RPI_BLENDER TAC-KBP2016 System Description

Dian Yu, Xiaoman Pan, Boliang Zhang, Lifu Huang, Di Lu, Spencer Whitehead, Heng Ji
Computer Science Department
Rensselaer Polytechnic Institute
jih@rpi.edu

1 Introduction

This year the RPI_BLENDER team participated in four tasks at KBP2016: Tri-lingual Entity Discovery and Linking (section 2), Tri-lingual cold-start slot filling (section 3), Tri-lingual Event Nugget Detection (section 4) and Event Nugget Coreference Resolution (section 5). For each task we developed systems for all three languages: English, Chinese and Spanish.

2 Tri-lingual Entity Discovery and Linking

2.1 Entity Mention Extraction

We used Stanford Corenlp toolkit (Manning et al., 2014b) for English name tagging. To extract name mentions from Chinese and Spanish documents, we use bi-directional LSTMs (Long Short Term Memory) networks which can leverage long distance features. The input of the networks are pre-trained word embeddings and randomly generalized character embeddings. Both word embedding and character embeddings are updated during the training. We used Chinese Wikipedia to pre-train Chinese embeddings, and used the monolingual corpora listed at: http://128.2.220.95/multilingual/ to train Spanish embeddings.

The name tagging training corpora we used include: 1) Chinese OntoNotes 5.0 and EDL2015 Chinese training set, 2) Spanish CoNLL2012 data set and EDL2015 Spanish training set. The Development and test sets of both languages are the EDL 2015 evaluation set. We also applied pre-processing on the training set: 1) removing XML tags, 2) re-segmentating and re-tokenizing, 3) correct name entity tag mistakes that caused by different entity type definition from different sources. For the third preprocessing method aforementioned, we used our entity linking module to correct incorrect tags. For example, in CoNLL training data, "Eurooa" in "Las 4 veces que me he hecho agredir en **Eurooa** han sido por negros!" is tagged as LOC, since our linking module returns "GPE" for "Eurooa", we changed "LOC" to "GPE" in the training data. We also acquired and exploited non-traditional language-specific resources, and incorporate them into a novel neural network framework.

Finally, we apply a cross-lingual knowledge transfer approach we recently developed (Lu et al., 2016) to discover comparable documents in English with rich semantic resources (e.g., Abstract Meaning Representation), and project Entity Discovery and Linking results from English documents back to Chinese and Spanish. We also learned entity priors from a multi-lingual knowledge base.

2.2 Entity Mention Translation

Since the target KB of Tri-lingual EDL is English KB, we first translate Chinese and Spanish entity mentions to English, and then apply an entity linker described in next section 2.3. We utilized name translation dictionaries mined from various approaches described in (Ji et al., 2009). We also developed a new word translation mining approach based on cross-lingual links in Wikipedia. Given a word w, we first filter source language and target language Wikipedia title pairs, then extract all pairs that contain w. Finally, we apply GIZA (Och and

Ney, 2003) to obtain a list of translation candidates of w. If an entity mention cannot be translated, we use Pinyin for Chinese and normalize special characters for Spanish.

2.3 Unsupervised Entity Linking

We utilize a domain and language independent entity linking system (Wang et al., 2015) which is based on an unsupervised collective inference approach. Given a set of English entity mentions $M = \{m_1, m_2, ..., m_n\}$, our system first constructs a graph for all entity mentions based on their co-occurrence within a context window. Then, for each entity mention m, our system uses the surface form dictionary $\langle f, e_1, e_2, ..., e_k \rangle$, where $e_1, e_2, ..., e_k$ is the set of entities with surface form f according to their properties (e.g., labels, names, aliases), to locate a list of candidate entities $e \in E$ and compute the importance score by an entropy based approach (Zheng et al., 2014). Finally, it computes similarity scores for each entity mention and candidate entity pair $\langle m, e \rangle$ and selects the candidate with the highest score as the appropriate entity for linking.

2.4 Experiments

Table 1 summarizes the overall performance on TEDL 2015 evaluation data set. In general entity mention extraction remains a bottleneck of our EDL system.

	DiscF	LinkF	CEAFmF
ENG	0.692	0.605	0.678
CMN	0.689	0.657	0.665
SPA	0.696	0.652	0.640

Table 1: Overall Performance On TEDL 2015 Evaluation Data

We have found for all three languages (English, Chinese and Spanish), training a name tagger from new data from similar time periods as the evaluation data, performs much better than a tagger trained from old data from a decade ago even though with a ten times size. Regardless of the recent success at applying deep neural networks to name tagging which can save efforts at human feature engineering, we have found explicit language-specific resources still provide significant gains.

3 Cold Start Slot Filling

We utilize an unsupervised graph mining method for trigger-driven slot types by deeply exploring the structures of dependency tress. It consists of the following three steps:

- **Step 1 - Candidate Relation Identification:** Construct an extended dependency tree for each sentence including any mention referring to the query entity. Identify candidate slot fillers based on slot type constraints (*e.g.*, the *spouse* fillers are limited to person entities).
- **Step 2 - Trigger Identification:** Measure the importance of each node in the extended dependency tree relative to query and filler, rank them and select the most important ones as the trigger set.
- **Step 3 - Slot Typing:** For any given new slot type, automatically expand a few trigger seeds using the Paraphrase Database (Ganitkevitch et al., 2013). Then we use the expanded trigger set to label the slot types of identified triggers (Section 3.3).

3.1 Candidate Relation Identification

We first present how to build an extended dependency graph for each evidence sentence and generate query and filler candidate mentions.

3.1.1 Extended Dependency Tree Construction

Given a sentence containing N words, we construct an undirected graph $G = (V, E)$, where $V = \{v_1, \ldots, v_N\}$ represents the words in a sentence, E is an edge set, associated with each edge e_{ij} representing a dependency relation between v_i and v_j. We first apply a dependency parser to generate basic uncollapsed dependencies by ignoring the direction of edges. Figure 1 shows the dependency tree built from the example sentence. In addition, we annotate an entity, time or value mention node with its type. For example, in Figure 1, "*Ellen Griffin Dunne*" is annotated as a person, and "*1997*" is annotated as a year. Finally we perform co-reference resolution, which introduces implicit links between nodes that refer to the same entity. We replace any nominal or pronominal entity mention with its coreferential name mention. For example, "*he*" is replaced by "*Dominick Dunne*" in Figure 1. Formally, an extended dependency tree is an annotated tree of entity mentions, phrases and their links.

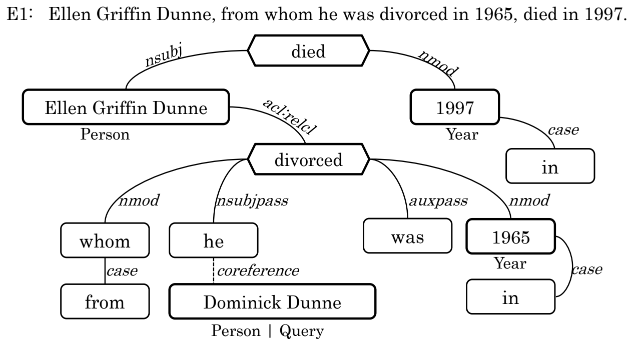

Figure 1: Extended dependency tree for E1.

3.1.2 Relation Arguments Identification

Given a query q and a set of relevant documents, we construct a dependency tree for each sentence. We identify a person entity e as a query mention if e matches the last name of q or e shares two or more tokens with q. For example, "*he/Dominick Dunne*" in Figure 1 is identified as a mention referring to the query *Dominick Dunne*. For each sentence which contains at least one query mention, we regard all other entities, values and time expressions as candidate fillers and generate a set of entity pairs (q, f), where q is a query mention, and f is a candidate filler. In Example $E1$, we can extract three entity pairs (*i.e.*, $\{$*Dominick Dunne*$\} \times \{$*Ellen Griffin Dunne*, 1997, 1965$\}$). For each entity pair, we represent the query mention and the filler candidate as two sets of nodes Q and F respectively, where $Q, F \subseteq V$.

3.2 Trigger Identification

We proceed to introduce an unsupervised graph-based method to identify triggers for each query and candidate filler pair. We rank all trigger candidates and then keep the top ones as the trigger set.

3.2.1 Trigger Candidate Ranking

As we have discussed earlier, we can consider trigger identification problem as finding the important nodes relative to Q and F in G. Algorithms such as Pagerank (Page et al., 1999) are designed to compute the global importance of each node relative to all other nodes in a graph. By redefining the importance according to our preference toward F and Q, we can extend PageRank to generate relative importance scores.

We use the random surfer model (Page et al., 1999) to explain our motivation. Suppose a random surfer keeps visiting adjacent nodes in G at random. The expected percentage of surfers visiting each node converges to the PageRank score. We extend PageRank by introducing a "back probability" β to determine how often surfers jump back to the ***preferred nodes*** (*i.e.*, Q or F) so that the converged score can be used to estimate the relative probability of visiting these preferred nodes.

Given G and a set of preferred nodes R where $R \subseteq V$, we denote the relative importance for all $v \in V$ with respect to R as $I(v \mid R)$, following the work of (White and Smyth, 2003).

For a node v_k, we denote $N(k)$ as the set of neighbors of v_k. We use $\pi(k)$, the k-th component of the vector $\boldsymbol{\pi}$, to denote the stationary distribution of v_k where $1 \leq k \leq |V|$. We define a preference vector $\boldsymbol{p_R} = \{p_1, ..., p_{|V|}\}$ such that the probabilities sum to 1, and p_k denotes the relative importance attached to v_k. p_k is set to $1/|R|$ for $v_k \in R$, otherwise 0. Let $\boldsymbol{A}$ be the matrix corresponding to the graph G where $A_{jk} = 1/|N(k)|$ and $A_{jk} = 0$ otherwise.

For a given $\boldsymbol{p_R}$, we can obtain the personalized PageRank equation (Jeh and Widom, 2003):

$$\boldsymbol{\pi} = (1 - \beta)\boldsymbol{A}\boldsymbol{\pi} + \beta\boldsymbol{p_R} \tag{1}$$

where $\beta \in [0, 1]$ determines how often surfers jump back to the nodes in R. We set $\beta = 0.3$ in our experiment. The solution π to Equation 1 is a steady-state importance distribution induced by $\boldsymbol{p_R}$. Based on a theorem of Markov Theory, a solution π with $\sum_{k=1}^{|V|} \pi(k) = 1$ always exists and is unique (Motwani and Raghavan, 1996).

We define relative importance scores based on the personalized ranks described above, *i.e.*, $I(v \mid R) = \pi(v)$ after convergence, and we compute the importance scores for all the nodes in V relative to Q and F respectively.

A query mention in a sentence is more likely to be involved in multiple relations while a filler is usually associated with only one slot type. Therefore we combine two relative importance scores by assigning a higher priority to $I(v \mid F)$ as follows.

$$\mathcal{I}(v \mid \{Q, F\}) = I(v \mid F) + I(v \mid F) \cdot I(v \mid Q) \quad (2)$$

We discard a trigger candidate if it is (or part of) an entity which can only act as a query or a slot filler. We assume a trigger can only be a noun, verb, adjective, adverb or preposition. In addition, verbs, nouns and adjectives are more informative to be triggers. Thus, we remove any trigger candidate v if it has a higher $\mathcal{I}(v \mid \{Q, F\})$ than the first top-ranked verb/noun/adjective trigger candidate.

For example, we rank the candidate triggers based on the query and slot filler pair ("*Dominick Dunne*", "*Ellen Griffin Dunne*") as shown in Figure 2.

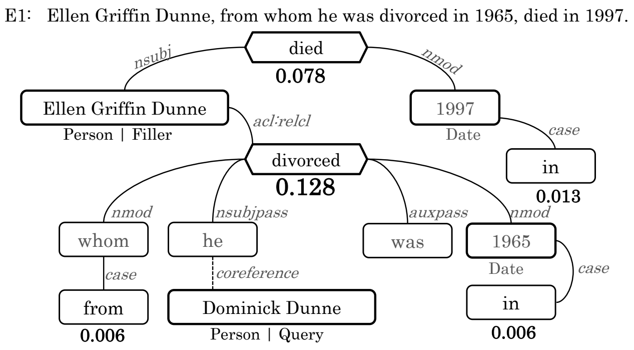

Figure 2: Importance scores of trigger candidates relative to query and filler in E1.

3.2.2 Trigger Candidate Selection

Given Q and F, we can obtain a relative importance score $\mathcal{I}(v \mid \{Q, F\})$ for each candidate trigger node v in V as shown in Section 3.2.1. We denote the set of trigger candidates as $T = \{t_1, \cdots, t_n\}$ where $n \leq |V|$.

Since a relation can be indicated by a single trigger word, a trigger phrase or even multiple non-adjacent trigger words, it is difficult to set a single threshold even for one slot type. Instead, we aim to automatically classify top ranked candidates into one group (*i.e.*, a trigger set) so that they all have similar higher scores compared to other candidates.

Therefore, we define this problem as a clustering task. We mainly consider clustering algorithms which do not require pre-specified number of clusters.

We apply the affinity propagation approach to take as input a collection of real-valued similarity scores between pairs of candidate triggers. Real-valued ***messages*** are exchanged between candidate triggers until a high-quality set of ***exemplars*** (centers of clusters), and corresponding clusters gradually emerges (Frey and Dueck, 2007).

There are two kinds of messages exchanged between candidate triggers: one is called ***responsibility*** $\gamma(i, j)$, sent from t_i to a candidate exemplar t_j; the other is ***availability*** $\alpha(i, j)$, sent from the candidate exemplar t_j to t_i.

The calculation of each procedure iterates until convergence. To begin with, the availabilities are initialized to zero: $\alpha(i, j) = 0$. Then the responsibilities are computed using the following rule:

$$\gamma(i, j) \leftarrow s(i, j) - \max_{j' s.t. j' \neq j} \{\alpha(i, j') + s(i, j')\} \quad (3)$$

where the similarity score $s(i, j)$ indicates how well t_j is suited to be the exemplar for t_i. Whereas the above responsibility update lets all candidate exemplars compete for the ownership of a trigger candidate t_i, the following availability update gathers evidence from trigger candidates as to whether each candidate exemplar would make a good exemplar:

$$\alpha(i, j) \leftarrow \min \Big\{0, \gamma(j, j) + \sum_{i' s.t. i' \notin \{i, j\}} \max\{0, \gamma(i', j)\}\Big\} \quad (4)$$

Given T, we can generate an $n \times n$ affinity matrix $\boldsymbol{M}$ which serves as the input of the affinity propagation. M_{ij} represents the negative squared difference in relative importance score between t_i and t_j (Equation 5).

$$M_{ij} = -(\mathcal{I}(i \mid \{Q, F\}) - \mathcal{I}(j \mid \{Q, F\}))^2 \quad (5)$$

We compute the average importance score for all the clusters after convergence and keep the one with the highest average score as the trigger set. For example, given the query and slot filler pair in Figure 3, we obtain trigger candidates $T = \{died, divorced, from, in, in\}$ and their corresponding relative importance scores. After the above clustering, we obtain three clusters and choose the cluster $\{divorced\}$ with the highest average relative importance score (0.128) as the trigger set.

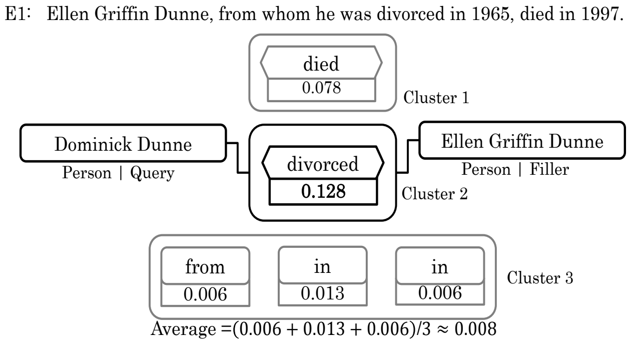

Figure 3: Trigger candidate filtering for E1.

3.3 Slot Type Labeling

To label the slot type for an identified relation tuple (*Query*, *Trigger*, *Filler*), the simplest solution is to match the trigger against existing trigger gazetteers for certain types of slots. For example, Figure 4 shows how we label the relation as a *spouse* slot type.

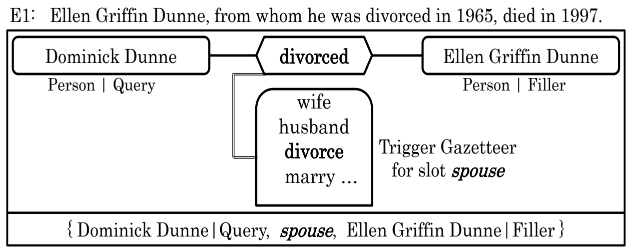
E1: Ellen Griffin Dunne, from whom he was divorced in 1965, died in 1997.
Dominick Dunne
Person | Query
divorced
Ellen Griffin Dunne
Person | Filler
wife
husband
divorce
marry ...
Trigger Gazetteer for slot spouse
{ Dominick Dunne | Query, spouse, Ellen Griffin Dunne | Filler }

Figure 4: Example of slot type labeling.

3.4 Experiment

In order to evaluate the quality of our proposed framework and its portability to a new language, we use TAC-KBP 2015 English Cold Start Slot Filling (CSSF) and TAC-KBP2015 Chinese Slot Filling (CSF) data sets for which we can compare with the ground truth and state-of-the-art results reported in previous work. The source collection includes news documents, web blogs and discussion forum posts. In ESF there are 50 person queries and on average 20 relevant documents per query; while in CSF there

Table 2: English Cold Start Slot Filling F_1 (%) (KBP2015 CSSF data set).

Slot Type	Our Approach	Angeli'15
siblings	**48.0**	26.1
other_family	0.0	**33.3**
spouse	14.3	**15.4**
children	**72.8**	0.0
parents	**25.0**	14.3
schools_attended	**63.6**	42.1
date_of_birth	0.0	**80.0**
date_of_death	**44.0**	0.0
state_of_birth	0.0	**33.3**
state_of_death	0.0	**15.4**
city_of_birth	0.0	**85.7**
city_of_death	0.0	0.0
country_of_birth	0.0	**66.7**
country_of_death	**100.0**	0.0
states_of_residence	0.0	0.0
cities_of_res.	0.0	**50.0**
countries_of_res.	0.0	0.0
employee_of	**60.0**	26.7
Overall	**39.2**	27.6

are 51 person queries, and on average 5 relevant documents per query.

We only test our method on 18 trigger-driven person slot types shown in Table 2. Some other slot types (*e.g.*, *age*, *origin*, *religion* and *title*) do not rely on lexical triggers in most cases; instead the query mention and the filler are usually adjacent or seperated by a comma. In addition, we do not deal with the two remaining trigger-driven person slot types (*i.e.*, *cause_of_death* and *charges*) since these slots often expect other types of concepts (*e.g.*, a disease or a crime phrase).

We apply Stanford CoreNLP (Manning et al., 2014a) for English part-of-speech (POS) tagging, name tagging, time expression extraction, dependency parsing and coreference resolution. Based on the released evaluation queries from KBP2015 Cold Start Slot Filling, our approach achieves 39.2% overall F-score on 18 person trigger-driven slot types, which is significantly better than state-of-the-art (Angeli et al., 2015) on the same set of news documents (Table 2).

Compared to the previous work, our method discards a trigger-driven relation tuple if it is not supported by triggers. For example, "*Poland*" is mistakenly extracted as the country of residence of "*Mandelbrot*" by distant supervision (Roth et al., 2013) from the following sentence:

*A professor emeritus at Yale University, **Mandelbrot** was born in **Poland** but as a child moved with his family to **France** where he was educated.*

maybe because the relation tuple (*Mandelbrot*, *live_in*, *Poland*) indeed exists in external knowledge bases. Given the same entity pair, our method identifies "*born*" as the trigger word and labels the slot type as *country_of_birth*.

When there are several triggers indicating different slot types in a sentence, our approach performs better in associating each trigger with the filler it dominates by analyzing the whole dependency tree. For example, given a sentence:

***Haig** is survived by his wife of 60 years, Patricia; his children Alexander, Brian and **Barbara**; eight grandchildren; and his brother, the Rev. Francis R. Haig.*

(*Haig*, *sibling*, *Barbara*) is the only relation tuple extracted from the above sentence by the previous method. Given the entity pair (*Haig*, *Barbara*), the relative importance score of "*children*" (0.1) is higher than the score of "*brother*" (0.003), and "*children*" is kept as the only trigger candidate after clustering. Therefore, we extract the tuple (*Haig*, *children*, *Barbara*) instead. In addition, we successfully identify the missing fillers for other slot types: *spouse* (*Patricia*), *children* (*Alexander, Brian and Barbara*) and *siblings* (*Francis R. Haig*) by identifying their corresponding triggers.

In addition, flat relation representations fail to extract the correct relation (*i.e.*, *alternate_names*) between "*Dandy Don*" and "*Meredith*" since "*brother*" is close to both of them in the following sentence:

*In high school and at Southern Methodist University, where, already known as **Dandy Don** (a nickname bestowed on him by his brother) , **Meredith** became an all-American.*

4 Nugget Detection

4.1 Event Nugget Detection

Event detection remains a challenge due to the difficulty at encoding word semantics and word senses in various contexts (Huang et al., 2016). Previous approaches heavily depend on language-specific knowledge and existing natural language processing (NLP) tools. However, compared to English, not all languages have such resources and tools available. A more promising approach is to automatically learn effective features from data, without relying on language-specific resources.

In this year's evaluation, we applied a language independent neural network architecture: Bi-LSTM-CRF, which can significantly capture meaningful sequential information and jointly model nugget type decisions, for multilingual (English, Spanish, Chinese) event nugget detection. For realis prediction, we incorporate both nugget-level and sentence-level information into Convolutional Neural Networks.

We first describe a Bidirectional LSTM model for event detection. Bi-LSTM (Huang et al., 2015; Lample et al., 2016; Feng et al., 2016) is a type of bidirectional recurrent neural networks (RNN), which can simultaneously model word representation with its preceding and following information. Word representations can be naturally considered as features to detect triggers and their event types. As show in (Chen et al., 2015), we take all the words of the whole sentence as the input and each token is transformed by looking up word embeddings. Specifically, we use the Skip-Gram model to pre-train the word embeddings to represent each word (Mikolov et al., 2013; Bahdanau et al., 2014).

We present the details of Bi-LSTM-CRF for event nugget detection in Figure 5.

We can see that Bi-LSTM is composed of two LSTM neural networks, a forward LSTM_F to model the preceding contexts, and a backward LSTM_B to model the following contexts respectively. The input of LSTM_F is the preceding contexts along with the word as trigger candidate, and the input of LSTM_B is the following contexts plus the word as trigger candidate. We run LSTM_F from the beginning to the end of a sentence, and run LSTM_B from the end to the beginning of a sentence. Afterwards, we concatenate the output F_v of LSTM_F and B_v of LSTM_B as the output of Bi-LSTM. One could also try averaging or summing the last hidden vectors of LSTM_F and LSTM_B as alternatives.

CRF(Lafferty et al., 2001) is widely used in many tagging tasks. It can incorporate the conditional observations of tags and predict the tag distributions

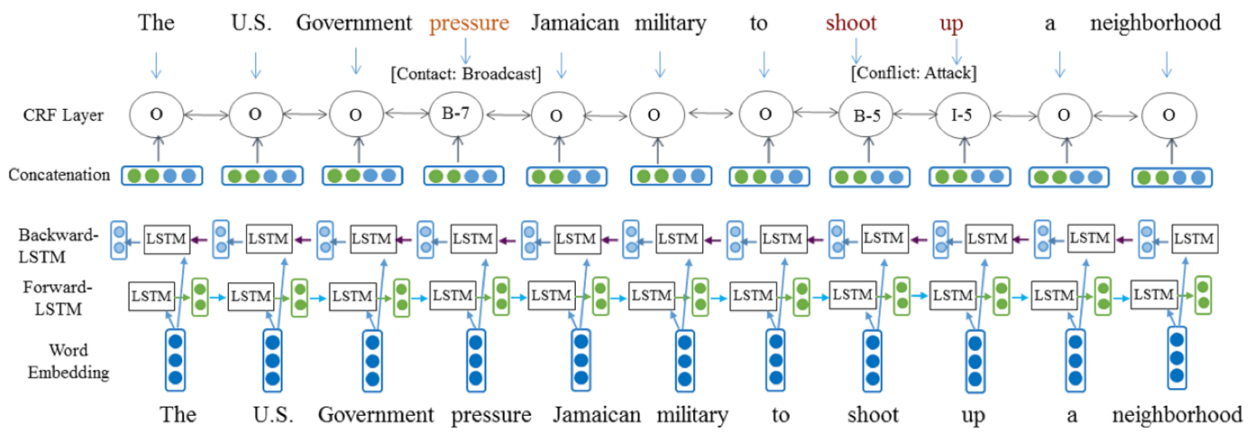

Figure 5: An illustration of our model for event nugget detection (here the nugget candidates are "*pressure*" and "*shoot up*").

on sentence level. For an input sequence $X = (X_1, X_2, ..., X_n)$ and their corresponding tags $Y = (Y_1, Y_2, ..., Y_n)$, n is the number of units contained in the sequence, we feed the output of each input unit from Bi-LSTM as features to CRF and maximize the log-probabilities of all tag predictions of the sequence.

4.2 Realis Prediction

For realis prediction, previous methods usually rely on hand-crafted features and dictionaries (Hong et al., 2015). Considering the limited resources for other languages, we apply a Convolutional Nueral Networks (CNNs) (Krizhevsky et al., 2012), incorporating both event nugget and its context information to predict the realis type.

The general architecture of the CNNs is shown in Figure 6. For each event nugget candidate, we apply a standard CNNs framework on both the left and right context of the nugget candidate and concatenate the max-pooling output of both the two CNNs as well as the representation of nugget candidate as input to a fully connected MLP layer. Finally we use Softmax function to predict the realis type.

5 Event Nugget Coreference

5.1 Approach

The algorithm of our Event Nugget Coreference system is based on our last year's system (Hong et al., 2015; Chen and Ji, 2009). We view the event nugget coreference space as an undirected weighted graph in which the nodes represent all the event nuggets and the edge weights indicate the coreference confidence between two event nuggets. And we applied the hierarchical clustering to classify the event nuggets into event hoppers.

We train a maximum entropy based classifier to generate the confidence matrix W. Each confidence value indicates the probability that there exists a coreference link C between two event nuggets em_i and em_j.

$$P(C|em_i, em_j) = \frac{e^{(\Sigma_k \lambda_k g_k(em_i, em_j, C))}}{Z(em_i, em_j)}$$

where $g_k(em_i, em_j, C)$ is a feature and λ_k is its weight; $Z(em_i, em_j)$ is the normalizing factor. The feature sets used during training are listed in Table 3. Some of the features such as Wordnet based similarities and POS tags are used for English only, because of the limitation of the resources for Chinese and Spanish.

5.1.1 Clustering

Let $EN = \{en_n : 1 \leq n \leq N\}$ be N event nuggets for one event type in a document and $EH = \{eh_k : 1 \leq k \leq K\}$ be K event hoppers. Let $f : EN \rightarrow EH$ be the function mapping from an event nugget $en_n \in EN$ to an event hopper $eh_k \in EH$. Let $coref : EN \times EN \rightarrow [0, 1]$ be the function that computes the coreference confidence value between two event nuggets $en_i, en_j \in EN$.

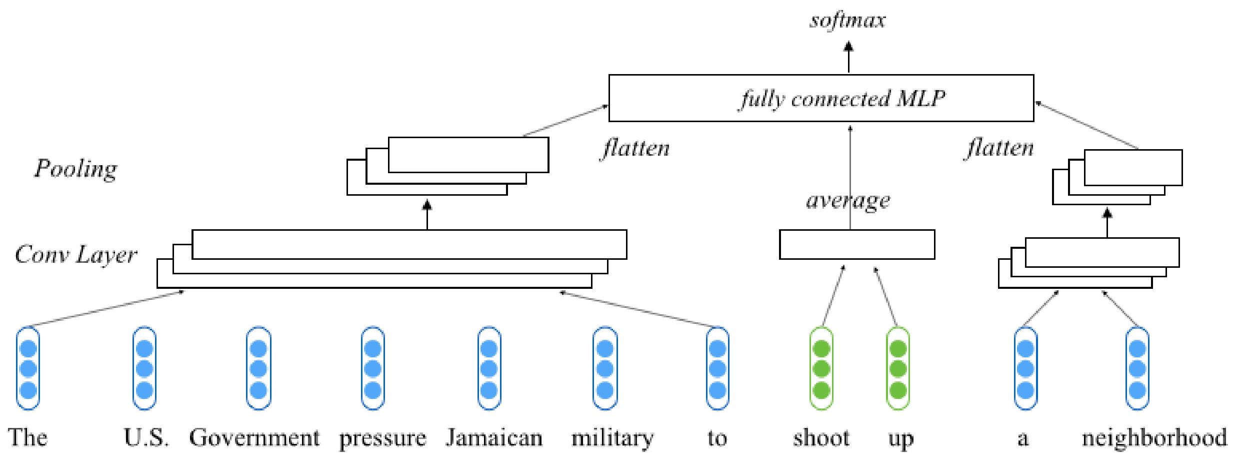

Figure 6: An illustration of CNNs Architecture for event nugget realis classification (here the nugget candidate is "*shoot up*").

Features	**Remarks(EM1: the first event mention, EM2: the second event mention)**
type_subtype_match	1 if the types and subtypes of the event nuggets match
trigger_pair_exact_match	1 if the spellings of triggers in EM1 and EM2 exactly match
stem_of_the_trigger_match†	1 if the stems of triggers in EM1 and EM2 match
similarity_of_the_triggers(wordnet)*	quantized semantic similarity score (0-5) using WordNet resource
similarity_of_the_triggers(word2vec)	quantized semantic similarity score (0-5) using word2vec embedding
POS_match*	1 if two sentences have the same ' NNP ' CD '
token_dist	how many tokens between triggers of EM1 and EM2 (quantized)
realis_conflict	1 if the realis in EM1 and EM2 exactly match
Entity_match	Number of entities appear both in sentences of EM1 and EM2
Entity_prior	Number of entities appear only in the sentence of EM1
Entity_act	Number of entities appear only in the sentence of EM2

Table 3: Featurs for Classifier. ($*$: For English only; $\dagger$: For English and Spanish only).

For each event type in the document, we construct a graph $G(V, E)$, where $V = \{en_n | f(en_n).en_n \in EN\}$ and $E = \{(en_i, en_j, coref(en_i, en_j)) | en_i, en_j \in EN\}$. We then apply a hierarchical clustering algorithm to the graph. Because the number of hoppers K, which has to be set in advance, is unknown, we define a parameter ϵ to quantify the performance of the clustering results by varying the number of hoppers K from 1 to N. N is the number of event nuggets. For each K, ϵ is the ratio of the number of conflicting edges(N_c) to the number of all edges(N_e). The smaller ϵ is, the better the result of clustering is.

$$\epsilon_K = \frac{N_e}{N_a}$$

An edge between two event nuggets is defined as conflicting in either of the following two cases, where δ is the confidence threshold:

1. $f(en_i) = f(en_j)$ but $coref(en_i, en_j) < \delta$
2. $f(en_i) \neq f(en_j)$ but $coref(en_i, en_j) > \delta$

Here's an example to demonstrate how to compute ϵ. In Figure 7, five event nuggets are classified into three event hoppers, and there are three conflicting edges($coref(en_1, en_5), coref(en_2, en_4), coref(en_3, en_4)$) according to our definition. Thus

$$\epsilon_3 = \frac{3}{10} = 0.3$$

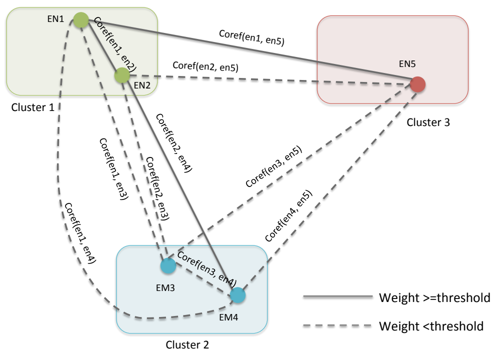

Figure 7: Clustering Example

5.2 Experiments

We used the ERE data provided by LDC to train three classifiers respectively. Table 4 shows the statistics for training and testing.

Language	**Training**		**Testing**	
	Newswire	Forum	Newswire	Forum
English	77	74	81	77
Spanish	154	30	15	5
Chinese	56	134	5	15

Table 4: Statistics of the data for training and testing.

And table 5 shows the F-scores based on four metrics: Muc, B^3, CEAFe and Blanc.

Language	English	Spanish	Chinese
Muc	51.03	62.28	61.86
B^3	81.45	51.91	55.63
CEAFe	75.34	54.31	51.48
Blanc	68.69	47.54	51.85

Table 5: Statistics of the data for training and testing.

We tuned the threshold, which is 0.54 for clustering to obtain the best average F-scores.

Acknowledgments

This work was supported by the U.S. DARPA DEFT Program No. FA8750-13-2-0041, ARL NS-CTA No. W911NF-09-2-0053, NSF CAREER Award IIS-1523198, and a gift award from Bosch. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

References

G. Angeli, V. Zhong, D. Chen, J. Bauer, A. Chang, V. Spitkovsky, and C. Manning. 2015. Bootstrapped self training for knowledge base population. In *Proc. Text Analysis Conference (TAC 2015)*.

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473*.

Zheng Chen and Heng Ji. 2009. Graph-based event coreference resolution. In *Proceedings of the 2009 Workshop on Graph-based Methods for Natural Language Processing*, pages 54–57. Association for Computational Linguistics.

Yubo Chen, Liheng Xu, Kang Liu, Daojian Zeng, and Jun Zhao. 2015. Event extraction via dynamic multi-pooling convolutional neural networks. In *Proceedings of ACL2015*.

Xiaocheng Feng, Lifu Huang, Duyu Tang, Bing Qin, Heng Ji, and Ting Liu. 2016. A language-independent neural network for event detection. In *The 54th Annual Meeting of the Association for Computational Linguistics*, page 66.

B. Frey and D. Dueck. 2007. Clustering by passing messages between data points. *science*.

J. Ganitkevitch, B. Van Durme, and C. Callison-Burch. 2013. PPDB: The paraphrase database. In *Proc. North American Chapter of the Association for Computational Linguistics - Human Language Technologies (NAACL-HLT 2013)*.

Yu Hong, Di Lu, Dian Yu, Xiaoman Pan, Xiaobin Wang, Yadong Chen, Lifu Huang, and Heng Ji. 2015. Rpi blender tac-kbp2015 system description. In *Proc. Text Analysis Conference (TAC2015)*.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Lifu Huang, Xiaocheng Feng Taylor Cassidy, Heng Ji, Clare R Voss, Jiawei Han, and Avirup Sil. 2016. Liberal event extraction and event schema induction. page 66.

G. Jeh and J. Widom. 2003. Scaling personalized web search. In *Proc. World Wide Web (WWW 2003)*.

H. Ji, R. Grishman, D. Freitag, M. Blume, J. Wang, S. Khadivi, R. Zens, and H. Ney. 2009. Name extraction and translation for distillation. *Handbook of Natural Language Processing and Machine Translation: DARPA Global Autonomous Language Exploitation*.

Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. 2012. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the eighteenth international conference on machine learning, ICML*, volume 1, pages 282–289.

Guillaume Lample, Miguel Ballesteros, Sandeep Subramanian, Kazuya Kawakami, and Chris Dyer. 2016. Neural architectures for named entity recognition. *arXiv preprint arXiv:1603.01360*.

Di Lu, Xiaoman Pan, Nima Pourdamghani, Shih-Fu Chang, Heng Ji, and Kevin Knight. 2016. A multi-media approach to cross-lingual entity knowledge transfer. In *Proc. the 54th Annual Meeting of the Association for Computational Linguistics (ACL2016)*.

C. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. Bethard, and D. McClosky. 2014a. The Stanford CoreNLP natural language processing toolkit. In *Proc. Association for Computational Linguistics (ACL 2014)*.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014b. The stanford corenlp natural language processing toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

R. Motwani and P. Raghavan. 1996. Randomized algorithms. *ACM Computing Surveys (CSUR)*.

Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational Linguistics*, 29(1):19–51.

L. Page, S. Brin, R. Motwani, and T. Winograd. 1999. The pagerank citation ranking: bringing order to the web.

B. Roth, T. Barth, M. Wiegand, M. Singh, and D. Klakow. 2013. Effective slot filling based on shallow distant supervision methods. In *Proc. Text Analysis Conference (TAC 2013)*.

Han Wang, Jin Guang Zheng, Xiaogang Ma, Peter Fox, and Heng Ji. 2015. Language and domain independent entity linking with quantified collective validation. In *Proc. Conference on Empirical Methods in Natural Language Processing (EMNLP2015)*.

S. White and P. Smyth. 2003. Algorithms for estimating relative importance in networks. In *Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM.

Jin Guang Zheng, Daniel Howsmon, Boliang Zhang, Juergen Hahn, Deborah McGuinness, James Hendler, and Heng Ji. 2014. Entity linking for biomedical literature. *BMC Medical Informatics and Decision Making*.