Improving DISCERN with Deep Learning

Greg Dubbin, Archna Bhatia, Bonnie Dorr, Adam Dalton
Kristy Hollingshead, **Suriya Kandaswamy**, **Ian Perera** and **Jena D. Hwang**
Institute for Human and Machine Cognition, Ocala, FL*

Abstract

IHMC designed and implemented two variants of an event-detection system, one that used manually created rules (DISCERN-R) and another one that used rules learned using multiple deep neural networks (DISCERN-D). The former uses very rich linguistic resources (VerbNet, CatVar, Semantic Role Labeling, NER, POS tagging, dependency parsing, and coreference resolution) and the latter supplants these features with learned embedding vectors. These systems were applied to two tasks in the NIST TAC KBP 2016 Event Track: Event Nugget Detection and Coreference (EN) and Event Argument Extraction and Linking (EAL) for English language. Additionally, the neural network system was used for these two tasks in Spanish.

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 2016 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 and Coreference (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 multiword 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

*This work was supported, in part, by the Defense Advanced Research Projects Agency (DARPA) under Contract No. FA8750-12-2-0348 (Adam Dalton and Greg Dubbin), and (Suriya Kandaswamy) through a summer internship funded through the Institute for Human and Machine Cognition.

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 two variants of our event detection system for the two tasks described above, as applied to a development data set from NIST TAC 2015. We also discuss the application of these variants to the NIST TAC 2016 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 two system variants we built in detail, and section 4 presents a discussion of the results for our two 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), Rusu et al. (2014), and Sun et al. (2015).

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. The growing need for bulk semantic information extraction across differing domains and genres has led to an expansion in annotated data, e.g., Bies et al. (2016); Song et al. (2016), ontologies with an event-based semantics, e.g., Bonial et al. (2016), and development of frameworks to build such resources manually or semi-automatically, e.g., Mirza and Tonelli (2014); Mostafazadeh et al. (2016); Nakamura and Kawahara (2016).

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 (Bhatia et al., 2015; Dorr et al., 2014; Dubbin et al., 2016), 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).

Another approach is based on *deep-learning*, using multi-layered neural networks to represent words and phrases in a continuous vector-space, called an *embedding* (Bengio et al., 2003; Hermann et al., 2014; Mikolov et al., 2013; Socher et al., 2013). Dasigi and Hovy (2014) use a recursive model of event structure to create a vector representation of events which can be used to distinguish between actual and anomalous events. Feng et al. (2016) combine a bi-directional, recurrent Long-Short Term Memory neural network (LSTM) with a convolutional neural network to create an embedding that captures local and long-range context that detects event trigger words. Additionally, Nguyen et al. (2016) also experiments with combining recurrent and convolutional neural networks to jointly detect event triggers as well as event arguments.

3 The Process of Event Detection

We developed two 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), NIST (2015) and NIST (2016) Event tasks. Specifically, these variants were namely DISCERN-R (that used manually created rules) and DISCERN-D (that used rules learned using a neural net system).[1] The description of the process of event detection used by each variant of the DISCERN system is provided below.

DISCERN-R relies on a resource of semantically-linked categorial variants (CatVar), to develop a mapping between verbs (and verb classes) to each event type of interest. CatVar clusters were used to extend the verb-centric syntactic patterns to additional patterns containing part-of-speech variants, e.g., "nabbing" was mapped to "nab" which was then mapped to "Justice.Arrest-Jail".

DISCERN-D excludes all of the features used by DISCERN-R except for NER, coreference resolution, and dependency parses. Words and context are represented by learned embedding vectors. These embeddings capture much of the same semantic and syntactic information as the features of earlier approaches, while allowing the discovery of more complicated pat-

[1] The Events evaluated in the TAC 2016 Evaluation are divided into 8 types, each with a number of subtypes: Conflict (Attack, Demonstrate), Contact (Broadcast, Contact, Correspondence, Meet), Justice (Arrest-Jail), Life (Die, Injure), Manufacture (Artifact), Movement (Transport-Artifact, Transport-Person), Personnel (Elect, End-Position, Start-Position), Transaction (Transaction, Transfer-Money, Transfer-Ownership). An area of future work is to expand from these more general categories to more refined ones, and also to add new domains.

terns when fed through multiple layers of a neural network.

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

1. Preprocessing the data: Dependency and constituency parses were generated for each sentence. These were subsequently annotated with linguistic features, such as VerbNet, CatVar, SRL, NER, and coreference for the DISCERN-R variant, and with NER and coreference for the DISCERN-D variant.

2. Implementation of DISCERN: Predefined rules were applied to identify the event triggers and their arguments, as well as assigning realis values to them. The two variants of DISCERN varied with respect to the rules that were applied (each variant created its own set of rules using different approaches, as described in sections 3.3 and 3.4).

The two stages of the event detection process are described in sections 3.1 and 3.2. In sections 3.3 and 3.4, the differences among the two DISCERN variants are discussed with respect to rule creation and learning.

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 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.

Each annotated sentence representation from a document was passed through a pipeline wherein additional lexical and semantic resources were automatically added to improve DISCERN-R'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. Support-verb detection was also used in determining the realis values for events with nominal triggers, as described in Section 3.3.

A key benefit to deep-learning is the reduced reliance on feature engineering and preprocessing. Accordingly, DISCERN-D only relies on the tokenization, dependency parsing, NER, and coreference annotations from the preprocessing step. Other differences will be expanded on in sections 3.3 and 3.4.

3.2 Implementation of DISCERN

Each of the two 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 children in the dependency tree. 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 Sections 3.3 and 3.4. DISCERN's operation relied on lemma and CatVar annotations primarily to find potential trigger words. CatVar annotations enabled the generalization of results to semantically related variants of the verbs denoting relevant events.

The next step was to determine an event's arguments from its trigger's dependents (i.e., it's children in the dependency tree). As with the first step, the method for detecting arguments was governed by the DISCERN variant. However, each variant generally relied on a combination of some or all of the following features: 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 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 children in the dependency tree 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 children 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").

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 determines the roles for various children (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.

Once a trigger was identified, realis was assigned to the corresponding *anchor* in the dependency tree according to a series of linguistically-motivated rules applied to an anchor and its children. The realis values (ACTUAL, GENERIC, or OTHER) were based on

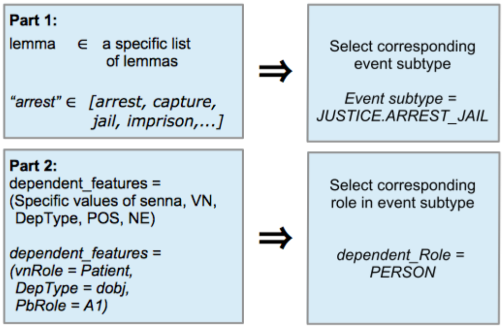
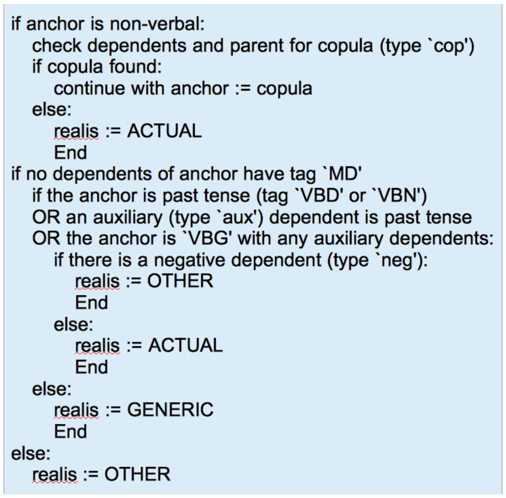

Part 1:
lemma ∈ a specific list of lemmas
"arrest" ∈ [arrest, capture, jail, imprison,…]
⇒
Select corresponding event subtype
Event subtype = JUSTICE.ARREST_JAIL
Part 2:
dependent_features = (Specific values of senna, VN, DepType, POS, NE)
dependent_features = (vnRole = Patient, DepType = dobj, PbRole = A1)
⇒
Select corresponding role in event subtype
dependent_Role = PERSON

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



```
if anchor is non-verbal:
   check dependents and parent 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
```

Figure 2: Realis rules depend on a combination of tense, aspect, POS, and negation.

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 Figure 2.

3.4 DISCERN-D: Based on Deep Neural Networks

DISCERN-D implements deep learning techniques to construct and train neural networks that detect event nuggets and their arguments. Like DISCERN-R, this approach first detects event nugget mentions and then finds arguments using the containing sentence's dependency structure. However, DISCERN-D learns from supervised training data, relying on the data to discover features and patterns, rather than engineering features and rules with linguistic expertise.

3.4.1 Event Nugget Detection

The DISCERN-D event nugget detection algorithm is based on four neural networks. Each network takes a tokenized sentence as input and maps the tokens to learned word embeddings. These embedding sequences are then fed into a bidirectional Long-Short Term Memory (LSTM) network layer (Hochreiter and Schmidhuber, 1997), the output of which is used by a task-specific classifier.

The first neural network is the sentence filter. As Figure 3 shows, the sentence filter network merges the final output of the forward and backward LSTMs to create a representation of the entire sentence, which is then fed into a binary classifier. This network predicts whether a sentence has at least one event nugget. Filtering out sentences without event nuggets reduces the imbalance of nugget to non-nugget tokens in the training data, allowing the remaining networks to boost recall without negatively affecting precision.

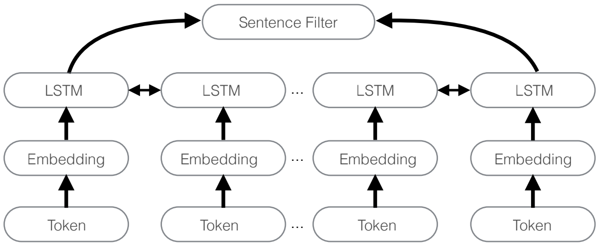


Figure 3: The architecture of the DISCERN-D sentence filter network.

The remaining three networks have roughly the same architecture, shown in Figure 4. In these networks, the hidden state of the bidirectional LSTM is used by the classifiers at each timestep (i.e. each token). The event nugget classifier uses a three state, Beginning, Inside, Outside (BIO), classifier to annotate event nugget phrases. The type and realis of the event are assigned according to the output of the type and realis classifiers for the first token of the nugget. To account for the possibility of a single nugget representing multiple event types, each multi-type combination observed in the training data is encoded as its own class.

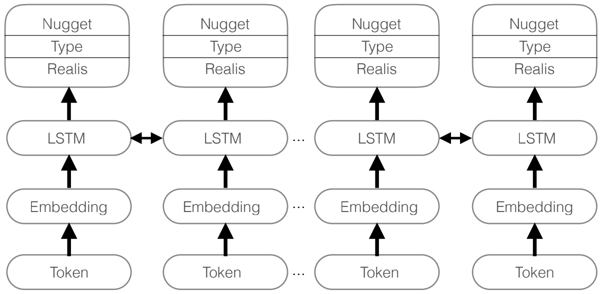


Figure 4: The architecture of the DISCERN-D nugget, event type, and realis networks.

3.4.2 Event Argument Detection

Once DISCERN-D has found all of the event nuggets in a sentence, it generates proposal arguments from the set of named entities and coreferenced entities annotated by Stanford CoreNLP. Each proposal argument in a sentence is potentially the argument of all event nuggets in that sentence, so the event argument network is applied to each <nugget, proposal argument> pair. Each pair is classified as either *none*, meaning the proposal argument is not an argument for that event, or one of the roles allowed by the event subtype. Only the argument role is classified, as the realis is assumed to be the same as that assigned to the paired event nugget.

Figure 5 demonstrates the architecture of the event argument neural network. First, the proposal argument and event nugget phrases are tokenized and sent through two bidirectional LSTM's, creating two phrasal embeddings. Then, the event subtype and realis are also included as features. Finally, the dependency path from the head of the event nugget phrase to the head of the proposal argument phrase is also fed into another bidirectional LSTM, giving the network a sense of the syntactic relationship between the nugget and the argument. The dependency path is represented as a sequence of triples representing the direction of the dependency, the type of the dependency, and the token of the dependent.

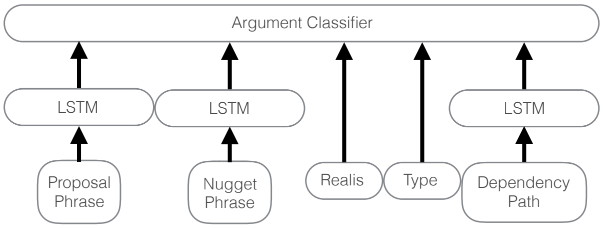


Figure 5: The architecture of the DISCERN-D event argument neural network.

4 Evaluation and Discussion

This section will analyze and compare the results of the two DISCERN variants on development data. The aim of this analysis is to determine the strengths and weaknesses of both approaches. Do these approaches compliment each other? If so, what aspects can or should be integrated to improve the overall DISCERN system? Sections 4.1 and 4.2 analyse the performance of DISCERN on the event nugget and event argument tasks, respectively.

4.1 Event Nuggets

In this section, DISCERN-R and DISCERN-D are evaluated against the TAC KBP 2015 Event Nugget evaluation data set LDC (2015) as a development data set.

Table 1 shows the overall performance of DISCERN-R and DISCERN-D on the development data. The data shows that DISCERN-D outperforms DISCERN-R across all three metrics. DISCERN-D achieved an F-score of 36.6%, a 8% improvement over the 28.4%

DISCERN-R achieved on the development data. This is representative of similar improvements in precision and recall.

Run	Precision	Recall	F1-score
DISCERN-R	31.7%	25.6%	28.4%
DISCERN-D	**41.2%**	**32.9%**	**36.6%**

Table 1: DISCERN-R and DISCERN-D development performance on TAC 2015 Event Nugget evaluation data.

Table 2 breaks down the performance of DISCERN-R and DISCERN-D by scored attribute. This breakdown allows a more thorough analysis of the differences between DISCERN-R and DISCERN-D.

Most of the overall improvement of the DISCERN-D event mention detection approach comes from improved plain mention detection with a plain F-score of 63.5% vs. 46.0%. We attribute this improvement to two key factors: 1) the sentence neural network allows DISCERN-D to skip sentences without event mentions, reducing false-positives; 2) the context provided by the bidirectional LSTMs enables the network to reasonably handle unknown words.

There was also a slight improvement to overall performance because of improved Realis classification. While 67.8% of DISCERN-R Realis labels are assigned correctly, DISCERN-D assigns correct Realis to 74.6% of mentions. DISCERN-R uses a small set of linguistic rules to capture most sentence constructions. DISCERN-D is more generalized and less focused on common special cases.

	Prec		Rec		F1	
	R	D	R	D	R	D
plain	52%	**72%**	42%	**57%**	46%	**64%**
type	47%	**55%**	38%	**44%**	42%	**49%**
realis	35%	**53%**	28%	**43%**	31%	**47%**
all	32%	**41%**	27%	**33%**	28%	**37%**

Table 2: DISCERN-R and DISCERN-D development performance by scoring attribute.

Interestingly, DISCERN-R was more accurate when assigning event types. 90.3% of the event types assigned by DISCERN-R were correct, compared to 76.4% for DISCERN-D. This can be attributed to the way DISCERN-D handles mentions with multiple types. DISCERN-D treats each combination of types assigned to a mention as distinct from either component type. This means that it can only discover combinations assigned in the training data and that information is not shared between combinations that share types. On the other hand, each possibly event type assignment is entirely independent of the other in DISCERN-R.

4.2 Event Arguments

The DISCERN event argument variants were both tested against the dual-agreement results from TAC 2014 KBP Event Argument Extraction Evaluation Assessment Results (LDC, 2014). The development data has been modified to match the ontology of the TAC 2016 KBP Event Argument Extraction evaluation.

Table 3 presents the results of DISCERN-R and DISCERN-D on the EAL development data. These results are not as clear cut as the event nugget performance in section 4.1. The balanced performance of DISCERN-R results in a higher overall F-score (7.7%) than DISCERN-D. DISCERN-D has 50% higher recall than DISCERN-R(11.4% compared to 7.4%).

System	Precision	Recall	F1-score
DISCERN-R	**7.9%**	7.4%	**7.7%**
DISCERN-D	4.9%	**11.4%**	6.8%

Table 3: Event argument performance of DISCERN-R and DISCERN-D on development data.

The relatively low precision of DISCERN-D is the lower result of two factors. First, the

DISCERN-R argument rules only consider arguments that are children of the event nugget. While this misses many arguments (which is why the DISCERN-D recall is higher), it acts as a useful heuristic. This also reduces the feature complexity, as there is only ever one dependency between a nugget and a proposal argument.

The second cause is the lack of semantic information included in the deep neural networks' features. DISCERN-R gets a large performance boost from semantic features like semantic role labels (Dubbin et al., 2016). While word and phrase embeddings have been shown to discover some level of semantics (Hermann et al., 2014; Mikolov et al., 2013; Socher et al., 2013), this usually involves large amounts of training data, which is limited for this task.

5 Conclusion and Future Work

We have presented two variants of our DISCERN Event Detection system that we developed for the TAC KBP 2016 Event Track, viz., a manual rule-based system DISCERN-R and a deep learning based system DISCERN-D. Sections 4.1 and 4.2 demonstrated that even though DISCERN-D had much better EN performance, there is room for improvement in both implementations of DISCERN. These results show that there is benefit to be achieved by incorporating a linguistic and semantic view of the data in event argument detection. Future work will focus on integrating this knowledge without sacrificing the efficiency and power of the machine learning approach.

For example, the data problem could be ameliorated by pretraining the initial layers of the neural network on large amounts of unannotated data. This would require the development of a loss function that applies to nugget and argument phrases as well as decision paths.

DISCERN-D may also benefit from a more semantically-informed sentence structure, like semantic role labels or Abstract Meaning Representations (Banarescu et al., 2013). These would connect semantically linked concepts in sentences, rather than strictly focusing on syntactic links.

Disclaimer

The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of DARPA, ODNI, IARPA, AFRL, or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes notwithstanding any copyright annotation thereon.

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.

Banarescu, L., Bonial, C., Cai, S., Georgescu, M., Griffitt, K., Hermjakob, U., Knight, K., Koehn, P., Palmer, M., and Schneider, N. (2013). Abstract Meaning Representation for Sembanking.

Bengio, Y., Ducharme, R., Vincent, P., and Janvin, C. (2003). A neural probabilistic language model. *J. Mach. Learn. Res.*, 3:1137–1155.

Bhatia, A., Dalton, A., Dorr, B., Dubbin, G., Hollingshead, K., Kandaswamy, S., and Perera, I. (2015). Event argument linking and event nugget detection task: IHMC DISCERN system report. In *Proceedings of NIST TAC KBP*.

Bies, A., Song, Z., Getman, J., Ellis, J., Mott, J., Strassel, S., Palmer, M., Mitamura, T., Freedman, M., Ji, H., and O'Gorman, T. (2016). A Comparison of Event Representations in DEFT. In *Proceedings of the 4th Workshop* *on Events: Definition, Detection, Coreference, and Representation*, pages 27–36.

Bonial, C., Tahmoush, D., Brown, S. W., and Palmer, M. (2016). Multimodal Use of an Upper-Level Event Ontology. In *Proceedings of the 4th Workshop on Events: Definition, Detection, Coreference, and Representation*, pages 18–26.

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*.

Dasigi, P. and Hovy, E. H. (2014). Modeling newswire events using neural networks for anomaly detection. In Hajic, J. and Tsujii, J., editors, *COLING*, pages 1414–1422. ACL.

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*.

Dubbin, G., Bhatia, A., Dorr, B., Dalton, A., Hollingshead, K., Perera, I., Kandaswamy, S., and Hwang, J. (2016). Event Nugget Detection and Argument Extraction with DISCERN. In *Proceedings of the Florida Artificial Intelligence Research Society Conference*.

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.

Feng, X., Huang, L., Tang, D., Ji, H., Qin, B., and Liu, T. (2016). A Language-Independent Neural Network for Event Detection. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 2: Short Papers*.

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*.

Hermann, K. M., Das, D., Weston, J., and Ganchev, K. (2014). Semantic frame identification with distributed word representations. In *Proceedings of the 52th Annual Meeting of the Association for Computational Linguistics*.

Hochreiter, S. and Schmidhuber, J. (1997). Long short-term memory. *Neural Comput.*, 9(8):1735–1780.

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

LDC (2015). LDC2015R26: TAC 2015 event nugget and event coreference linking. Distributed by Linguistic Data Consortium.

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*.

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*, pages 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*.

Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., and Dean, J. (2013). Distributed representations of words and phrases and their compositionality. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc.

Mirza, P. and Tonelli, S. (2014). An analysis of causality between events and its relation to temporal information. In *Proceedings of the 25th International Conference on Computational Linguistics*, pages 23–29.

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.

Mostafazadeh, N., Grealish, A., Chambers, N., Allen, J., and Vanderwende, L. (2016). CaTeRS: Causal and Temporal Relation Scheme for Semantic Annotation of Event Structures. In *Proceedings of the 4th Workshop on Events: Definition, Detection, Coreference, and Representation*, pages 51–61.

Nakamura, T. and Kawahara, D. (2016). Constructing a Dictionary Describing Feature Changes of Arguments in Event Sentences. In *Proceedings of the 4th Workshop on Events: Definition, Detection, Coreference, and Representation*, pages 46–50.

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.

Nguyen, T. H., Cho, K., and Grishman, R. (2016). Joint Event Extraction via Recurrent Neural Networks. In *NAACL-HLT*.

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.

NIST (2016). TAC KBP 2016 Event Track. https://tac.nist.gov//2016/KBP/Event/index.html.

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.

Socher, R., Perelygin, A., Wu, J., Chuang, J., Manning, C. D., Ng, A., and Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 1631–1642, Seattle, Washington, USA. Association for Computational Linguistics.

Song, Z., Bies, A., Strassel, S., Ellis, J., Mitamura, T., Dang, H., Yamakawa, Y., and Holm, S. (2016). Event Nugget and Event Coreference Annotation. In *Proceedings of the 4th Workshop on Events: Definition, Detection, Coreference, and Representation*, pages 37–45.

Sun, R., Zhang, Y., Zhang, M., and Ji, D.-H. (2015). Event-driven headline generation. In *Proceedings of the 53rd Annual Meeting of the ACL*, pages 462–472. Association for Computational Linguistics.