntactic head word (e.g., "acts" for the anchor "**acts as a costimulatory signal**").

We implemented this model using a logistic regression classifier with L2 regularization over the following features:

[3] We experimented with using conditional random fields as a sequence labeler but did not see improvements in the biomedical domain. We hypothesize that the sequence tagger fails to capture potential dependencies between anchor labels – which are its main advantage over an i.i.d. classifier – because anchor words are typically far apart in text. This result is consistent with observations in previous work (Björne et al., 2009).

- **Token-level:** The form, lemma, and whether the token is present in a gazetteer of known anchor words.[4]
- **Surface context:** The above token features extracted from a context of two words around the current token. Additionally, we build token bigrams in this context window, and model them with similar features.
- **Syntactic context:** We model all syntactic dependency paths up to depth two starting from the token to be classified. These paths are built from Stanford syntactic dependencies (Marneffe and Manning, 2008). We extract token features from the first and last token in these paths. We also generate combination features by concatenating: (a) the last token in each path with the sequence of dependency labels along the corresponding path; and (b) the word to be classified, the last token in each path, and the sequence of dependency labels in that path.
- **Bag-of-word** and **entity count:** Extracted from (a) the entire sentence, and (b) a window of five words around the token to be classified.

3.3 Parsing Event Structures

Given the entities and event anchors from the previous stages in the pipeline, the parser generates labeled dependency links between them. Many dependency parsers are available and we chose MSTParser for its ability to produce non-projective and n-best parses directly. MSTParser frames parsing as a graph algorithm. To parse a sentence, MSTParser finds the tree covering all the words (nodes) in the sentence (graph) with the largest sum of edge weights, i.e., the maximum weighted spanning tree. Each labeled, directed edge in the graph represents a possible dependency between its two endpoints and has an associated score (weight). Scores for edges come from the dot product between the edge's corresponding feature vector and learned feature weights. As a result, all features for MSTParser must be *edge-factored*, i.e., functions of both endpoints and the label connecting them. McDonald et al. (2006) extends the basic model to include second-order dependencies (i.e., two adjacent sibling nodes and their parent). Both first and second-order modes include projective and non-projective decoders.

Our features for MSTParser use both the event structures themselves as well as the surrounding English sentences which include them. By mapping event anchors and entities back to the original text, we can incorporate information from the original English sentence as well its syntactic tree and corresponding Stanford dependencies. Both forms of context are valuable and complementary. MSTParser comes with a large number of features which, in our setup, operate on the event structure level (since this is the "sentence" from the parser's point of view). The majority of additional features that we introduced take advantage of the original text as context (primarily its associated Stanford dependencies). Our system includes the following first-order features:

- **Path:** Syntactic paths in the original sentence between nodes in an event dependency (as in previous work by Björne et al. (2009)). These have many variations including using Stanford dependencies ("collapsed" and "uncollapsed") or constituency trees as sources, optionally lexicalizing the path, and using words or relation names along the path. Additionally, we include the bucketed length of the paths.
- **Original sentence words:** Words from the full English sentence surrounding and between the nodes in event dependencies, and their bucketed distances. This additional context helps compensate for how our anchor detection provides only the head word of each anchor, which does not necessarily provide the full context for event disambiguation.
- **Graph:** Parents, children, and siblings of nodes in the Stanford dependencies graph along with label of the edge. This provides additional syntactic context.
- **Consistency:** Soft constraints on edges between anchors and their arguments (e.g., only regulation events can have edges labeled with CAUSE). These features fire if their constraints are violated.
- **Ontology:** Generalized types of the endpoints of edges using a given type hierarchy (e.g., POSITIVE REGULATION is a COM-

[4] These are automatically extracted from the training corpus.

PLEX EVENT[5] is an EVENT). Values of this feature are coded with the types of each of the endpoints on an edge, running over the cross-product of types for each endpoint. For instance, an edge between a BINDING event anchor and a POSITIVE REGULATION could cause this feature to fire with the values [head:EVENT, child:COMPLEX EVENT] or [head:SIMPLE EVENT, child:EVENT].[6] The latter feature can capture generalizations such as "simple event anchors cannot take other events as arguments."

Both **Consistency** and **Ontology** feature classes include domain-specific information but can be used on other domains under different constraints and type hierarchies. When using second-order dependencies, we use additional **Path** and **Ontology** features. We include the syntactic paths between sibling nodes (adjacent arguments of the same event anchor). These **Path** features are as above but differentiated as paths between sibling nodes. The second-order **Ontology** features use the type hierarchy information on both sibling nodes and their parent. For example, a POSITIVE REGULATION anchor attached to a PROTEIN and a BINDING event would produce an **Ontology** feature with the value [parent:COMPLEX EVENT, child_1:PROTEIN, child_2:SIMPLE EVENT] (among several other possible combinations).

To prune the number of features used, we employ a simple entropy-based measure. Our intuition is that good features should typically appear with only one edge label.[7] Given all edges enumerated during training and their gold labels, we obtain a distribution over edge labels (d_f) for each feature f. Given this distribution and the frequency of a feature, we can score the feature with the following:

$$score(f) = \alpha \times \log_2 \big(freq(f)\big) - H(d_f)$$

The α parameter adjusts the relative weight of the two components. The log frequency component favors more frequent features while the entropy component favors features with low entropy in their edge label distribution. Features are pruned by accepting all features with a score above a certain threshold.

3.4 Reranking Event Structures

When decoding, the parser finds the highest scoring tree which incorporates global properties of the sentence. However, its features are edge-factored and thus unable to take into account larger contexts. To incorporate arbitrary global features, we employ a two-step reranking parser. For the first step, we extend our parser to output its n-best parses instead of just its top scoring parse. In the second step, a discriminative reranker rescores each parse and reorders the n-best list. Rerankers have been successfully used in syntactic parsing (Collins, 2000; Charniak and Johnson, 2005; Huang, 2008) and semantic role labeling (Toutanova et al., 2008).

Rerankers provide additional advantages in our case due to the mismatch between the dependency structures that the parser operates on and their corresponding event structures. We convert the output from the parser to event structures (Section 3.1) before including them in the reranker. This allows the reranker to capture features over the actual event structures rather than their original dependency trees which may contain extraneous portions.[8] Furthermore, this lets the reranker optimize the actual BioNLP F1 score. The parser, on the other hand, attempts to optimize the Labeled Attachment Score (LAS) between the dependency trees and converted gold dependency trees. LAS is approximate for two reasons. First, it is much more local than the BioNLP metric.[9] Second, the converted gold dependency trees lose information that doesn't transfer to trees (specifically, that event structures are really multi-DAGs and not trees).

We adapt the maximum entropy reranker from Charniak and Johnson (2005) by creating a customized feature extractor for event structures — in all other ways, the reranker model is unchanged. We use the following types of features in the reranker:

- **Source:** Score and rank of the parse from the

[5] We define complex events are those which can accept other events are arguments. Simple events can only take PROTEINs.

[6] We omit listing the other two combinations.

[7] Labels include ROOT-LABEL, THEME, CAUSE, and NULL. We assign the NULL label to edges which aren't in the gold data.

[8] For instance, event anchors with no arguments could be proposed by the parser. These event anchors are automatically dropped by the conversion process.

[9] As an example, getting an edge label between an anchor and its argument correct is unimportant if the anchor is missing other arguments.

Decoder(s)	Unreranked			Reranked		
	R	P	F1	R	P	F1
1P	65.6	76.7	70.7	68.0	77.6	72.5
2P	67.4	77.1	71.9	67.9	77.3	72.3
1N	67.5	76.7	71.8	—	—	—
2N	68.9	77.1	72.7	—	—	—
1P, 2P, 2N	—	—	—	68.5	78.2	**73.1**

(a) Gold event anchors

Decoder(s)	Unreranked			Reranked		
	R	P	F1	R	P	F1
1P	44.7	62.2	52.0	47.8	59.6	53.1
2P	45.9	61.8	52.7	48.4	57.5	52.5
1N	46.0	61.2	52.5	—	—	—
2N	38.6	66.6	48.8	—	—	—
1P, 2P, 2N	—	—	—	48.7	59.3	**53.5**

(b) Predicted event anchors

Table 1: BioNLP recall, precision, and F1 scores of individual decoders and the best decoder combination on development data with the impact of event anchor detection and reranking. Decoder names include the features order (1 or 2) followed by the projectivity (P = projective, N = non-projective).

decoder; number of different decoders producing the parse (when using multiple decoders).

- **Event path:** Path from each node in the event tree up to the root. Unlike the **Path** features in the parser, these paths are over event structures, not the syntactic dependency graphs from the original English sentence. Variations of the **Event path** features include whether to include word forms (e.g., "binds"), types (BINDING), and/or argument slot names (THEME). We also include the path length as a feature.
- **Event frames:** Event anchors with all their arguments and argument slot names.
- **Consistency:** Similar to the parser **Consistency** features, but capable of capturing larger classes of errors (e.g., incorrect number or types of arguments). We include the number of violations from four different classes of errors.

To improve performance and robustness, features are pruned as in Charniak and Johnson (2005): selected features must distinguish a parse with the highest F1 score in a n-best list, from a parse with a suboptimal F1 score at least five times.

Rerankers can also be used to perform model combination (Toutanova et al., 2008; Zhang et al., 2009; Johnson and Ural, 2010). While we use a single parsing model, it has multiple decoders.[10] When combining multiple decoders, we concatenate their n-best lists and extract the unique parses.

4 Experimental Results

Our experiments use the BioNLP'09 shared task corpus (Kim et al., 2009) which includes 800 biomedical abstracts (7,449 sentences, 8,597 events) for training and 150 abstracts (1,450 sentences, 1,809 events) for development. The test set includes 260 abstracts, 2,447 sentences, and 3,182 events. Throughout our experiments, we report BioNLP F1 scores with approximate span and recursive event matching (as described in the shared task definition). For preprocessing, we parsed all documents using the self-trained biomedical McClosky-Charniak-Johnson reranking parser (McClosky, 2010). We bias the anchor detector to favor recall, allowing the parser and reranker to determine which event anchors will ultimately be used. When performing n-best parsing, $n = 50$. For parser feature pruning, $\alpha = 0.001$.

Table 1a shows the performance of each of the decoders when using gold event anchors. In both cases where n-best decoding is available, the reranker improves performance over the 1-best parsers. We also present the results from a reranker trained from multiple decoders which is our highest scoring model.[11] In Table 1b, we present the output for the predicted anchor scenario. In the case of the 2P decoder, the reranker does not improve performance, though the drop is minimal. This is because the reranker chose an unfortunate regularization constant during crossvalidation, most likely due to the small size of the training data. In later experiments where more

[10] We only have n-best versions of the projective decoders. For the non-projective decoders, we use their 1-best parse.

[11] Including the 1N decoder as well provided no gains, possibly because its outputs are mostly subsumed by the 2N decoder.

data is available, the reranker consistently improves accuracy (McClosky et al., 2011). As before, the reranker trained from multiple decoders outperforms unreranked models and reranked single decoders. All in all, our best model in Table 1a scores 1 F1 point higher than the best system at the BioNLP'09 shared task, and the best model in Table 1b performs similarly to the best shared task system (Björne et al., 2009), which also scores 53.5% on development.

We show the effects of each system component in Table 2. Note how our upper limit is 87.1% due to our conversion process, which enforces the tree constraint, drops events spanning sentences, and performs approximate reconstruction of BINDING events. Given that state-of-the-art systems on this task currently perform in the 50-60% range, we are not troubled by this number as it still allows for plenty of potential.[12] Björne et al. (2009) list 94.7% as the upper limit for their system. Considering this relatively large difference, we find the results in the previous table very encouraging. As in other BioNLP'09 systems, our performance drops when switching from gold to predicted anchor information. Our decrease is similar to the one seen in Björne et al. (2009).

To show the potential of reranking, we provide oracle reranker scores in Table 3. An oracle reranker picks the highest scoring parse from the available parses. We limit the n-best lists to the top k parses where $k \in \{1, 2, 10, \text{All}\}$. For single decoders, "All" uses the entire 50-best list. For multiple decoders, the n-best lists are concatenated together. The oracle score with multiple decoders and gold anchors is only 0.4% lower than our upper limit (see Table 2). This indicates that parses which could have achieved that limit were nearly always present. Improving the features in the reranker as well as the original parsers will help us move closer to the limit. With predicated anchors, the oracle score is about 13% lower but still shows significant potential.

Our final results on the test set, broken down by class, are shown in Table 4. As with other systems, complex events (e.g., REGULATION) prove harder than simple events. To get a complex event correct, one must correctly detect and parse all events in the event subtree allowing small errors to have large effects. Top systems on this task obtain F1 scores of 52.0% at the shared task evaluation (Björne et al., 2009) and 56.3% post evaluation (Miwa et al., 2010a). However, both systems are tailored to the biomedical domain (the latter uses multiple syntactic parsers), whereas our system has a design that is virtually domain independent.

[12] Additionally, improvements such as document-level parsing and DAG parsing would eliminate the need for much of the approximate and lossy portions of the conversion process.

AD	Parse	RR	Conv	R	P	F1
✓	✓		✓	45.9	61.8	52.7
✓	✓	✓	✓	48.7	59.3	53.5
G	✓		✓	68.9	77.1	72.7
G	✓	✓	✓	68.5	78.2	73.1
G	G	G	✓	81.6	93.4	87.1

Table 2: Effect of each major component to the overall performance in the development corpus. Components shown: AD — event anchor detection; Parse — best individual parsing model; RR — reranking multiple parsers; Conv — conversion between the event and dependency representations. 'G' indicates that gold data was used; '✓' indicates that the actual component was used.

		n-best parses considered			
Anchors	Decoder(s)	1	2	10	All
Gold	1P	70.7	76.6	84.0	85.7
	2P	71.8	77.5	84.8	86.2
	1P, 2P, 2N	—	—	—	86.7
Predicted	1P	52.0	60.3	69.9	72.5
	2P	52.7	60.7	70.1	72.5
	1P, 2P, 2N	—	—	—	73.4

Table 3: Oracle reranker BioNLP F1 scores for our n-best decoders and their combinations before reranking on the development corpus.

5 Discussion

We believe that the potential of our approach is higher than what the current experiments show. For example, the reranker can be used to combine not only several parsers but also multiple anchor recognizers. This passes the anchor selection decision to the reranker, which uses global information not available to the current anchor recognizer or parser. Furthermore, our approach can be adapted to parse event structures in entire documents (instead of in-

Event Class	Count	R	P	F1
Gene Expression	722	68.6	75.8	72.0
Transcription	137	42.3	51.3	46.4
Protein Catabolism	14	64.3	75.0	69.2
Phosphorylation	135	80.0	82.4	81.2
Localization	174	44.8	78.8	57.1
Binding	347	42.9	51.7	46.9
Regulation	291	23.0	36.6	28.3
Positive Regulation	983	28.4	42.5	34.0
Negative Regulation	379	29.3	43.5	35.0
Total	3,182	42.6	56.6	48.6

Table 4: Results in the test set broken by event class; scores generated with the main official metric of approximate span and recursive event matching.

dividual sentences) by using a representation with a unique ROOT node for all event structures in a document. This representation has the advantage that it maintains cross-sentence events (which account for 5% of BioNLP'09 events), and it allows for document-level features that model discourse structure. We plan to explore these ideas in future work.

One current limitation of the proposed model is that it constrains event structures to map to trees. In the BioNLP'09 corpus this leads to the removal of almost 5% of the events, which generate DAGs instead of trees. Local event extraction models (Björne et al., 2009) do not have this limitation, because their local decisions are blind to (and hence not limited by) the global event structure. However, our approach is agnostic to the actual parsing models used, so we can easily incorporate models that can parse DAGs (Sagae and Tsujii, 2008). Additionally, we are free to incorporate any new techniques from dependency parsing. Parsing using dual-decomposition (Rush et al., 2010) seems especially promising in this area.

6 Conclusion

In this paper we proposed a simple approach for the joint extraction of event structures: we converted the representation of events and their arguments to dependency trees with arcs between event anchors and event arguments, and used a reranking parser to parse these structures. Despite the fact that our approach has very little domain-specific engineering, we obtain competitive results. Most importantly, we showed that the joint modeling of event structures is beneficial: our reranker outperforms parsing models without reranking in five out of the six configurations investigated.

Acknowledgments

The authors would like to thank Mark Johnson for helpful discussions on the reranker component and the BioNLP shared task organizers, Sampo Pyysalo and Jin-Dong Kim, for answering questions. We gratefully acknowledge the support of the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0181. Any opinions, findings, and conclusion or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the view of DARPA, AFRL, or the US government.

References

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

Nate Chambers and Dan Jurafsky. 2009. *Unsupervised Learning of Narrative Schemas and their Participants*. Proceedings of ACL.

Eugene Charniak and Mark Johnson. 2005. Coarse-to-fine *n*-best parsing and MaxEnt discriminative reranking. In *Proceedings of the 2005 Meeting of the Association for Computational Linguistics (ACL)*, pages 173–180

Nancy Chinchor. 1997. *Overview of MUC-7*. Proceedings of the Message Understanding Conference (MUC-7).

Michael Collins. 2000. Discriminative reranking for natural language parsing. In *Machine Learning: Proceedings of the Seventeenth International Conference (ICML 2000)*, pages 175–182.

Jenny R. Finkel and Christopher D. Manning. 2009. *Joint Parsing and Named Entity Recognition*. Proceedings of NAACL.

Jenny R. Finkel and Christopher D. Manning. 2009b. *Nested Named Entity Recognition*. Proceedings of EMNLP.

Jan Hajič, Massimiliano Ciaramita, Richard Johansson, Daisuke Kawahara, Maria A. Marti, Lluis Marquez, Adam Meyers, Joakim Nivre, Sebastian Pado, Jan Stepanek, Pavel Stranak, Mihai Surdeanu, Nianwen

Xue, and Yi Zhang. 2009. *The CoNLL-2009 Shared Task: Syntactic and Semantic Dependencies in Multiple Languages*. Proceedings of CoNLL.

Eduard Hovy, Mitchell P. Marcus, Martha Palmer, Lance Ramshaw, and Ralph Weischedel. 2006. *OntoNotes: The 90% Solution*. Proceedings of the NAACL-HLT.

Liang Huang. 2008. Forest reranking: Discriminative parsing with non-local features. In *Proceedings of ACL-08: HLT*, pages 586–594, Association for Computational Linguistics.

Mark Johnson and Ahmet Engin Ural. 2010. *Reranking the Berkeley and Brown Parsers*. Proceedings of NAACL.

Jin-Dong Kim, Tomoko Ohta, Sampo Pyysalo, Yoshinobu Kano, and Jun'ichi Tsujii. 2009. *Overview of the BioNLP'09 Shared Task on Event Extraction*. Proceedings of the NAACL-HLT 2009 Workshop on Natural Language Processing in Biomedicine (BioNLP'09).

Mitchell P. Marcus, Beatrice Santorini, and Marry Ann Marcinkiewicz. 1993. *Building a Large Annotated Corpus of English: The Penn Treebank*. Computational Linguistics, 19(2):313–330.

Marie-Catherine de Marneffe and Christopher D. Manning. 2008. *Stanford Typed Hierarchies Representation*. Proceedings of the COLING Workshop on Cross-Framework and Cross-Domain Parser Evaluation.

David McClosky. 2010. *Any Domain Parsing: Automatic Domain Adaptation for Natural Language Parsing*. PhD thesis, Department of Computer Science, Brown University.

David McClosky, Mihai Surdeanu, and Christopher D. Manning. 2011. Event extraction as dependency parsing in BioNLP 2011. In *BioNLP 2011 Shared Task (*submitted*)*.

Ryan McDonald, Koby Crammer, and Fernando Pereira. 2005. *Online Large-Margin Training of Dependency Parsers*. Proceedings of ACL.

Ryan McDonald, Fernando Pereira, Kiril Ribarov, and Jan Hajič. 2005b. *Non-projective Dependency Parsing using Spanning Tree Algorithms*. Proceedings of HLT/EMNLP.

Ryan McDonald and Fernando Pereira. 2006. *Online Learning of Approximate Dependency Parsing Algorithms*. Proceedings of EACL.

Scott Miller, Michael Crystal, Heidi Fox, Lance Ramshaw, Richard Schwartz, Rebecca Stone, and Ralph Weischedel. 1997. *BBN: Description of the SIFT System as Used for MUC-7*. Proceedings of the Message Understanding Conference (MUC-7).

Makoto Miwa, Sampo Pyysalo, Tadayoshi Hara, and Jun'ichi Tsujii. 2010a. *A Comparative Study of Syntactic Parsers for Event Extraction*. Proceedings of the 2010 Workshop on Biomedical Natural Language Processing.

Makoto Miwa, Rune Saetre, Jin-Dong Kim, and Jun'ichi Tsujii. 2010b. *Event Extraction with Complex Event Classification Using Rich Features*. Journal of Bioinformatics and Computational Biology, 8 (1).

Hoifung Poon and Lucy Vanderwende. 2010. *Joint Inference for Knowledge Extraction from Biomedical Literature*. Proceedings of NAACL.

Sebastian Riedel, Hong-Woo Chun, Toshihisa Takagi, and Jun'ichi Tsujii. 2009. *A Markov Logic Approach to Bio-Molecular Event Extraction*. Proceedings of the Workshop on BioNLP: Shared Task.

Alexander M. Rush, David Sontag, Michael Collins, and Tommi Jaakkola. 2010. On dual decomposition and linear programming relaxations for natural language processing. *Proceedings of EMNLP*.

Kenji Sagae and Jun'ichi Tsujii. 2008. *Shift-Reduce Dependency DAG Parsing*. Proceedings of the COLING.

Mihai Surdeanu, Richard Johansson, Adam Meyers, Lluis Marquez, and Joakim Nivre. 2008. *The CoNLL-2008 Shared Task on Joint Parsing of Syntactic and Semantic Dependencies*. Proceedings of CoNLL.

Kristina Toutanova, Aria Haghighi, and Christopher D. Manning. 2008. *A Global Joint Model for Semantic Role Labeling*. Computational Linguistics 34(2).

Zhang, H. and Zhang, M. and Tan, C.L. and Li, H. 2009. *K-best combination of syntactic parsers*. Proceedings of Empirical Methods in Natural Language Processing.