Event Argument Linking and Event Nugget Detection Task: IHMC DISCERN System Report

Archna Bhatia and **Adam Dalton** and **Bonnie Dorr** and **Greg Dubbin**
Kristy Hollingshead and **Suriya Kandaswamy** and **Ian Perera**
Institute for Human and Machine Cognition, Ocala, FL
{abhatia, adalton, bdorr, gdubbin, kseitz, iperera}@ihmc.us, suriya.kandaswamy@gmail.com

Abstract

We present the design and implementation of three variants of an event-detection system called DISCERN, one that used manually created rules (DISCERN-R), one that used machine-learned rules (DISCERN-ML), and one that combined manual and machine-learned rules (DISCERN-C). All three used linguistic resources (VerbNet, CatVar, Semantic Role Labeling, NER, POS tagging, dependency parsing, and coreference resolution) and were applied to two tasks in the NIST TAC KBP 2015 Event Track: Event Nugget Detection (EN) and Event Argument Extraction and Linking (EAL). Two contributions of this work are: (a) a web interface that improved efficiency during system development by enabling quick changes to linguistic rules and examination of their effect on precision and recall at runtime; (b) an approach to collapsing support-verb and event nominals that improved recall of event argument detection enough to surpass the median in the NIST TAC KBP evaluation. Future refinements to the combination of linguistic and machine learning approaches may result in improvements due to the complementarity of these approaches: linguistically informed rules can improve precision and machine-learned systems can improve recall.

1 Introduction

With increasingly large volumes of textual data available, most of which is unstructured, it has become necessary to build and apply automatic systems for extraction of information for the analysis of data that is too large for fully manual processing. The Text Analysis Conference (TAC) at NIST attempts to encourage research and development of such systems "by providing a large test collection, common evaluation procedures, and a forum for organizations to share their results."

The 2015 NIST TAC Event track focuses on detection of information about events from unstructured text. The extracted information could be used to populate a knowledge base, among other uses. Two NIST TAC KBP tasks are described in this paper, one for Event Nugget Detection (EN) and one for Event Argument Extraction and Linking (EAL). Event Nugget Detection refers to the identification of explicit events mentions, sometimes called "nuggets" or "triggers", in English texts.

The relevant event types/subtypes are taken from the Rich ERE annotation guidelines. The examples in 1.1 and 1.2 (Mitamura et al., 2015) express the same event type, *Conflict.Attack*; however, as the examples show, an event mention may involve a single word (1.1) or a multi-word expression (1.2).

Example 1.1
*The **attack** by insurgents occurred on Saturday.*

Example 1.2
Kennedy was ***shot dead*** *by Oswald.*

The EN task additionally involves identifying a realis state (ACTUAL, GENERIC, OTHER) for each event mention.

EAL involves extracting information about entities and possibly times and/or locations of an event, and the role these entities play in the event. For example, in 1.2, the event type is *Conflict.Attack*, the entity *Kennedy* plays the role of a *Target* and entity *Oswald* plays the role of an *Attacker* in the event. The EAL task also involves linking the arguments that belong to the same event, as well as identifying each event's realis state.

We present three variants of our event detection system for the two tasks described above, as applied to a development data set from NIST TAC 2014. We also discuss the application of these variants to the NIST TAC 2015 evaluation data. The structure of the paper is as follows: Section 2 presents related work, section 3 describes our process of event detection and the three system variants we built in detail, and section 4 presents a discussion of the results for our three variants on the development as well as the evaluation data. Finally, section 5 presents a summary of our findings and a brief discussion of potential work.

2 Related Work

Previous work on event detection has focused primarily on formal genre, such as news articles. For example, Roberts and Harabagiu (2011) focused on extraction and representation of the type of event and its participants, using topic modeling to detect 'event scenarios' in formal texts. However, in current times, a huge amount of data is becoming available in other genres as well. Social media, discussion forums, and various types of outlets where individuals independently publish (with corresponding comment sections) can provide the most up-to-date information about current events. The NIST TAC tasks involve detection of events in both formal (news genre) and informal (social media) texts. Thus, our work focuses on both these genres of texts.

In terms of the approaches used, many event extraction systems use syntax-based approaches to event detection. For example, Riloff (1993) used syntactic patterns, while Grishman et al. (2005) and McClosky et al. (2011) used a combination of syntactic patterns and statistical classifiers. Dependency parsing has been used quite widely for relation and event extraction, e.g., Nakashole et al. (2012), Alfonseca et al. (2013), Lewis and Steedman (2013), and Rusu et al. (2014).

While syntactic patterns can help us to detect events and their arguments to some extent, they are not always sufficient. Sometimes an accurate characterization of an event requires semantic context. Exner and Nugues (2011) used semantic parsing (semantic role labeling; SRL) to extract events from texts automatically, but their system misidentified agents quite frequently. Such errors could be reduced with the help of named entity recognition (NER) and syntactic parsing.

DISCERN differs from approaches above in that it makes use of both syntactic and semantic information, as well as manual and machine-learning techniques, for the detection of event triggers and their arguments. Prior work on event detection (Dorr et al., 2014), combined with a semantic approach (Ferguson et al., 1996), enables a more robust event detection capability, starting with syntactic dependency relations upon which semantic analysis is applied. A semantic classification of verbs and arguments that takes into account *categorial variants* of verbs widens the potential for event extraction (see VerbNet (Levin, 1993; Schuler, 2005), the NIST ontology (NIST, 2015), and CatVar (Dorr et al., 2003)).

Chen et al. (2014) designed ClearEvent, which is similar to our approach in that it combined syntax, deep semantic analysis as well as machine learning. The use of CatVar in DISCERN shows promising results for a wider coverage of event triggers beyond what would be available in the ClearEvent system.

A system developed by Mannem et al. (2014), that also employed machine learning techniques (joint extraction using beam search for decoding and an early update perceptron for training the model) and some syntactic features, yielded results that were accurate, but with limited recall. An advantage of this approach is that it captured interdependencies between event triggers and their arguments. Although not yet explored, it is expected that the combination of this approach with the semantic classification of verbs and categorial variants in DISCERN will be an important step in addressing the precision/recall tradeoff.

Sammons et al. (2014) used bag-of-words and part-of-speech (POS) as features for determination of realis. Additional information beyond these features was used in DISCERN; for example, negative lemmas such as *not* and the notion of "collapsing" for support-verb triggers such as the word *conduct* in *conduct an attack*. These additions were critical for the correct assignment of realis, e.g., *did not conduct an attack* (where the realis is OTHER) versus *attacked* (where the realis is ACTUAL).

3 The Process of Event Detection

We developed three variants of DISCERN (Discovering and Characterizing Emerging Events), a system designed to detect a set of events, such as those specified in the NIST (2014) and NIST (2015) Event tasks.[1] The description of the process of event detection used by the three variants of the DISCERN system is provided below.

For both EN and EAL, DISCERN was applied after a set of rules for event trigger and argument detection were manually crafted or learned:

1. Preprocessing the data: Dependency and constituency parses were generated for each sentence and subsequently annotated with linguistic features, such as VerbNet, CatVar, SRL, NER, and coreference. This step was common across the three variants of DISCERN.

2. Implementation of DISCERN: Application of predetermined rules to identify the event triggers and their arguments, as well as assigning realis values to them. The three variants of DISCERN varied with respect to the rules they applied (each variant created its own set of rules using different approaches, as described in section 3.3).

The two stages of the event detection process are described in sections 3.1 and 3.2, and in section 3.3, the differences among the three DISCERN variants are discussed with respect to rule creation and learning. Finally, Section 3.4 describes a web interface created for rapid revision of the manually created linguistic rules. This interface was designed to provide quantifiable guidance for determining where effort should be expended in manual rule construction.

3.1 Preprocessing the data

Documents were first stripped of XML, tokenized, and then split into sentences using the Stanford CoreNLP Natural Language Processing Toolkit (Manning et al., 2014). Next, POS

[1]The Events evaluated in the TAC 2015 Evaluation are divided into 9 types, each with a number of subtypes: Business (Start, End, Declare Bankruptcy, Merge), Conflict (Attack, Demonstrate), Contact (Meet, Correspondence, Broadcast, Contact), Manufacture (Artifact), Life (Be Born, Marry, Divorce, Injure, Die), Transaction (Transfer Ownership, Money, Transaction), Personnel (Start-Position, End-ition, Nominate, Elect), Movement (Transport-Person, Transport-Artifact), and Justice (Arrest-Jail, Release-Parole, Trial-Hearing, Sentence, Fine, Charge-Indict, Sue, Extradite, Acquit, Convict, Appeal, Execute, Pardon). An area of future work is to expand from these more general categories to more refined ones, and also to add new domains.

tagging, lemmatization, named entity recognition, and coreference annotations were applied using the default CoreNLP English probabilistic context-free grammars (PCFG) parse model and 3class, 7class, and MISCclass (in that order) NER models. The Stanford annotations were then serialized to XML, and further broken out so that the annotations for each sentence were written to a separate file to improve parallelizability during subsequent annotation stages, as described below.

Each annotated sentence representation from a document was passed through a pipeline wherein additional lexical and semantic resources were automatically added to improve DISCERN's capabilities in recognizing patterns. The pipeline included the following stages:

1. Each lemma was used to search CatVar. Any variations found were added as Word-POS pairs.

2. Each token was POS-tagged as a verb was augmented with the corresponding Verb Class from VerbNet.

3. Each sentence was reprocessed by the SENNA semantic role labeler and each token was labeled if it occurred within a semantic role.

4. Finally, the sentence was restructured if it contained a support-verb.

The primary benefit of CatVar was its ability to determine whether a categorial variation of a known verb was encountered in the input. This extends our ability to identify possible triggers beyond only verbal lemmas for an event to include categorial variants as well. The focus here was on detecting nominalized verbs such as "the *merger* of the two companies" or "the *destruction* of the city" beyond the verbal lemmas *merge* or *destroy*, respectively. If a token was found to have a verb variation in addition to its given POS, the serialized annotation was extended to include it.

The final step of the pipeline used all previous information for the application of a support-verb and event nominal merger rule that "collapsed" the structure of a phrasal unit containing a support-verb head coupled with a nominal trigger. For example, while the dependency parser might pick "declare" as the root of the tree in the phrase "declare bankruptcy", the desired event is a *Business.Declare-Bankruptcy* event, not a declare event. Detecting support-verbs was also used in determining the realis values for events with nominal triggers, as described in Section 3.2.

3.2 Implementation of DISCERN

Each of the DISCERN variants was applied to the preprocessed data in 4 steps. First, DISCERN located potential triggers for each event subtype. Second, each trigger was assigned a realis value according to linguistic rules. Next, the system located role filling arguments for a trigger among its dependents. Finally, DISCERN resolved arguments to canonical argument strings (CAS) according to annotated coreference and named-entity data. For the EN task, the process stopped after the realis assignment took place.

Each DISCERN variant employed a different strategy for locating potential triggers, as described in section 3.3. DISCERN's operation relied on lemma and CatVar annotations primarily to find potential trigger words. CatVar annotations enabled the generalization of results to unspecified, but semantically related, variants of the verbs denoting relevant events.

Once a trigger was identified, realis was assigned according to a series of linguistically-motivated rules. The realis values (ACTUAL, GENERIC, or OTHER) were based on tense and aspect encoded in the POS tags, negative lemmas, etc. For the cases where the triggers involved support-verbs and event nominals, realis was assigned after the support-verb trigger collapsing had taken place, so the anchor for the realis value was the merged result and had the POS of the original support-verb. The rules are described in pseudocode in Code 1.

The next step was to determine an event's arguments from its trigger's dependents. As with the first step, the method for detecting arguments was dependent on the DISCERN variant. However, each variant generally relied on some combination of dependency type, semantic role label, named entity type, and POS annotations when extracting event arguments.

The final canonical argument string (CAS) represented the first mention of entity arguments. The DISCERN variants resolved CAS according to the Stanford CoreNLP coreference annotations where available. For named entities, entity type information was used to find the full named entity string, e.g., "States" becomes "The United States". Lastly, time arguments were resolved according to TIMEX annotations.

3.3 Three DISCERN Variants

The three variants of DISCERN, namely DISCERN-R, DISCERN-ML and DISCERN-C, differed with respect to how the rules were created or learned for the detection of event types and corresponding arguments. The DISCERN-R system variant was based on hand-designed linguistic rules that were later applied automatically. The DISCERN-ML variant applied rules that were learned through a supervised machine learning algorithm. The DISCERN-C variant combined these two approaches to deriving the rules. Below we discuss each of these approaches in more detail.

3.3.1 DISCERN-R: Based on Manually Created Linguistic Rules

DISCERN-R used output from the Stanford Dependency Parser to extract events through previously hand-crafted linguistically motivated rules according to the NIST event descriptions. Triggers for event types were identified in the rules based on lemma matching against various lexical resources, such as dictionaries, thesaurus, VerbNet, CatVar, and OntoNotes. Once a trigger was identified, each of its dependents was considered as a possible argument for the event-type. Semantic rules for roles such as Agent, Victim, Prosecutor, etc. were used to determine which dependents filled them. For example, a *Conflict.Attack* event requires an Agent role to be filled by an entity, hence based on a rule for the Agent role for this event type, that entity was extracted from the dependency relations nsubj (subject for a verb) or poss (possessive, as in "The United State's invasion of Iraq").

3.3.2 DISCERN-ML: Based on Machine Learned Rules

DISCERN-ML employed a supervised ML algorithm, a variation of the iterative dichotomiser 3 (ID3) algorithm (Quinlan, 1986), to induce a random forest of decision tree rules for the detection of event triggers and their arguments. It used 10 decision trees that were trained on 10 random partitions of the training data (Rich ERE Training, 2015 EAL Training and 2014 EA Assessments). For each tree, the algorithm selected a random sample (with replacement) of training documents where the sample size was 66% the size of the entire training set.

The ID3 algorithm created a decision tree by greedily partitioning training data based on the attribute which maximizes *information gain* of the next partition. The information gain $IG(A, S)$ for attribute A on data subset S was defined as

$$IG(A, S) = H(S) - \sum_{t \in T} p(t)H(t).$$

where T was the set of partitions of S for each value of attribute A, $p(t)$ was the proportion of data in partition t, and $H(S)$ was the entropy of

Code 1: Realis rules depend on a combination of tense, aspect, POS, and negation.

```
if anchor is non-verbal:
    check dependents and governor for copula (type 'cop')
    if copula found:
        continue with anchor := copula
    else:
        realis := ACTUAL
        End
if no dependents of anchor have tag 'MD'
    if the anchor is past tense (tag 'VBD' or 'VBN')
    OR an auxiliary (type 'aux') dependent is past tense
    OR the anchor is 'VBG' with any auxiliary dependents:
        if there is a negative dependent (type 'neg'):
            realis := OTHER
            End
        else:
            realis := ACTUAL
            End
    else:
        realis := GENERIC
        End
else:
    realis := OTHER
```

the set S. The entropy of a set S was

$$H(S) = -\sum_{c \in C} p(c) \log_2 p(c)$$

where C is the set of target classes in the training data.

During implementation, each tree voted on the triggers. If the majority of decision trees determined that a token was a trigger for a given event, it was assigned a realis and a set of arguments as described in section 3.2. Each trigger detection decision tree had an associated argument detection decision tree. These candidate arguments were voted on similarly to the triggers, with a majority of votes determining an argument. An argument detection tree only voted on an argument if its associated trigger detection tree voted for the trigger to which that argument would be assigned.

3.3.3 DISCERN-C: Combining the Manually Created Linguistic Rules and the Machine Learned Rules

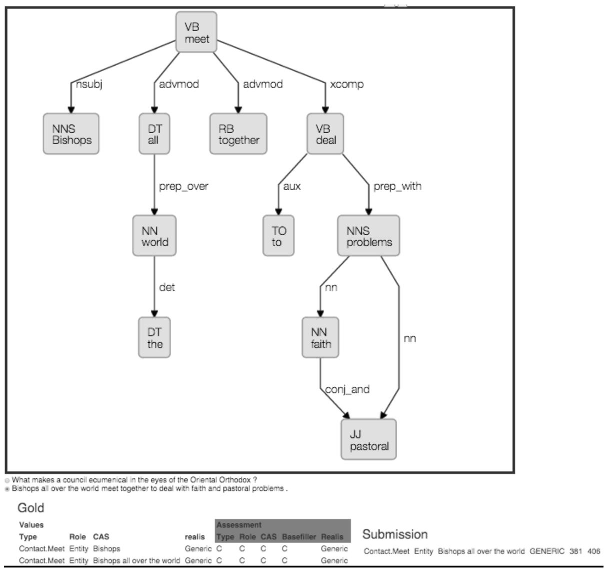
One would anticipate that the DISCERN-ML variant might discover complementary rules to the ones encoded in DISCERN-R. Therefore, a third variant, DISCERN-C, combined both of the other variants. This variant was a preliminary exploration of further attempts to merge linguistic and machine-learned knowledge.

DISCERN-C combined the two sets of rules from the other two runs (with DISCERN-R rules and with the random forest of decision trees from DISCERN-ML). The fact that both DISCERN-R and DISCERN-ML followed roughly the same path for implementation, as described in section 3.2, allowed DISCERN-C to compare the output of both without reconfiguring either variant. In fact, DISCERN-C implemented the same voting approach as DISCERN-ML, but the rules from DISCERN-R counted for 5 votes while each decision tree rule from DISCERN-ML counted for 1.

3.4 The Web Interface for Developing the DISCERN-R **variant**

A web-based front-end was added to DISCERN in 2015 to improve decision making of linguistic informants in deciding where to focus efforts when designing new rules. The front-end includes interfaces such as (i) an overview of the comparative performance of different rule sets in terms of precision, recall, and F_1 score, (ii) a detailed breakdown of error types (true positive vs false negative) and location of errors (for example, if the base filler, event type, event argument role or realis was inaccurate) per run, and (iii) an in-depth view of the annotations and parse structure of each sentence to provide the necessary information for constructing new rules. Finally, DISCERN now includes an interface where informants can interactively design rules and apply them to the development data set in order to determine whether a new rule effectively captures the syntactic and semantic phenomena it targeted while improving the overall performance of the system.

The interface proved to be particularly useful when the SENNA module was added to the DISCERN pipeline and new rules needed to be created to incorporate semantic role labeling. Figure 1 shows a portion of the document view that included a *Contact.Meet* event that we had not been able to capture before the SENNA module was added. Using the insights gained through improved visualizations and structured layout of the document content, we were able to improve the overall F_1 score on the development set from 8.8 to 9.4.



○ What makes a council ecumenical in the eyes of the Oriental Orthodox ?
● Bishops all over the world meet together to deal with faith and pastoral problems .

Gold

Values				Assessment				
Type	Role	CAS	realis	Type	Role	CAS	Basefiller	Realis
Contact.Meet	Entity	Bishops	Generic	C	C	C	C	Generic
Contact.Meet	Entity	Bishops all over the world	Generic	C	C	C	C	Generic

Submission

Contact.Meet Entity Bishops all over the world GENERIC 381 406

Figure 1: A sample of the DISCERN sentence detail interface

4 Evaluation and Discussion

This section describes the results of development experiments as well as the evaluation results for the TAC 2015 Event Argument and Event Nugget detection tasks. Section 4.1 describes the various experiments run during the development of the DISCERN system, including feature ablation studies. Section 4.2 analyzes the results of the TAC KBP 2015 evaluation, exploring what worked and what might be improved.

4.1 Development Experiments

As described in section 3, the implementation of the three variants of DISCERN can be broadly divided into two stages: preprocessing/annotation and detection. The development process consisted of iteratively experimenting with annotation features and incorporating them into rules for DISCERN-R or learning for DISCERN-ML.

DISCERN includes five features: support-verb collapsing, semantic role labels, named entity recognition, CatVar, and dependency types. Table 1 presents the results of an ablation experiment to determine the benefits of each of

support-verbs	+	-	-	-	-	-	-
SRL	+	+	-	-	-	+	+
NER	+	+	+	-	-	+	-
CatVar	+	+	+	+	-	-	+
dependencies	+	+	+	+	+	+	-
Precision	10.88%	10.89%	11.99%	11.00%	11.71%	**12.08%**	10.93%
Recall	**5.49%**	5.39%	3.76%	3.76%	3.66%	3.66%	4.99%
F-Score	**7.30%**	7.21%	5.73%	5.61%	5.58%	5.62%	6.85%

Table 1: Ablation results showing the effects of five features on precision, recall, and F-score with rules from DISCERN-R.

these features on the joint detection of events and their arguments. The experiment used the rules from DISCERN-R. DISCERN uses either syntactic dependencies or SRL relations to search for potential arguments, so the last experiment explored the use of SRL with no dependency information or features annotated on the dependencies (CatVar is still used to find triggers).

As the features were removed, the recall and/or precision of the event detection system decreased, indicating the contribution each of these features makes in DISCERN-R. Performance dropped when the two rule-based features, CatVar and support-verb rules, are removed. Losing the support-verb merger rules resulted in a drop in recall (as well as the F-score). Additionally, without the categorial variants from CatVar, DISCERN-R missed syntactic variations with same eventuality (e.g., verb *destroy* and noun *destruction*). Thus, the recall dropped, and while precision went up without CatVar, the overall F-score was still lowered.

As the semantic role labeller (SRL) was removed, there was a big drop in recall (an increase in false negatives). This is presumably due to the fact that SRL helps the system identify arguments corresponding to participant roles in an event by providing the semantic links between a verb and its arguments when there is no direct syntactic link. DISCERN-R relied on named entity recognition (NER) to eliminate inappropriate argument roles (e.g., a PERSON entity cannot fill the role of a Crime argument) – notice the drop in precision from 11.99% to 11% when NER is removed.

The final column of Table 1 shows the effect of semantic role information on determining arguments and their roles. More than one in five of the arguments found by the system with only SRL would not be detected without it. This reinforces the hypothesis that semantic roles as a feature provided the biggest boost to event argument recall. However, the low precision indicates that many arguments that were detected using the SRL feature were actually incorrect—although not enough to outweigh the benefits to F-score. Improved precision in the annotated semantic roles would greatly improve the precision of DISCERN.

The linguistically-motivated rules for support-verbs were synergistic with those for categorial variants. The support-verb collapser captured arguments of nominal triggers that CatVar found. The combination of the two contributed almost as much to recall as SRL (an absolute gain of 1.83 versus 1.63) with a similar loss of precision. DISCERN captured many support-verb instances with a list of only 42 different verbs. CatVar is not exhaustive and could be expanded using Verb-Noun lists and dictionary resources. Improving either of these two resources would also improve DISCERN's performance.

The benefits of the data-heavy features, by

Run	Precision	Recall	F-Score
DISCERN-R	**23.3%**	26.7%	**24.9%**
DISCERN-ML	5.9%	**30.7%**	9.9%
DISCERN-C	6.1%	25.3%	9.9%

Table 2: Precision, Recall, and F-score results from final event nugget systems on development data.

Run	Plain	Type	Realis	Both
R	**43.4%**	**38.9%**	**25.5%**	**23.3%**
ML	12.5%	10.4%	7.4%	5.9%
C	13.5%	11.1%	7.7%	6.1%

Table 3: Precision scores for each DISCERN variant on event nugget development data by error.

themselves, were less clear-cut. While SRL provided the largest positive effect on F-score of any single feature, it also had the most negative effect on precision (as seen by the drop from 11.99% to 10.89% when the feature is added). For every true positive found with SRL that was not found without it, about 9 false positives were wrongly detected. DISCERN-R used the SENNA semantic role labeler (Collobert et al., 2011) for its SRL system. SENNA's SRL system was trained on WSJ sections 02-21 (Collobert et al., 2011) annotated for the CoNLL 2005 SRL shared task. Its training data was comprised of approximately 85,000 sentences with 250,000 arguments (Carreras and Màrquez, 2005). On the other hand, NER had a positive effect on precision (increasing precision from 11% to 11.99% when included) without reducing recall. However, Stanford NER system was trained on Reuters Corpus (Collobert et al., 2011), which contained approximately 14,987 sentences and 203,621 tokens (Tjong Kim Sang and De Meulder, 2003), considerably less than the data needed for SRL.

While the ablation studies showed the benefits of each feature on argument detection, they focused only on the DISCERN-R variant. Table 2 shows the performance of each variant of DISCERN on the event nugget detection development set from the TAC 2014 evaluation data.

Table 2 shows that while each DISCERN variant had similar recall, the DISCERN-ML and DISCERN-C had much lower precision than DISCERN-R. In fact, DISCERN-C had almost the same precision as DISCERN-ML, indicating that many of the incorrect nuggets detected by the DISCERN-ML variant had a strong enough majority to override the extra votes of DISCERN-R.

Table 3 provides a more fine-grained insight into the precision of the DISCERN variants, breaking it down by error type. Approximately 1 in 8 nuggets identified by DISCERN-ML are true positives, half of which are assigned the correct type and realis. On the other hand, 43.4% of the nuggets identified by DISCERN-R were correct, with 23.3% overall precision (almost four times DISCERN-ML). This suggests that the DISCERN-ML decision trees did not generalize well, which could be addressed with better pruning algorithms. Furthermore, a considerable number of errors were a result of incorrect realis assignment – nearly half of correctly identified nuggets were labeled with incorrect realis across all variants.

Finally, Table 4 shows the results of the final DISCERN event argument system for each variant. Interestingly, while DISCERN-R had worse recall than DISCERN-ML on nugget detection, its recall was better for arguments. This relative improvement in recall for DISCERN-R comes at a cost: while DISCERN-ML event argument precision was almost the same as its event nugget precision, the DISCERN-R precision dropped considerably. This suggests that when DISCERN-ML found a correct trigger, most of the arguments it found for that trigger were also correct. This may be because the triggers it does identify have simple argument constructions. Alternatively, the argument decision trees might have relatively high precision.

Run	Precision	Recall	F-Score
DISCERN-R	**11.6%**	**8.4%**	**9.7%**
DISCERN-ML	5.8%	5.9%	5.9%
DISCERN-C	4.8%	3.3%	3.9%

Table 4: Precision, Recall, and F-score results from final DISCERN event argument variants on development data.

4.2 Evaluation Results

The evaluation results of DISCERN do not completely match those from the development experiments: for example, DISCERN-C outperforms DISCERN-ML in both tasks on the evaluation data. This section presents and analyzes the evaluation results for each task.

4.2.1 Event Nuggets

Table 5 presents the DISCERN results from the Event Nugget evaluation. Unlike the development results, DISCERN-C showed the highest recall for each possible error type. However, the low precision of DISCERN-ML was still evident and still severely affected the F-score of both DISCERN-ML and DISCERN-C. Future improvements to the machine learning system would benefit from an emphasis on improving precision. Pruning the learned decision trees to create more general leaf nodes would be one possible approach. Alternatively, a new combined system could be created that uses the high precision rules from DISCERN-R for trigger detection and only learns argument detection trees.

As in the development results, precision, recall, and F-score dropped considerably when assigning realis. Few nuggets were assigned the correct realis type but wrong event type relative to the converse. The realis rules were designed to capture a variety of simple cases; more comprehensive rules might improve overall event nugget performance.

Run	Precision	Recall	F-Score
DISCERN-R	**12.8%**	14.1%	**13.5%**
DISCERN-ML	7.4%	9.2%	8.2%
DISCERN-C	8.2%	**15.0%**	10.6%
median	30.7%	11.7%	16.9%
LDC	73.6%	39.4%	51.4%

Table 6: Precision, Recall, and F-score results for final event argument system variants in the TAC KBP 2015 Event Argument Evaluation.

4.2.2 Event Arguments

Table 6 presents the event argument evaluation results of DISCERN. As with the event nugget evaluation results and contrary to the development experiments, DISCERN-C outperformed DISCERN-ML. Consistent with the development tests, the low precision of DISCERN-ML brought down the precision of DISCERN-C.

Both DISCERN-R and DISCERN-C surpassed the median recall on this task by a fair margin. Based on the development ablation experiments, this improvement in recall was attributable to two new additions to DISCERN. The first was the annotation of semantic role labels as features for argument detection. The second was the collapsing of support-verbs and event nominals. Detecting nominal triggers with CatVar benefited the event nugget detection, but did not improve event argument extraction without the support-verb collapsing.

However, low precision in all variants resulted in below-median F-scores. Precision overall is a key focus for further improvement for DISCERN. One possible solution would be the implementation of role saturation. Approximately 4.3% of the base fillers found by DISCERN-R in the evaluation data were assigned to multiple roles. If those base fillers were only assigned to the one correct role, precision could increase by up to 4.6%.

Another avenue of improvement for precision is to improve the precision of the semantic role annotations. Most of the arguments

Run	Plain			Type			Realis			Both		
	P	R	F1	P	R	F1	P	R	F1	P	R	F1
DISCERN-R	**52%**	42%	**46%**	**47%**	38%	**42%**	**35%**	28%	**31%**	**32%**	26%	**29%**
DISCERN-ML	18%	51%	27%	14%	41%	21%	12%	34%	18%	9%	26%	14%
DISCERN-C	17%	**60%**	26%	13%	**48%**	21%	11%	**40%**	17%	9%	**31%**	14%

Table 5: Event Nugget evaluation precision (P), recall (R), and F-score (F1) results for each DISCERN variant by error type

that were detected due to SRL alone were incorrect. More accurate labels would improve the precision of DISCERN. One potential improvement would be to adapt the semantic role labeler to the domain to improve accuracy. Another would be to migrate to a deeper semantic parser, such as TRIPS (Allen et al., 2008).

5 Conclusion and Future Work

In this paper we presented our results on the three variants of DISCERN. The first main contribution of this work is the web interface that improved efficiency during system development by enabling quick revisions to linguistic rules and examination of their effect on precision and recall at runtime.

Our second contribution is the merger between support-verbs and event nominals for detection of event triggers. This process, in conjunction with CatVar, greatly improved the recall of event argument detection on the evaluation data. The combined system variant DISCERN-C, which outperforms DISCERN-R and DISCERN-ML, surpassed the median recall value.

Future efforts to improve DISCERN will focus on improving precision. There are several potential directions for improving precision of DISCERN. One possible solution is to explore a joint learning method for event triggers and event argument extraction. Another possible solution would be the implementation of role saturation so that the base fillers will be assigned only to the one correct role. Adapting semantic role labeler to the specific domain may be another option or migrating to a deeper semantic parser, like TRIPS (Allen et al., 2008) for an overall improvement in semantic parsing accuracy. Additionally, a better handling of events expressed by multi-word expressions might also lead to more precise event nugget detection. Finally, currently the dependency parse plays a small role in the current DISCERN system, as only the trigger's immediate dependents are taken into account. In the future, we seek to take further advantage of the information provided by the dependency parses.

References

Alfonseca, E., Pighin, D., and Garrido, G. (2013). Heady: News headline abstraction through event pattern clustering. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, page 1243–1253.

Allen, J. F., Swift, M., and de Beaumont, W. (2008). Deep semantic analysis of text. In *Proceedings of the 2008 Conference on Semantics in Text Processing*, STEP '08, pages 343–354, Stroudsburg, PA, USA. Association for Computational Linguistics.

Carreras, X. and Màrquez, L. (2005). Introduction to the CoNLL-2005 shared task: Semantic role labeling. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, pages 152–164. Association for Computational Linguistics.

Chen, J., O'Gorman, T., Wu, S., Stowe, K., and Palmer, M. (2014). Clearevent: A semantically motivated event extraction system. In *Proceedings of the NIST TAC KBP 2014 Event Track*.

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., and Kuksa, P. (2011). Natural language processing (almost) from scratch. *J. Mach. Learn. Res.*, 12:2493–2537.

Dorr, B., Park, C., and Park, C. (2003). Cat-Var : A database of categorial variations for English. In *Proceedings of the North American Association for Computational Linguistics*, pages 96–102, Edmonton, Canada.

Dorr, B. J., Petrovic, M., Allen, J. F., Teng, C. M., and Dalton, A. (2014). Discovering and characterizing emerging events in big data (DISCERN). In *Proceedings of the AAAI Fall Symposium Natural Language Access to Big Data*.

Exner, P. and Nugues, P. (2011). Using semantic role labeling to extract evetns from wikipedia. In *Proceedings of the Workshop on Detection, Representation, and Exploitation of Events in the Semantic Web*, pages 23–24.

Ferguson, G. M., Allen, J. F., Miller, B. W., and Ringger, E. K. (1996). The design and implementation of the TRAINS-96 system: A prototype mixed-initiative planning assistant. Technical report, University of Rochester.

Grishman, R., Westbrook, D., and Meyers, A. (2005). NYU's English ACE 2005 system description. In *Proceedings of the ACE Evaluation Workshop*.

Levin, B. (1993). *English Verb Classes and Alternation, A Preliminary Investigation*. The University of Chicago Press.

Lewis, M. and Steedman, M. (2013). Unsupervised induction of crosslingual semantic relations. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, page 681–692.

Mannem, P., Ma, C., Fern, X., Tadepalli, P., Dietterich, T., and Doppa, J. (2014). Oregon state university at tac kbp 2014. In *Proceedings of the NIST TAC KBP 2014 Event Track*.

McClosky, D., Surdeanu, M., and Manning, C. D. (2011). Event extraction as dependency parsing. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*.

Mitamura, T., Yamakawa, Y., Holm, S., Song, Z., Bies, A., Kulick, S., and Strassel, S. (2015). Event nugget annotation: Processes and issues. In *Proceedings of the 3rd Workshop on EVENTS at the NAACL-HLT 2015*, page 66–76.

Nakashole, N., Weikum, G., and Suchanek, F. (2012). Patty: a taxonomy of relational patterns with semantic types. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, page 1135–1145.

NIST (2014). TAC KBP 2014 Event Track. http://www.nist.gov/tac/2014/KBP/Event/index.html.

NIST (2015). TAC KBP 2015 Event Track. http://www.nist.gov/tac/2015/KBP/Event/index.html.

Quinlan, J. R. (1986). Induction of decision trees. *MACH. LEARN*, 1:81–106.

Riloff, E. (1993). Automatically constructing a dictionary for information extraction tasks. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*.

Roberts, K. and Harabagiu, S. M. (2011). Detecting new and emerging events in streaming news documents. *International Journal of Semantic Computing*, 5(4):407–431.

Rusu, D., Hodson, J., and Kimball, A. (2014). Unsupervised techniques for extracting and clustering complex events in news. *Proceedings of the 2nd Workshop on EVENTS: Definition, Detection, Coreference, and Representation*, pages 26–34.

Sammons, M., Song, Y., Wang, R., Kundu, G., Tsai, C.-T., Upadhyay, S., Ancha, S., Mayhew, S., and Roth, D. (2014). Overview of ui-ccg systems for event argument extraction, entity discovery and linking, and slot filler validation. In *Proceedings of the NIST TAC KBP 2014 Event Track*.

Schuler, K. K. (2005). *Verbnet: A Broad-coverage, Comprehensive Verb Lexicon*. PhD thesis, University of Pennsylvania, Philadelphia, PA, USA. AAI3179808.

Tjong Kim Sang, E. F. and De Meulder, F. (2003). Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, pages 142–147. Association for Computational Linguistics.