Proceedings of the Twenty-Ninth International
Florida Artificial Intelligence Research Society Conference

Event Nugget Detection and Argument Extraction with DISCERN

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

Abstract

This paper addresses the problem of detecting information about events from unstructured text. An event-detection system, DISCERN, is presented; its three variants DISCERN-R (rule-based), DISCERN-ML (machine-learned), and DISCERN-C (combined), were evaluated in the NIST TAC KBP 2015 Event Nugget Detection and Event Argument Extraction and Linking tasks. Three contributions of this work are: (a) an approach to collapsing support verb and event nominals that improved recall of argument linking, (b) a new linguist-in-the-loop paradigm that enables quick changes to linguistic rules and examination of their effect on precision and recall at runtime, (c) an analysis of the synergy between the semantic and syntactic features. Results of experimentation with event-detection approaches indicate that linguistically-informed rules can improve precision and machine-learned systems can improve recall. Future refinements to the combination of linguistic and machine learning approaches may involve making better use of the complementarity of these approaches.

1 Introduction

With increasingly large volumes of textual data, most of which is unstructured, it has become necessary to build and apply automatic systems for extraction of information for analysis of data that is too large for fully manual processing. A broad-scale automatic detection and characterization of events of interest (e.g., a natural disaster, a new scientific breakthrough, a terrorist event, or an epidemic) in textual data streams is vital to any tools that can help us increase situation awareness in the rapidly changing world.

This paper addresses the problem of detecting information about events from unstructured text, using linguistic features associated with events. The approach enables the characterization of an event mention and the arguments of the associated event. The resulting event detection system, DISCERN (Discovering and Characterizing Emerging Events), was evaluated in the Event track of the Text Analysis Conference at NIST. This track focused on detection of information about events and their arguments from unstructured text in both formal (news genre) and informal (social media) texts.

DISCERN was evaluated for its performance on two tasks: (1) Event Nugget Detection (EN); and (2) Event Argument Extraction and Linking (EAL). EN involves identification of explicit event mentions, called "nuggets" or "triggers", in English texts. The relevant event types/subtypes are taken from the Rich ERE annotation guidelines (LDC 2015b).[1] The examples in 1 and 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 or a multi-word expression, respectively. The EN task additionally involves identifying a realis state (ACTUAL, GENERIC, OTHER) for each mention.[2]

1. The **attack** by insurgents occurred on Saturday.
2. Kennedy was **shot dead** by Oswald.

EAL involves extracting information about entities in events, the role they play, and times or locations of the event. For example, in sentence 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. The realis state is identified for this task as well.

In this paper, we describe three variants of the DISCERN system, DISCERN-R (rule-based), DISCERN-ML (machine-learned), and DISCERN-C (combined). We present our development approach and results, as applied to the evaluation data set from NIST TAC KBP 2015 for the two tasks mentioned above. Finally, we discuss our findings with regard to the complementary nature of the linguistically informed and machine-learned systems.

2 Related Work

Several prior systems used syntax-based approaches for tasks related to event detection. For example, Riloff (1993)

[1] The Events evaluated in the TAC 2015 Evaluation are divided into 9 types, each with a number of subtypes (for a total of 38 pairings). Examples of types include Business, Conflict, Contact, Manufacture, etc. Examples of subtypes include Attack, Meet, Marry, Nominate, and Acquit.

[2] ACTUAL is used for events that actually happened as in *attacked*; GENERIC refers to general or habitual events *Weapon sales are a problem*; OTHER is used for all the other types of events, e.g., future, hypothetical or non-generic negated events.

used syntactic patterns, while Grishman, Westbrook, and Meyers (2005), McClosky, Surdeanu, and Manning (2011), and Mannem et al. (2014) used a combination of syntactic patterns and statistical classifiers. Dependency parsing has been used quite widely for relation and event extraction (*c.f.* Alfonseca, Pighin, and Garrido (2013), Rusu, Hodson, and Kimball (2014)). While syntactic patterns can help in event detection, an accurate characterization of an event requires semantic context. Chen et al. (2014) designed ClearEvent that takes both the semantic and syntactic levels of analysis in the event detection task with relative success.

Much like the ClearEvent system, DISCERN 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) enables a more robust event detection capability, starting with syntactic dependency relations upon which semantic analysis is applied. More specifically, the use of *categorial variations* from CatVar (Dorr and Habash 2003) in DISCERN shows promising results for a wider coverage of event triggers beyond what would be available in the ClearEvent system.

3 The Process of Event Detection

DISCERN was designed to detect a set of events specified in the NIST (2014) and NIST (2015) Event tasks.

3.1 Preprocessing the Data

The first step of preprocessing was the application of the Stanford CoreNLP Natural Language Processing Toolkit (Manning et al. 2014) to all documents in the development set: the TAC 2014 KBP Event Argument Extraction Evaluation Assessment Results (LDC 2014). The same preprocessing steps were later applied to the TAC-2015 evaluation set (LDC 2015a) during the actual evaluation. Using this toolkit, all documents were stripped of XML, tokenized, and split into sentences. Each sentence was then parsed, POS-tagged, and named-entity tagged (NER). Additionally, coreference resolution was applied to all named entities.

Following the steps above, each annotated sentence was passed through a pipeline wherein additional lexical and semantic resources were automatically added to improve DISCERN's capabilities in recognizing patterns. First, variations of lemmas (extracted from CatVar (Dorr and Habash 2003), a database of 63,000 clusters of lemmas and their categorial variants) were added as Word-POS pairs. The primary benefit of CatVar was that it extended the ability to identify possible triggers beyond only verbal lemmas for an event, for example in phrases like "the *destruction* of the city". Next, all tokens were labeled with the semantic role label (SRL) output of SENNA (Collobert et al. 2011) while verb tokens were also augmented with the corresponding verb class from VerbNet (Schuler 2005).

Finally, all above information was used for the application of a merger rule to "collapse" the structure of a phrasal unit containing a support verb (SV) and event nominal. For example, while the dependency parser might pick "declare" as the root of the dependency tree assigned to the phrase "declare bankruptcy", the desired event is *Business.Declare-Bankruptcy*, not *Declare*. The identification of SVs such as "declare" was a crucial step in the accurate assignment of realis values for events with nominal triggers (see 3.2).

3.2 Implementation of DISCERN

Each of the DISCERN variants was applied to the preprocessed data in four steps. First, DISCERN located potential triggers for each event subtype. For example, *Attack* is a subtype of *Conflict*—and a trigger word for this might be *strike*. Each DISCERN variant employed a different strategy for locating potential triggers (see 3.3).

Next, realis was assigned according to a series of linguistically-motivated rules. Its values were based on tense and aspect encoded in the POS tags, negative lemmas, etc. For the cases where the triggers involved SVs and event nominals, realis was assigned after the SV trigger collapsing had taken place, so the anchor for the realis value was the merged result and had the POS of the original SV.

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, SRL (PropBank), named entity (NE) type, and POS annotations.

A canonical argument string (CAS), representing the first mention of each entity argument, was resolved according to Stanford CoreNLP coreference annotations where available. For NEs, entity type was used to find the full NE string, e.g., "States" becomes "The United States". Time arguments were resolved using timex annotations.

3.3 Three DISCERN Variants

The DISCERN-R variant applied linguistic rules manually generated in advance for the NIST events to the output from the Stanford Dependency Parser.[3] Triggers for event types were identified 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, the *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, an entity was extracted with the dependency relations *nsubj* (subject for a verb) or *poss* (possessive, as in "The United State's invasion of Iraq").

Figure 1 shows a diagram representing DISCERN-R rules with an example from the event sub-type *Justice.Arrest-Jail*. Part 1 of the rule determines the event subtype to be *Justice.Arrest-Jail* based on the lemma. Part 2

[3] A total of 72 rules for 38 event types/subtypes were developed by two linguistic experts, at a rate of approximately one hour per rule. A large portion of the time was spent mapping event types to lemmas that could serve as triggers; semi-automation of this step based on thesaurus look-up will speed up this process in the future.

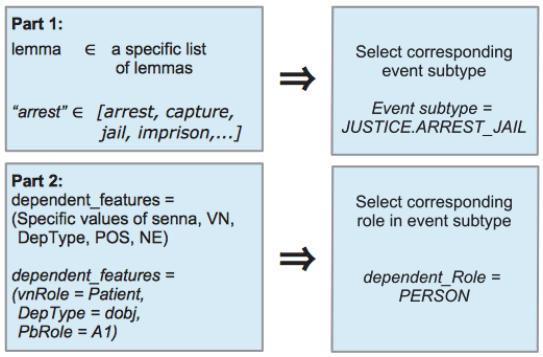

Figure 1: A representative diagram of a DISCERN-R rule with an example from the *Justice.Arrest-Jail* event sub-type.

determines the roles for various dependents (possible arguments) of the lemma in the event subtype based on a variety of semantic and syntactic features; this is done for each role allowed by that event.

DISCERN-ML employs a supervised ML algorithm to induce a random forest of decision tree rules. Ten decision trees were trained on 10 random partitions of the training data (Rich ERE Training, 2015 EAL Training and 2014 EA Assessments), where the sample/partition size was 66% of the size of the entire training set. If a majority of trees determined a token was a trigger for a given event, the algorithm checked dependents for possible arguments.

A variation of the iterative dichotomiser 3 (ID3) algorithm (Quinlan 1986) was used to generate a decision tree for every role for each event type. The algorithm created a decision tree by greedily splitting training data based on the attribute that maximizes information gain of the partition. The information gain $IG(A, S)$ for attribute A on data subset S was

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

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 the set S defined as

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

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

Figure 2 shows a partial representation of one of the trees trained to identify arguments to fill the Entity role of the *Contact.Meet* event sub-type. Each node splits to the left or right depending on the value of a feature. This decision tree shows that if an argument is the direct object of the trigger and is a NE of type "NUMBER" or "null" (i.e. it is not a NE), then the argument fills the Entity role.

Any candidate trigger voted on by a majority of trees in the random forest was assigned a realis and a set of arguments (see 3.2). Each trigger detection tree had an associated argument detection decision tree and arguments were also chosen by majority vote, with each tree voting if its associated trigger detection tree voted for that trigger.

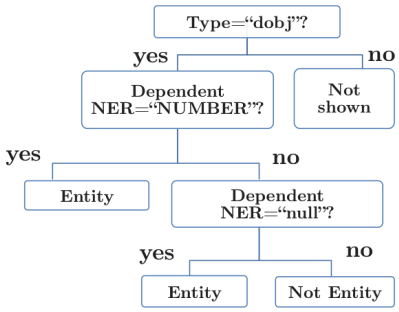

Figure 2: A partial decision tree for the Entity role of the *Contact.Meet* event sub-type.

The DISCERN-ML variant could learn a large number of rules from the training data, but no rules could be generated for any unobserved conditions. By contrast, the rules associated with DISCERN-R could capture generalizations that were not necessarily observed in the training data but were derivable from general linguistic knowledge. Hence, one would expect that the DISCERN-ML system and the DISCERN-R system could discover complementary rules to each other. To leverage this complementarity, a third system, DISCERN-C, operated on the basis of both linguistic (hand-generated) and machine-learned knowledge.

DISCERN-C combined the two sets of rules from the other two runs (the DISCERN-R rules and the DISCERN-ML random forest of decision trees) by applying a weighted voting scheme such that the DISCERN-R rules counted for 5 votes while each DISCERN-ML decision tree counted for 1 (i.e. the DISCERN-R rules weighed 5 times more than the DISCERN-ML rules). As a result, neither DISCERN-R nor DISCERN-ML could unilaterally decide on nuggets or arguments. The fact that both DISCERN-R and DISCERN-ML followed roughly the same execution path (see 3.2) allowed DISCERN-C to compare the output of both without reconfiguring either system.

4 Linguist-In-The-Loop Paradigm

A new Linguist-In-The-Loop paradigm was used for rapid evaluation and testing of linguistic rules for DISCERN-R. Using a web-based interface, the linguist was able to make informed decisions about where to focus rule-development efforts and to see the immediate effects of those decisions.[4] The web interface provided immediate access to: (i) 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 versus false negative) and location (basefiller, event type, event role, realis, or other) per run, and (iii) an in-depth view of the annotations and parse structure of each sentence. Each new rule could be applied immediately on the development data to reveal whether it effectively captured the syntactic and semantic phenomena it targeted while improving the overall system performance.

[4]Parts of the web interface utilize licensed data from LDC. We are currently exploring approaches to make the interface more ac-

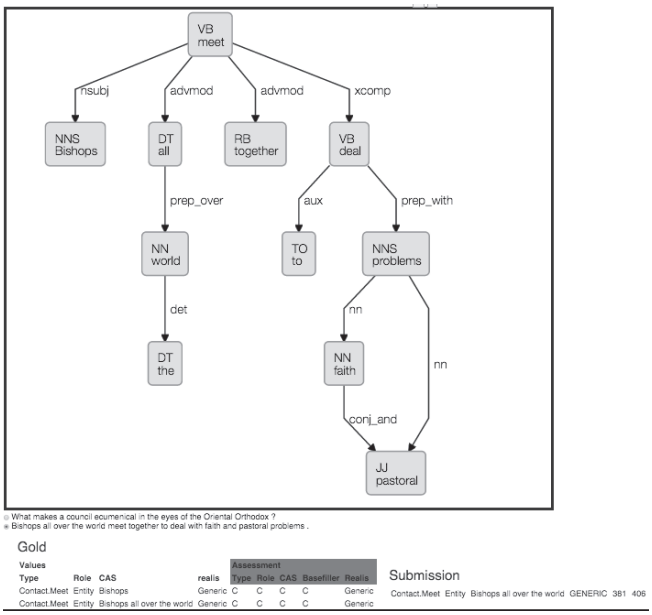

Figure 3: DISCERN sentence detail interface example.

This paradigm proved to be particularly useful for the addition of SENNA-style SRL information to the rules employed by DISCERN. Figure 3 shows a portion of the document view that included a *Contact.Meet* event. The insights gained through improved visualizations and structured layout of the document content resulted in a 7% relative improvement in F_1 score in our preliminary runs on the development data. This experience suggested that this new paradigm allows informants to easily identify critical flaws, improving the rate at which they could develop DISCERN.

5 Results

The three variants of DISCERN were submitted to the 2015 Text Analytics Conference for the Event Argument Linking and Event Nugget tasks. Submissions to these tasks identified event nuggets and arguments according to LDC's rich entity, relation, and event (Rich ERE) annotation guidelines (LDC 2015b). Table 1 shows the performance of each variant of DISCERN on the EN evaluation dataset. With respect to precision and recall, DISCERN-R was surprisingly balanced in contrast to the other two variants—an indication that linguistic rules support comprehensive event detection without introducing the high false-positive rate of the DISCERN-ML variant. In fact, the precision of the DISCERN-R variant was more than three times higher than that of both DISCERN-ML and DISCERN-C.

DISCERN-ML's rules identified more nuggets than there were nuggets. Its rules also overfit the training data, lowering precision. DISCERN-C neared the precision of DISCERN-ML—indicating many of the incorrect nuggets detected by the DISCERN-ML system had a strong enough majority to override the extra votes of DISCERN-R. This

cessible and available to the public.

Run	Precision	Recall	F-Score
DISCERN-R	**31.7%**	25.6%	**28.4%**
DISCERN-ML	9.3%	26.0%	13.7%
DISCERN-C	8.6%	**31.5%**	13.6%
Median TAC	52.6%	29.8%	34.8%

Table 1: Results from final DISCERN event nugget (EN) variants on evaluation data. Median TAC represents median of the 14 participants in the TAC KBP 2015 EN task.

suggests that the extent of overlap in the tree's training data caused the system to be overconfident for incorrect nuggets.

The lower precision of DISCERN-ML and DISCERN-C also suggests that the decision trees did not generalize as well as the hand-crafted rules. This shortcoming could be alleviated by pruning the learned decision trees to create more general leaf nodes. Alternatively, a new combined system could be created that uses the high precision rules from DISCERN-R for event nugget (trigger) detection and uses the combination of linguistic rules and machine learning for event argument extraction.

For realis assignment, a set of rules were designed to assign realis values to simple but common syntactic constructions. Each DISCERN variant used the same set of realis rules. The precision of each variant was tested during the EN evaluation, both with and without inclusion of realis. A reduction of 40% precision resulted from the addition of realis, as its inclusion introduced a level of complexity beyond what the hand-crafted realis rules were able to characterize. However, the rules captured the most common constructions (60%); in future work, more comprehensive rules might improve overall EN performance.

Table 2 shows the results of the DISCERN EAL system for each variant. The better recall for DISCERN-R than DISCERN-ML on the EAL task was due to the fact that linguistic resources (e.g., VerbNet and PropBank) allowed for the development of rules that encoded generalizations not observed in training data. This captured arguments that DISCERN-ML missed, but also led to reduced precision.

DISCERN-R outperformed both DISCERN-ML and DISCERN-C in precision for event argument extraction, in part because DISCERN-R was able to identify a larger portion of event nuggets correctly than the other variants, as seen in the results in Table 1. We will shortly discuss a set of ablation studies that further explore how various features contributed to the precision of DISCERN-R.

DISCERN-ML's precision on the EAL task was almost the same as its precision on the EN task, implying that when DISCERN-ML detected the correct nugget, any arguments it found were very likely to be correct. This was because the decision space for DISCERN-ML was smaller for the arguments than it was for the nuggets, as it only needed to search for arguments among a nugget's dependents in the dependency tree. In addition, each event sub-type only allowed between 3 to 9 possible argument roles. The DISCERN-ML algorithm was therefore better suited to identifying event arguments if given the correctly identified event nuggets.

The performance of all system variants decreased on EAL

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 TAC	30.7%	11.7%	16.9%
Human	73.6%	39.4%	51.4%

Table 2: Results from DISCERN event argument (EAL) variants on evaluation data. 'Median TAC' (baseline) represents the median of the 6 participants in the TAC KBP 2015 EAL task, while 'Human' represents manual annotation.

SV	+	-	-	-	-	-	-
SRL	+	+	-	-	-	+	+
NER	+	+	+	-	-	+	-
CatVar	+	+	+	+	-	-	+
Depend.	+	+	+	+	+	+	-
Precision	10.9	10.9	12.0	11.0	11.7	**12.1**	10.9
Recall	**5.5**	5.4	3.8	3.8	3.7	3.7	5.0
F-Score	**7.3**	7.2	5.7	5.6	5.6	5.6	6.9

Table 3: Ablation results on EAL development data showing the effects of five features with rules from DISCERN-R.

in comparison to EN, which was expected as any errors in EN carried forward to EAL. Despite its better performance on precision compared to the other two variants, DISCERN-R's performance decreased more dramatically on the EAL task than the other variants, with a 15% absolute F-score drop. Further rule refinements are needed for a more accurate identification of event arguments and roles.

DISCERN-C employed an approach to merging machine-learned and linguistic knowledge based on the assumption that its components' capabilities were relatively equally balanced. It performed better than either of the two component approaches on recall, indicating the benefit it received from complementarity of the two approaches. Precision in DISCERN-C was lower than DISCERN-ML in the EN task, yet higher than DISCERN-ML in the EAL task. This is a result of the restriction that only components that have voted for a nugget may vote for its arguments.

In EAL, both DISCERN-ML and DISCERN-R would likely have a hypothesis about the application of a rule and the arrangement of arguments, which would allow information to be combined more effectively. Future work on amplification of the strengths and dampening of the weaknesses of the symbolic and machine-learned approaches may bring greater benefits. For example DISCERN-ML found a large number of incorrect mentions in the open-ended nugget problem, but found only a few accurate roles in the more constrained argument task. Because DISCERN-R had access to linguistic knowledge, it found a number of roles that were not learned by DISCERN-ML, as these were not observed in the training data. These results suggest that the injection of linguistic knowledge was most effective when the decision space was too big to learn easily.

Table 3 presents the results of an ablation experiment on the development data, using the rules from DISCERN-R, to determine the benefits of a number of features: SV collapsing, SRLs, NE recognition, CatVar, and dependency types on the EAL task. DISCERN used either syntactic dependencies or SRL relations to search for potential arguments, so the experiments also explored the use of SRL with no dependency information or dependency-based features (CatVar was still used to find triggers).

Without SV merging, there was a drop in recall; without SRLs, it was even larger. This was presumably due to the fact that SRL helped the system identify arguments corresponding to participant roles in an event by providing the semantic links between a verb and its arguments. DISCERN-R relied on NER to eliminate inappropriate argument roles (e.g., a PERSON entity cannot fill the role of a Crime argument); without NER, precision dropped considerably. Additionally, without the variants from CatVar, DISCERN-R missed syntactic variations, resulting in a drop in recall.

A drop in recall was observed when both CatVar and SV rules were excluded; the process of collapsing SVs captured arguments of nominal triggers that CatVar then found. In fact, the combination of the two contributed almost as much to recall as SRL did. DISCERN captured many SV instances with a list of only 42 different verbs. Although very large, CatVar did not provide an exhaustive listing of variations and could be expanded using verb-noun lists and dictionaries. Improving either of these resources would also improve DISCERN's performance.

The final column of Table 3 shows the argument detection results with only SRL and CatVar. More than one in five arguments found by the system with only SRL relied upon it. This reinforces the hypothesis that semantic roles provided the largest boost to event argument recall. However, the low precision indicates many arguments that were detected using the SRL feature were actually incorrect—although not enough to outweigh the benefits to F-score.

Based on the development ablation experiments, this improvement in recall was attributable to two new additions to DISCERN: adding SRLs as features for argument detection, and collapsing SVs and event nominals. Detecting nominal triggers with CatVar benefited the EN task, but did not improve the EAL task without the SV collapsing.

A final observation about our system variants is based on the comparison of our results with the baseline in Table 2. The DISCERN-R and DISCERN-C variants have a recall better than baseline (the TAC Median). However, the comparison indicates a need for improvements in precision. Low precision from all variants also resulted in below-median F-scores. One possible solution to low precision would be implementing semantic role constraints to ensure each argument was assigned to at most one role. Of the base fillers found by DISCERN-R in the data, 4.3% were assigned to multiple roles. If those base fillers were only assigned to the one correct role, precision would increase by up to 4.6%.

The low rate of human performance in Table 2 shows the difficulty of this task for human annotators to perform and agree upon. Even so, we see many areas for future work.

6 Conclusion and Future Work

In this paper we presented results on three variants of DISCERN. The first main contribution of this work was show-

ing the benefit of merging SVs and event nominals together with semantic information such as CatVar and SRLs for event trigger detection, which improved the recall of event argument detection on the evaluation data. Our second contribution was a new linguist-in-the-loop paradigm for rapid iteration and immediate verification of rule changes. The third contribution was the finding that using semantics (e.g., SRLs) could help in cases where syntactic information (e.g., dependencies) may fall short (e.g., when the dependency labels are too heterogeneous). But care must be taken with the use of SRLs, as they also have their own limitations. For example, some SRLs might be too general and automatic SRL annotations may not be 100% accurate.

A key insight from these experiments is the utility of merging linguistic knowledge with machine learning for NLP. The rules from DISCERN-R largely represent a priori linguistic knowledge, with the web interface facilitating the encoding of this knowledge. Therefore, the human effort and time cost of encoding the knowledge is largely independent of data size. On the other hand, machine learning techniques benefit from additional data, finding patterns and special cases to fill in missing knowledge. Integrating the two together is a challenge with great potential.

There are several potential directions for improving DISCERN. A portion of the errors involved the assignment of the same role to multiple arguments while other roles remained unassigned (e.g., two arguments were identified for a *Conflict.Attack* event, but both were marked as "Attacker"; "Victim" was left unassigned). Semantic role constraints could be implemented to ensure that all the roles available to an event were used only once, thus increasing precision.

Another area of future work might be joint learning of event triggers and arguments to eliminate error propagation from sequential application. In addition, adapting semantic-role labeling to the specific domain or migrating to a deeper semantic parser such as TRIPS (Allen, Swift, and de Beaumont 2008) could result in overall improvement in semantic parsing accuracy. A better handling of events expressed by multi-word expressions might also lead to more precise EN detection. Finally, a detailed investigation into the performance of each variant on different categories of events may lead to insights about the nature of events, and eventually aid in better detection of similar events.[5] Finally, further leveraging of the strengths of DISCERN-R and DISCERN-ML may enable improved performance of DISCERN-C.

7 Acknowledgements

This work was supported, in part, by the Defense Advanced Research Projects Agency (DARPA) under Contract No. FA8750-12-2-0348, the DARPA Big Mechanism program under ARO contract W911NF-14-1-0391, and the Nuance Foundation.

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*, 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, 343–354. Stroudsburg, PA, USA: 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., and Habash, N. 2003. CatVar : A database of categorial variations for English. In *Proceedings of the North American Association for Computational Linguistics*, 96–102.

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*.

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

LDC. 2014. LDC2014E88: TAC 2014 KBP English event argument extraction evaluation assessment results V2.0. Distributed by Linguistic Data Consortium.

LDC. 2015a. Distributed by Linguistic Data Consortium.

LDC. 2015b. DEFT rich ERE annotation guidelines: Events V2.7. Distributed by Linguistic Data Consortium.

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*.

Manning, C. D.; Surdeanu, M.; Bauer, J.; Finkel, J.; Bethard, S. J.; and McClosky, D. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 55–60.

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*, 66–76.

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*.

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* 26–34.

Schuler, K. K. 2005. *Verbnet: A Broad-coverage, Comprehensive Verb Lexicon*. Ph.D. Dissertation, Philadelphia, PA, USA. AAI3179808.

[5] Currently, TAC provides system-level results that are not granular enough for a category-level analysis across systems.