In Proceedings of the 26th Conference on Artificial Intelligence (AAAI-12)

Modeling Textual Cohesion for Event Extraction

Ruihong Huang and Ellen Riloff
School of Computing
University of Utah
Salt Lake City, UT 84112
{huangrh,riloff}@cs.utah.edu

Abstract

Event extraction systems typically locate the role fillers for an event by analyzing sentences in isolation and identifying each role filler independently of the others. We argue that more accurate event extraction requires a view of the larger context to decide whether an entity is related to a relevant event. We propose a bottom-up approach to event extraction that initially identifies candidate role fillers independently and then uses that information as well as discourse properties to model textual cohesion. The novel component of the architecture is a sequentially structured sentence classifier that identifies event-related story contexts. The sentence classifier uses lexical associations and discourse relations across sentences, as well as domain-specific distributions of candidate role fillers within and across sentences. This approach yields state-of-the-art performance on the MUC-4 data set, achieving substantially higher precision than previous systems.

1 Introduction

The aim of event extraction systems is to identify noun phrases that represent *role fillers* for a specific type of event. Role fillers are the participants, objects, and properties associated with an event. For example, event extraction systems have been created to identify role fillers for management succession events (e.g., the names of people being hired or fired, and the companies involved), corporate acquisitions (e.g., purchased companies, and the purchasers), terrorism events (e.g., perpetrators, victims, and targets), and many others.

Most event extraction systems use patterns or classifiers to decide which noun phrases are role fillers based on the local context around them. However, each sentence in a story is usually processed independently, ignoring the rest of the document. Processing sentences in isolation can cause several problems. False hits can occur due to ambiguity and metaphor. For example, "Obama was attacked" may lead to Obama being extracted as the victim of a physical attack, even if the preceding sentences describe a presidential debate and the verb "attacked" is being used metaphorically. Many sentences also contain phrases that are role fillers only if the preceding context describes a relevant event. For example, people can be injured or killed in many ways (e.g., vehicle accidents, military engagements, natural disasters, and civilian crime). Someone who is injured or killed should be characterized as a victim of terrorism only if the discourse is describing a terrorist event.

Role fillers can also be overlooked because a sentence does not appear to be relevant when viewed in isolation. For example, consider the sentence "He used a gun". Without considering the surrounding story context, event extraction systems will either extract from these types of sentences, which improves recall but lowers precision (e.g., false hits will occur when non-perpetrators use weapons, such as police and soldiers), or they will ignore them, which means that some legitimate role fillers will be missed. In this paper, we argue that event extraction systems need to incorporate contextual influence across sentences in order to achieve better performance.

We propose a bottom-up approach for event extraction that aggressively identifies *candidate role fillers* based on local (intra-sentential) context, and then uses distributional properties of the candidate role fillers as well as other discourse features to model textual cohesion across sentences. Our event extraction architecture has two components: (1) a set of local role filler extractors, and (2) a sequential sentence classifier that identifies event-related story contexts. The novel component is the sentence classifier, which uses a structured learning algorithm, conditional random fields (CRFs), and features that capture lexical word associations and discourse relations across sentences, as well as distributional properties of the candidate role fillers within and across sentences. The sentence classifier sequentially reads a story and determines which sentences contain event information based on both the local and preceding contexts. The two modules are combined by extracting only the candidate role fillers that occur in sentences that represent event contexts, as determined by the sentence classifier.

2 Related Work

Most event extraction systems, both pattern-based [Appelt et al., 1993; Riloff, 1993; Soderland et al., 1995; Sudo, Sekine, and Grishman, 2003] and classifier-based [Chieu and Ng, 2002; Li, Bontcheva, and Cunningham, 2005], scan a text and search for individual role fillers based on local con-

texts around noun phrases. However, recent work has begun to explore the use of additional context to improve performance. [Maslennikov and Chua, 2007] used discourse trees and local syntactic dependencies *within* sentences in a pattern-based framework. [Gu and Cercone, 2006] created HMMs to first identify relevant sentences and then trained another set of HMMs to extract individual role fillers from the relevant sentences. [Patwardhan and Riloff, 2007] learned to recognize event sentences in a weakly-supervised learning paradigm and then extracted role fillers from the event sentences using patterns. GLACIER [Patwardhan and Riloff, 2009] jointly considered sentential evidence and local phrasal evidence in a probabilistic framework to extract role fillers.

Only a few event extraction models have gone beyond individual sentences to make extraction decisions. [Liao and Grishman, 2010] calculated document-level role filler statistics and used the co-occurrence information of different types of events and role fillers to train better extraction models. Ji and Grishman [2008] enforced event role consistency across different documents. TIER [Huang and Riloff, 2011] used a document genre classifier to recognize event narrative stories and then identified *event sentences* as well as *role-specific sentences* in the event narratives, but each sentence was classified and used independently.

Structured models have been applied in several areas of natural language processing, including event extraction. But previous event extraction research has used structured models to sequentially label noun phrases, not sentences (e.g., [Chieu and Ng, 2002; Finn and Kushmerick, 2004; Li, Bontcheva, and Cunningham, 2005; Yu, Guan, and Zhou, 2005]). Our research is the first to sequentially label sentences to identify domain-specific event contexts.

Our work is related to the document-level content models introduced by [Barzilay and Lee, 2004], which utilized a novel adaptation of the generative sequential model HMMs [Rabiner, 1989] to capture the topics that the texts address and the transitions between topics. The learned topic sequences improved two applications, information ordering and extractive summarization. Recently, [Sauper, Haghighi, and Barzilay, 2010] incorporates the latent content structure directly into two text analysis tasks, extractive summarization and sentiment analysis, in a joint learning framework. Our research also learns a structured sequential model for the sentences in a document. However, we are not aiming to model the content flow between all sentences. Our goal is to capture content transitions and discourse relations that can recognize event-related story contexts for a specific domain.

3 Improving Event Extraction by Modeling Textual Cohesion

Our event extraction model involves two processes that each focus on a different aspect of the problem. The left side of Figure 1 shows the two components and illustrates how they interact. The top component on the left is a set of traditional role filler detectors, one for each event role. This component identifies candidate role fillers based on the immediate context surrounding a noun phrase. These role fillers tend to be overly aggressive on their own, producing many correct extractions but also many false hits.

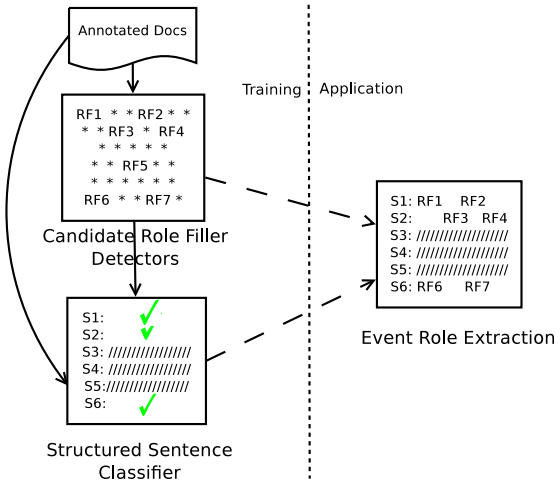

Figure 1: A Bottom-up Architecture for Event Extraction

The bottom component on the left side of Figure 1 is a structured sentence classifier that identifies event-related story contexts. This classifier determines whether a sentence is discussing a domain-relevant event based on two types of information. The structured learning algorithm explicitly models whether the previous sentence is an event context, which captures discourse continuity across sentences. We also provide the learner with features representing other textual cohesion properties, including lexical associations and discourse relations between adjacent sentences. In addition, the bottom-up design of the architecture provides information about candidate role fillers found by the local detectors. This domain-specific information is incorporated into features that represent the number, types, and distribution of the candidate role fillers both within and across sentences.

The two components provide different sources of evidence that are both considered when making final extraction decisions. The right side of Figure 1 illustrates how the two components are used. The event extraction system only produces a role filler if the noun phrase was hypothesized to be a candidate role filler based on local context *and* it appears in an event-related story context, as determined by the sequential sentence classifier. In the following sections, we describe each of these components in more detail.

4 Candidate Role Filler Detectors

The local role filler detectors are support vector machine (SVM) classifiers that label noun phrases with respect to an event role. We create a set of binary classifiers, one for each event role. If multiple classifiers assign positive scores to the same noun phrase, then the event role that receives the highest score is assigned.

Three types of features represent the local context surrounding a noun phrase (NP). Lexical features consist of four words to the left and and four words to the right of the targeted NP, as well as its head and premodifiers. Semantic

features include named entity tags produced by the Stanford named entity recognizer [Finkel, Grenager, and Manning, 2005] and semantic class labels assigned by the Sundance parser [Riloff and Phillips, 2004]. Finally, lexico-syntactic pattern features are produced by AutoSlog [Riloff, 1993; Riloff and Phillips, 2004], which automatically generates patterns for expressions in which the NP participates as a syntactic subject, direct object, or prepositional phrase.[1] The candidate role filler detectors only consider the local context surrounding a NP, so they tend to overgenerate role filler hypotheses when they see any context that could be relevant.

The classifiers are trained using the gold standard MUC-4 answer key templates. For each event role, the noun phrases matching a document's answer key strings for that event role are positive training instances. All other noun phrases in the document are negative instances. Since the number of negative instances is far greater than the number of positive instances, we randomly choose among the negative instances to create a 10:1 ratio of negative to positive instances.

Our candidate role filler detectors are identical to the local role filler extractors used by TIER [Huang and Riloff, 2011], which allows for direct comparisons between TIER and our new model. They are also very similar to the plausible role filler detectors used by GLACIER [Patwardhan and Riloff, 2009] (the other system we compare against in Section 6), except for small differences in the lexical features and the positive/negative training ratios.

5 Structured Sentence Classification to Identify Event Contexts

The sequential sentence classifier is responsible for determining which sentences are related to domain-relevant events. We utilize conditional random fields (CRFs) [Lafferty, McCallum, and Pereira, 2001] to carry out this sequential labeling task. A sequential CRF is a structured discriminative learning model that produces a sequence of labels using features derived from the input sequence. This component will sequentially read the sentences in a story and determine whether each sentence is discussing a relevant event based on direct evidence from both the current sentence and the previous sentence. All other sentences only affect the results indirectly through label transitions.

We used the CRF++ [2] toolkit to create our structured sentence classifier. CRF++ performs sequential labeling tasks and requires each unit in the input to have a fixed number of raw features. Since the length of sentences can vary, affecting the number of n-grams and other features accordingly, we expand the feature vector for each sentence with pseudo-tokens[3] as needed to ensure that every sentence has the same number of features. The toolkit was modified not to generate real features from the pseudo-tokens.

We provide the classifier with four types of features to represent individual sentences and textual cohesion properties linking adjacent sentences: basic features, lexical bridges, discourse bridges and role filler distributions. The following sections describe each of these feature sets.

5.1 Basic Features

As the basic representation of a sentence, we use unigram and bigram features. We create features for every unigram and bigram, without stemming or stopword lists. In addition, we found it beneficial to create five additional features representing the first five bigrams in the sentence. We define features for positions 1 through 5 of a sentence to represent the bigrams that begin in each of these positions. We hypothesize that these positional bigram features help to recognize expressions representing discourse cue phrases at the beginning of a sentence, as well as the main subject of a sentence.

5.2 Lexical Bridge Features

An important aspect of textual cohesion is lexical word associations across sentences. This idea has been explored in [Soricut and Marcu, 2006] to model the intuition that the use of certain words in a discourse unit (e.g., sentence) tends to trigger the use of other words in subsequent discourse units. In the context of event extraction, a pair of related event keywords may occur in consecutive sentences. For example, it is common to see "bombed" in one sentence and "killed" in the next sentence because bombing event descriptions are often followed by casualty reports. Similarly, we may see "attacked" and "arrested" in adjacent sentences because a mention of an attack is often followed by news of the arrest of suspected perpetrators.

To capture lexical associations between sentences, we create *lexical bridge* features that pair each verb in the current sentence ($Verb_i$) with each verb in the preceding sentence sentence ($Verb_{i-1}$):

$$< Verb_{i-1}, Verb_i >$$

To obtain better generalization, we stem the verbs before creating the bridge features using the Porter stemmer [Porter, 1980]. For example, a sentence that mentions a bombing followed by a sentence containing "killed" would generate the following lexical bridge feature:

$$< bomb, kill >$$

Event keywords could also appear as nouns, such as "assassination" and "death". Therefore, we also create lexical bridge features by pairing nouns from the current sentence and the preceding sentence:

$$< Noun_{i-1}, Noun_i >$$

For example, if we see the word "explosion" in the preceding sentence and the nouns "people" and "offices" in the current sentence, then two features will be created as follows:

$$< explosion, people >$$
$$< explosion, offices >$$

We also tried including associations between nouns and verbs in adjacent sentences (i.e. $< Verb_{i-1}, Noun_i >$ and $< Noun_{i-1}, Verb_i >$), but they did not improve performance. To focus on event recognition, the lexical bridges are

[1] These patterns are similar in spirit to the relations produced by dependency parsers.

[2] http://crfpp.sourceforge.net/#tips [crfpp.sourceforge.net]

[3] We define a special token for this purpose.

only created between sentences that each contains at least one candidate role filler.

5.3 Discourse Bridge Features

We also represent two types of discourse relations between consecutive sentences: discourse relations produced by a Penn Discourse Treebank (PDTB) trained discourse parser, and syntactic discourse focus relations. We hypothesized that these features could provide additional evidence for event label transitions between sentences by recognizing explicit discourse connectives or a shared discourse focus.

PDTB-style discourse relations [Prasad et al., 2008] are organized hierarchically in three levels based on different granularities. We use the discourse relation output produced by a PDTB-style discourse parser [Lin, Ng, and Kan, 2010]. Given a text, the discourse parser generates both explicit (triggered by cue phrases such as "if" or "because") and implicit level-2 PDTB discourse relations, such as cause, condition, instantiation, and contrast. A discourse relation may exist within a sentence or between two adjacent sentences in the same paragraph. We create features representing the intra-sentential discourse relations found in the current sentence, as well as the inter-sentential discourse relations connecting the current sentence with the previous one. Each discourse relation produced by the parser yields a feature for its discourse relation type:

$$< DiscRelType >$$

We also create features designed to (approximately) recognize shared discourse focus. We consider the noun phrases in three syntactic positions: subject, direct object, and the objects of "by" prepositional phrases (PP-by). Sentences in active voice constructions are typically focused on the entities in the subject and direct object positions as the central entities of the discourse. Sentences in passive voice constructions are usually focused on the entities in the subject and PP-by positions as the most central entities. We use the Stanford parser [Marneffe, MacCartney, and Manning, 2006] to identify these syntactic constituents.

The motivation for this type of feature is that sentences which have a shared discourse focus probably should be assigned the same event label (i.e., if one of the sentences is discussing a domain-relevant event, then the other probably is too). To capture the intuition behind this idea, consider the following two sentences:

(1) A customer in the store was shot by masked men.
(2) The two men used 9mm semi-automatic pistols.

Because the same entity (the men) appears in both the "by" PP of sentence (1) and the subject position of sentence (2), the classifier should recognize that the second sentence is connected to the first. Recognizing this connection may enable the extraction system to correctly identify the pistols as instruments used in the shooting event, even though sentence (2) does not explicitly mention the shooting.

We create a discourse focus feature for each shared noun phrase that occurs in two adjacent sentences in one of the designated syntactic positions. We consider any two noun phrases that have the same head word to match. We encode each feature as a triple consisting of the head word of the shared noun phrase ($NPHead$), the NP's position in the current sentence ($SynPos_i$), and the NP's position in the preceding sentence ($SynPos_{i-1}$):

$$< NPHead, SynPos_i, SynPos_{i-1} >$$

For example, sentences (1) and (2) would produce the following discourse focus feature:

$$< men, subject, PP\text{-}by >$$

5.4 Role Filler Distribution Features

The motivation for the bottom-up design of our event extraction architecture is that the sentence classifier can benefit from knowledge of probable role fillers hypothesized by the local detectors. Intuitively, the presence of multiple role fillers within a sentence or in the preceding sentence is a strong indication that a domain-relevant event is being discussed. The local detectors are not perfect, but they provide valuable clues about the number, types, and density of probable role fillers in a region of text.

First, we create features that capture information about the candidate role fillers within a single sentence. We create features for the event role type and the head noun of each candidate role filler in the sentence. We also encode two types of features that capture properties of the set of candidate role fillers. For each event role, we define a binary feature that indicates whether there are multiple candidate role fillers for that role. For example, if we see multiple victims in a sentence, this is more evidence than seeing a single victim. The second type of feature represents combinations of different event role types detected in the same sentence. We define 10 binary features that represent the presence of pairs of distinct event roles occurring in the same sentence.[4] For example, if we see both a perpetrator and a victim in a sentence, we may be more confident that the sentence is describing a crime.

We also create several types of features that represent role filler distributions across sentences. Intuitively, the presence of a particular type of role filler in one sentence may predict the presence of a role filler in the next sentence. For example, a gun is more likely to be an instrument used in a crime if the preceding sentences mention perpetrators and victims than if they only mention other weapons. To capture domain-specific distributional properties of the candidate role fillers, we create features for the role fillers found in adjacent sentences. We use both the head word of the noun phrase as well as the type of the event role. If the local detectors produce a candidate role filler of type $RFType_{i-1}$ with head $RFHead_{i-1}$ in the previous sentence, and a role filler of type $RFType_i$ with head $RFHead_i$ in the current sentence, then two features are generated:

$$< RFHead_{i-1}, RFType_i >$$

[4] Since there are 5 event roles, there are 10 pairs of distinct roles because the order of them doesn't matter.

$< RFHead_{i-1}, RFType_{i-1}, RFType_i >$

For example, assuming that three candidate role fillers have been detected for the example sentences in Section 5.3 (*Victim(customer)* and *Perpetrator(men)* from sentence (1) and *Weapon(pistols)* from sentence (2)), the following features will be created:

$< customer, Weapon >$
$< customer, Victim, Weapon >$
$< men, Weapon >$
$< men, Perpetrator, Weapon >$

We also create features to represent role fillers that occur in adjacent sentences and share a discourse relation. If two adjacent sentences share a discourse relation ($DiscRelType$), then we represent the types of role fillers found in those sentences, coupled with the discourse relation. For example, if two sentences are in a causal relation and the candidate role filler detectors found a candidate victim in the previous sentence and a candidate perpetrator in the current sentence, then the causal relation provides further evidence that the victim and perpetrator are likely correct. These types of features are represented as:

$< RFType_{i-1}, DiscRelType, RFType_i >$

For the example above, the feature would be:

$< Victim, cause, Perpetrator >$

Finally, verbs often provide valuable clues that a sentence is discussing an event, so the presence of a specific verb in the previous sentence may bolster a role filler hypothesis in the current sentence. We create an additional feature that links each verb in the previous sentence to each candidate role filler in the current sentence:

$< Verb_{i-1}, RFType_i >$

For example, a sentence containing a candidate victim preceded by a sentence containing the word "bombed" would produce the following feature:

$< bombed, Victim >$

When generating these features during training, the gold standard role fillers are not suitable because gold role fillers will not be available in new texts. A model trained with gold role fillers would probably not be effective when applied to new documents that have system-generated candidate role fillers. To obtain realistic values for the candidate role filler distributions, we used 5-fold cross-validation on the training data. To get the candidate role fillers for one fold, we trained the role filler detectors using the other four folds and then applied the detectors to the selected fold.

6 Evaluation

We evaluated our approach on a standard benchmark collection for event extraction research, the MUC-4 data set [MUC-4 Proceedings, 1992]. The MUC-4 corpus consists of 1700 documents with associated answer key templates. To be consistent with previously reported results on this data set, we use the 1300 DEV documents for training, 200 documents (TST1+TST2) as a tuning set, and 200 documents (TST3+TST4) as the test set.

PerpInd	PerpOrg	Target	Victim	Weapon
129	74	126	201	58

Table 1: # of Role Fillers in the MUC-4 Test Set

Following previous studies, we evaluate our system on the five MUC-4 "string-fill" event roles: *perpetrator individuals, perpetrator organizations, physical targets, victims* and *weapons*. These event roles (essentially) represent the agents, patients, and instruments associated with terrorism events. Table 1 shows the distribution of gold role fillers in the MUC-4 test set. The complete IE task involves template generation, which requires event segmentation because many documents discuss multiple events. Our work focuses on extracting individual role fillers and not template generation per se, so we follow the same evaluation paradigm of recent research and evaluate the accuracy of the role fillers directly (i.e., if the role filler has the correct label and appears in any event template for the document, then it is correct). We use head noun matching against the answer key strings (e.g., "armed guerrillas" is considered to match "guerrillas")[5]. Our results are reported as Precision/Recall/F(1)-score for each event role separately. We also show the macro average over all five event roles.

6.1 Experimental Results

Table 2 shows the evaluation results on the five event roles for the MUC-4 task, and the macro-average over all five roles. Each cell in the table shows the precision (P), recall (R), and F scores, written as P/R/F. The first row of numbers shows the results for the candidate role filler detectors when used by themselves. These local role filler extractors produce relatively high recall, but consistently low precision.

The next set of rows in Table 2 shows the effect of adding the structured sentence classifier to create the complete bottom-up event extraction model. We incrementally add each set of of textual cohesion features to assess the impact of each one separately. The Basic feature set row uses only the N-gram features. Even with just these simple features, incorprating the structured sentence classifier into the model yields a large improvement in precision (+25) but at the expense of substantial recall (-19).

The **+ Candidate RF features** row shows the impact of providing the candidate role filler information to the sentence classifier (see Section 5.4). Compared with the previous row, the role filler features produce an average recall gain of +3, with only a one point loss of precision. When looking at the event roles individually, we see that recall improves for all of the event roles except Targets.

The **+ Lexical Bridge features** row shows the impact of the lexical bridge features (Section 5.2). These features produced a two point gain in precision, yielding a one point gain

[5]Pronouns were discarded since we do not perform coreference resolution. Duplicate extractions with the same head noun were counted as one hit or one miss.

System	PerpInd	PerpOrg	Target	Victim	Weapon	Average
Local Extraction Only						
Candidate RF Detectors	25/67/36	26/78/39	34/83/49	32/72/45	30/75/43	30/75/42
with Structured Sentence Classifier						
Basic feature set	56/54/55	47/46/46	55/69/**61**	61/57/59	58/53/56	55/56/56
+ Candidate RF features	51/57/54	47/47/47	54/69/60	60/58/59	56/60/58	54/59/56
+ Lexical Bridge features	51/57/53	51/50/50	55/69/**61**	60/58/59	62/62/62	56/59/57
+ Discourse features	54/57/**56**	55/49/**51**	55/68/**61**	63/59/**61**	62/64/**63**	58/60/**59**
Previous Systems						
TIER (2011)	48/57/52	46/53/50	51/73/60	56/60/58	53/64/58	51/62/56
GLACIER (2009)	51/58/54	34/45/38	43/72/53	55/58/56	57/53/55	48/57/52

Table 2: Experimental results, reported as Precision/Recall/F-score.

in F-score. Two of the event roles (PerpOrg and Weapon) showed improvement in both precision and recall.

The **+ Discourse features** row shows the performance after adding the discourse bridge features (Section 5.3). The discourse features improve precision for three of the five event roles (PerpInd, PerpOrg, and Victim). Weapons also gain two points of recall. Overall, the discourse features yield a two point increase in the F score.

Together, all of the textual cohesion features yield a 3 point gain in precision and a 4 point gain in recall relative to the basic feature set (N-grams), achieving an F-score improvement of 3 points.

6.2 Comparison with Other Systems

We compare the performance of our event extraction model with two relatively recent event extraction systems that have been evaluated on the same MUC-4 data set: TIER [Huang and Riloff, 2011] and GLACIER [Patwardhan and Riloff, 2009]. TIER is a multi-layered architecture for event extraction. Documents pass through a pipeline where they are analyzed at different levels of granularity: document level, sentence level and phrase level. TIER is designed to identify secondary role filler contexts in the absence of event keywords by using a document genre classifier, a set of *role-specific sentence classifiers*, one per event role, in addition to an *event sentence classifier* (similar to classifiers used in other work [Patwardhan and Riloff, 2009; Gu and Cercone, 2006]). TIER has produced the best results reported to date on the MUC-4 event extraction data set [Huang and Riloff, 2011] for learning-based role filler extraction systems.

As a second baseline, we also compare our results with GLACIER [Patwardhan and Riloff, 2009]. GLACIER uses a unified probabilistic model for event extraction that jointly considers sentential evidence and phrasal evidence when extracting each role filler. It consists of an sentential event recognizer and a set of plausible role filler recognizers, one for each role. The final extraction decisions are based on the product of the normalized sentential and the phrasal probabilities.

The last two rows in Table 2 show the results for TIER and GLACIER, using the same evaluation criteria as our system. We compare their results with the performance of our complete event extraction system using all of the feature sets, which is shown in the **+ Discourse Features** row of Table 2. Compared with TIER, our model achieves 7 points higher precision, although with slightly lower recall (-2). Overall, our model yields a 3 point higher F score than TIER. If we look at the individual event roles, our model produces substantially higher precision across all five event roles. Recall is comparable for PerpInd, Victim, and Weapon, but is several points lower on the PerpOrg and Target roles. Compared with GLACIER, our model also shows significant gains in precision over all five event roles. Furthermore, the average recall is 3 points higher, with Weapons showing the largest benefit (+11 recall gain).

In summary, our bottom-up event extraction model yields substantially higher precision than previous event extraction systems on the MUC-4 data set, with similar levels of recall.

7 Conclusions

We have presented a bottom-up architecture for event extraction that demonstrates how textual cohesion properties can be integrated into an event extraction model to improve its accuracy. Our event extraction system has two components: (1) local role filler detectors that identify candidate role fillers, and (2) a structured sentence classifier that identifies event-related story contexts. The main contribution of our work is the integration of a sequential sentence classifier that utilizes features representing several types of properties associated with textual cohesion, including lexical associations and discourse relations across sentences. In addition, the bottom-up design of the architecture allows the sentence classifier to consider distributional properties of the domain-specific candidate role fillers, both within and across sentences. This model yields state-of-the-art performance on the MUC-4 data set for learning-based systems, achieving substantially higher precision than previous models. In future work, we hope to explore additional textual cohesion properties and discourse issues associated with event descriptions to further improve event extraction performance.

8 Acknowledgments

We gratefully acknowledge the support of the National Science Foundation under grant IIS-1018314 and the Defense Advanced Research Projects Agency (DARPA) Machine Reading Program under Air Force Research Laboratory (AFRL) prime contract no. FA8750-09-C-0172. Any opinions, findings, and conclusions or recommendations ex-

pressed in this material are those of the authors and do not necessarily reflect the view of the DARPA, AFRL, or the U.S. government.

References

Appelt, D.; Hobbs, J.; Bear, J.; Israel, D.; and Tyson, M. 1993. FASTUS: a finite-state processor for information extraction from real-world text. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence*.

Barzilay, R., and Lee, L. 2004. Catching the drift: Probabilistic content models, with applications to generation and summarization. In *Proceedings of the Annual Meeting of the North American Chapter of the Association for Computational Linguistics (HLT/NAACL 2004)*.

Chieu, H., and Ng, H. 2002. A Maximum Entropy Approach to Information Extraction from Semi-Structured and Free Text. In *Proceedings of the 18th National Conference on Artificial Intelligence*.

Finkel, J.; Grenager, T.; and Manning, C. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, 363–370.

Finn, A., and Kushmerick, N. 2004. Multi-level Boundary Classification for Information Extraction. In *Proceedings of the 15th European Conference on Machine Learning*, 111–122.

Gu, Z., and Cercone, N. 2006. Segment-Based Hidden Markov Models for Information Extraction. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, 481–488.

Huang, R., and Riloff, E. 2011. Peeling Back the Layers: Detecting Event Role Fillers in Secondary Contexts. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (ACL-11)*.

Ji, H., and Grishman, R. 2008. Refining Event Extraction through Cross-Document Inference. In *Proceedings of ACL-08: HLT*, 254–262.

Lafferty, J.; McCallum, A.; and Pereira, F. 2001. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *Proceedings of the Eighteenth International Conference on Machine Learning*.

Li, Y.; Bontcheva, K.; and Cunningham, H. 2005. Using Uneven Margins SVM and Perceptron for Information Extraction. In *Proceedings of Ninth Conference on Computational Natural Language Learning*, 72–79.

Liao, S., and Grishman, R. 2010. Using Document Level Cross-Event Inference to Improve Event Extraction. In *Proceedings of the 48st Annual Meeting on Association for Computational Linguistics (ACL-10)*.

Lin, Z.; Ng, H. T.; and Kan, M.-Y. 2010. A PDTB-Styled End-to-End Discourse Parser. In *Technical Report TRB8/10, National University of Singapore, August*.

Marneffe, M.-C. d.; MacCartney, B.; and Manning, C. D. 2006. Generating Typed Dependency Parses from Phrase Structure Parses. In *Proceedings of the Fifth Conference on Language Resources and Evaluation (LREC-2006)*.

Maslennikov, M., and Chua, T. 2007. A Multi-Resolution Framework for Information Extraction from Free Text. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*.

MUC-4 Proceedings. 1992. *Proceedings of the Fourth Message Understanding Conference (MUC-4)*. Morgan Kaufmann.

Patwardhan, S., and Riloff, E. 2007. Effective Information Extraction with Semantic Affinity Patterns and Relevant Regions. In *Proceedings of 2007 the Conference on Empirical Methods in Natural Language Processing (EMNLP-2007)*.

Patwardhan, S., and Riloff, E. 2009. A Unified Model of Phrasal and Sentential Evidence for Information Extraction. In *Proceedings of 2009 the Conference on Empirical Methods in Natural Language Processing (EMNLP-2009)*.

Porter, M. 1980. An algorithm for suffix stripping. *Program* 14(3):130–137.

Prasad, R.; Dinesh, N.; A., L.; Miltsakaki, E.; Robaldo, L.; A., J.; and Webber, B. 2008. The Penn Discourse Treebank 2.0. In *Proceedings of the Sixth Conference on Language Resources and Evaluation (LREC-2008)*.

Rabiner, L. 1989. A tutorial on hidden Markov models and selected applications in speech recognition. *Proceedings of the IEEE* 77(2):257–286.

Riloff, E., and Phillips, W. 2004. An Introduction to the Sundance and AutoSlog Systems. Technical Report UUCS-04-015, School of Computing, University of Utah.

Riloff, E. 1993. Automatically Constructing a Dictionary for Information Extraction Tasks. In *Proceedings of the 11th National Conference on Artificial Intelligence*.

Sauper, C.; Haghighi, A.; and Barzilay, R. 2010. Incorporating Content Structure into Text Analysis Applications. In *Proceedings of 2010 the Conference on Empirical Methods in Natural Language Processing (EMNLP-2010)*.

Soderland, S.; Fisher, D.; Aseltine, J.; and Lehnert, W. 1995. CRYSTAL: Inducing a conceptual dictionary. In *Proc. of the Fourteenth International Joint Conference on Artificial Intelligence*, 1314–1319.

Soricut, R., and Marcu, D. 2006. Discourse Generation Using Utility-Trained Coherence Models. In *Proceedings of Joint Conference of the International Committee on Computational Linguistics and the Association for Computational Linguistics (COLING/ACL-06)*, 803–810.

Sudo, K.; Sekine, S.; and Grishman, R. 2003. An Improved Extraction Pattern Representation Model for Automatic IE Pattern Acquisition. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics (ACL-03)*.

Yu, K.; Guan, G.; and Zhou, M. 2005. Resumé Information Extraction with Cascaded Hybrid Model. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, 499–506.