SoochowNLP System Description for 2016 KBP Slot Filling and Nugget Detection Tasks

Yu Hong[1] Yingying Qiu Zengzhuang Xu Wenxuan Zhou Jian Tang
Xiaobin Wang Liang Yao Jianmin Yao

School of Computer Science and Technology
Soochow University
No.1 Shizi ST Suzhou City Jiangsu Province China 215006

1 Abstract

We submitted a Code Start-Slot Filling (SF) system and a Nugget Detection (ND) system in this year's KBP evaluation conference. There is a brand new provenance retrieval and a filler filtering method used to implement the SF system. For the ND system, we employed a new propagation method.

2 Code Start Slot Filling

We built a social-network based filler verification method to enhance our basic filler discovery and extraction system.

2.1 Baseline System

The baseline is a pattern based search engine, which generate a query by using entity name and retrieves pseudo-relevant documents. For every entity mentions occurred in the documents, the system determines them as reliable candidate fillers when meeting a predefined relation pattern. The pattern requires that a candidate necessarily co-occur with at least one trigger word and the query in a text span, closely and frequently.

We use the following query as an example throughout the discussion in this section.

```
<query id="CSSF16_ENG_01e5f32687">
  <name>Viktor Yanukovych</name>
  <docid>NYT_ENG_20131213.0091</docid>
  <beg>2764</beg>
  <end>2780</end>
  <enttype>per</enttype>
  <slot0>per:employee_or_member_of</slot0>
  <slot1>org:top_members_employees</slot1>
</query>
```

We use the query name "Viktor Yanukovych" to strictly match with tokens in every document. A strict matching means that a token or a chain of sequentially-occurred tokens is the same with the query name. If a document contains tokens that do match the query, we regard the document as related to the target entity. In terms of the request of strict matching, for example, a document which only contains a partial name like "Viktor" is determined as irrelevant to "Viktor Yanukovych". Only the documents which contain an unbroken chain "Viktor Yanukovych" will be defined as available relevant documents. We conduct searching process over the eligible data resources (LDC2016E63[2]) which contain 30K documents.

1 Corresponding author (tianxianer@gmail.com)

2 TAC KBP 2016 Evaluation Source Corpus V1.1

In related documents, we further search candidate fillers for the particularly specified slot types of the target query, such as <slot 0> if available, the organization which employed Viktor Yanukovych, as well as <slot 1>, top members of the organization.

We use a rule based method to retrieve the candidates. A qualified candidate needs to meet the following rules:

- To be an entity mention whose type is the same with the slot type of <slot*>, e.g., an organization (org) entity mention can be used as a candidate of <slot0> of "Viktor Yanukovych".
- Co-occurs with the query name in a sentence. This sentence will be used as the provenance if the candidate is eventually determined as a filler.
- Co-occurs with a trigger word of the slot type of <slot*> in a sentence, such as the triggers like "*hire*", "*engage*", "*reassignment*" etc when we are considering the <slot0> of "*michael browns*".

In practice, we generate patterns using the rules, and search the sentences in which there is at least one entity mention (besides the target) that follows the patterns. For example, the case-specific patterns to the query name "Viktor Yanukovych" include (but not limited to):

< Viktor Yanukovych, hire, ORG>

< Viktor Yanukovych, engage, ORG>

< Viktor Yanukovych, reassignment, ORG>

We employed the entity recognition results of IBM entity linking system of version 2 on LDC2016E63, which include the mentions along with their types (PER, ORG, GPE, LOC and FAC). We use the types to verify their eligibility for those case-specific patterns.

We use Bird's NLTK[3] toolkit (Bird 2006) to segment sentences in the related documents.

By the patterns, we extract entity mentions from sentences in the related documents. We use the mentions as candidate fillers.

2.2 Trigger Lexicon

We build an English lexicon of triggers. A trigger refers to a lexical unit (word or phrase) that represents the relations held between entities, or state and action of an entity. For example, the words "*reassignment*", "*engage*" and "*hire*" imply the relation between an employee and an employer, triggering a filler of the slot type per: employee_or_member_of, similarly, the phrases "*giving life*" and "*give the birth*" represent an birth event, triggering the fillers of slot types per: date_of_birth and per: city/state/province/country_of_birth.

We build the lexicon for every slot type by two steps: seed trigger collection and lexicon expansion.

We collect the seeds from ground-truth triggers of slots in 2014 slot filling training data and test data. By using the seeds, we build a basic lexicon, where there is a one-to-many mapping schema between seeds and slot types, e.g., *born* corresponds to per_date/city/state/province/country_of_birth

In order to expand the base, we search concept-similar lexical units in FrameNet, and combine them with the seeds. FrameNet[4] (Baker et al 2003) is a frame-semantics based dictionary. It provides thousands of frames that define the common meanings or contexts of a cluster of lexical units. Table 1 shows the definition of a frame Giving_Birth along with lexical units attached with it.

We filter the ambiguous lexical units, such as "*drop*", "*lay*", "*have*" and "*get*" in Table 1.

3 http://www.nltk.org/

4 http://framenet.icsi.berkeley.edu/

The units generally bring large-scale noises into the trigger-slot mapping schema. Table 2 shows a partial type-trigger schema

Giving_Birth (Def.) A Mother and Father produce a Child or an Egg
Lexical units *bear (v) beget (v) birth (v&n) bring forth (v) calve (v) calving (n) carry to term (v) drop (v) father (v) mother (v) generate (v) kid (v) lambing (v) propagate (v) sire (v) spawn (v) whelp (v) lay (v) have (v)*

Table 1: Example of frame and lexical units

2.3 Social Network Based Filler Filtering

Over the candidate fillers, we carry out a filtering process in terms of the filler verification results. We go to verify the eligibility of a filler by assessing relationship between the filler and the target entity's social network.

It should be thoughtfully believable that a filler most probably presents an aspect of the entity if it falls into the social network. The aspect may be the attribute of the entity or relation with other entities. Considering that most of the slot types of concern in KBP are awaited to be filled into by entity mentions (fillers of entity), e.g., parents, spouses, siblings, school, employers, members, etc. Social network is a reliable reference to verify the eligibility of a filler, i.e., a filler who plays a role of attribute of the target entity or has a relation with the entity.

In this year's evaluation, we only consider the fillers of entity in the process of social network based filler verification. For the other types of fillers, such as cause of death, we directly adopt the candidates as the final results without verification. Besides, we ignore extracting fillers of time expression due to provably worse performance of expression recognition..

per:children	*child，babies, children, daughter, father*
per:employee_or_member_of	*worker, staff, hire, employer, hired*
per:place_of_death	*die, died, pass away, die*
org:founded_by	*founded, set up, start-up, organize, formed*
org:members	*Involve, join, combine, league, armed wing*
org:organizations_founded	*form, set up, organize, start-up, founder*
org:parents	*central office, head office, central-office, branch, sub-company*
gpe:employees_or_members	*Employee, hire, staff, hireling, wage worker*
gpe:headquarters_in_place	*Headquarter, head office, based, built, central-office*
gpe:residents_of_place	*grew-up, grew up, resident, live, move back*

Table 2: Partial type-trigger mapping table

2.4 Social Network Building

We simply use a set of name entities found in the reference document and closely related documents to build the social network.

The reference document is the one which contains the provenance of the query. It can be obtained through the document ID in the query block, such as that of "Viktor Yanukovych", indicated by <docid> NYT_ENG_20131213.0091 </docid>. The network in the reference document is named as a RSO.

Besides, we use the reference document to generate a keyword-based topic description, and search related documents from the corpus of LDC2016E63. By this search engine, we retrieve a set of pseudo relevant documents as a larger background knowledge base. From this base, we extract entity mentions to build a more complete social network. We name this network as ASO.

It is noteworthy that the quality of the retrieved documents is crucial for collection of truly related social network to the target entity. The available document-level corpus, LDC2016E63, have smaller-scale data available for IR. In our top-n (n=50) pseudo relevant documents for each query, there should be lots of noises. They will definitely mislead the network building process and further filler verification. Besides, there are lack of related documents in the search results. This reduce the scale of social network. As we will show below, such an incomplete network may reduce the probability to successfully detect qualified fillers.

2.5 Filler Verification

We attempt to extract fillers of all types of slots for a target entity by the pattern based extraction system. This has been conducted generally in previous KBP slot filling evaluation. Though it is widely known that this year's case-specific slot filling task do not request a system to return that complete results, but the ones limited to one (<slot0>) or two (<slot0 &1>) particular types, such as the following ones of "Viktor Yanukovych".

The goal of that we extract fillers of all types is to detect candidate social networks (CSOs) of the target entity. We argue that the extracted fillers for all available slot types contain at least one CSO. We only concern the entity mentions in the CSO. They are most probably related to each other, such as parents, clauses, siblings. On the basis, we can verify the eligibility of a candidate filler by following rules:

- The CSO is truly related to the target entity but not the ones which have the same name.
- The candidate occurs in the CSO.

Empirically, we can obtain more than one CSO for each target entity due to two reasons:

- IR system aims to recall all documents that contain the query name.
- Query name is ambiguous, the retrieved documents that contain the name are probably related to different entities.
- The fillers of all types we extracted from the retrieved documents involve multiple social networks of different entities with the same name.

We obtain CSOs by clustering the fillers of all types in the retrieved documents. A hierarchical clustering method (Dasgupta et al 2002) is employed. The similarity between pairwise clusters is calculated by mutual information.

CSO is different from RSO and ASO. CSO is obtained from fillers of all types in all documents that contain the target entity name. RSO is the set of potentially related entity mentions in the reference document. ASO is that in pseudo relevant documents to the reference document. For a target entity, we generally obtain multiple CSOs but only one RSO and ASO. The RSO is used as an incomplete but reliable social network, while the ASO a complete but noisy social network.

We determine whether a CSO is eligible by similarity between the CSO and the RSO and ASO respectively. The similarity is simply calculated by the numbers of entity mentions two social networks commonly contains.

For a query, we first mine candidate fillers for the case-specific slot types, second we detect an eligible CSO in the fillers of all types, finally we inspect whether the candidates occur in the CSO, if yes, they will be determined as positive fillers, otherwise negative. We output positive ones as extraction results.

2.6 Experimental Results

We evaluate the verification method over fillers submitted by all participants of KBP-2013 slot filling task. In the pool, every filler

was previously labeled positive or negative. We use our verification method to detect and filler the negative ones. Experimental results show that the precision is improved from 0.33 to 0.39, while the recall is reduced from 1 to 0.98. Note that we assume that the recall rate is 1 in the pool. We only filter negative fillers out of the pool. This definitely reduce the recall. By contrast, a random filtering method reduce both precision and recall. See Table 3 which show all the performance of the filtering methods, along with the original by all participants.

	P	R	F
Original	0.21	1	0.34
Our best	0.24	0.85	0.38
Random_drop	0.21	0.87	0.33

Table 3: Verification performance over pooling results of participants of KBP-SF 2013

However, we achieve surprisingly worse performance in this year's slot filling evaluation. We find that the baseline system gives a lowest recall, no more than 0.05. This give a reason to believe that the proposed verification and filtering method cannot provide considerable positive effect on the extraction results. Besides the errors of entity recognition, off-set and provenance segmentation negatively influence the eventual performance.

3 Nugget Detection

We use a new propagation method to improve basic nugget classification system.

The propagation method propagates classification results within text spans that share the same topic. The foundation behind is that those spans may contain similar event mentions if they have. Thus the propagation process will introduce fewer noises into detection results.

3.1 Baseline System

We follow Yu et al (2015)'s nugget detection system, rebuilding a maximum entropy based nugget classification. All features in the original system are employed again. A minor difference lies in refinement of trigger words of events.

The trigger words were collected from existing training data, and semantic frames of the triggers were regarded as concept-level representations of event types. For example, the frame of the trigger bombing is Attack, it is defined as one of the concepts of KBP attack events. In practice, a word or phrase was translated into its frame. The frame would be used together with other features in event type classification. The use of frames enables discovery of event mentions that triggered by lexical units out of prior trigger list, only if they adhere to the same frames with those in the list.

However, we find that some prior trigger words are ambiguous in senses, unavoidably corresponding to multiple semantic frames. If employing all their frames as representations of events, we will recall many text spans that have nothing to do with the event types of concern.

We refine the trigger list to reduce the ambiguous fillers. All the triggers that respond to more than one frames were filtered. In the future we will change this method by gently adding confidence of frame recognition to feature space. The recognition process will take contexts into account but not just simply translating lexical units into frames based on a fixed mapping table and lexicon.

3.2 Allied Propagation

Yu et al's propagation method verify whether a lexical unit can play a role of reliable nugget in a particular text. If it does, all the mentions of the unit in the text will be specified as a nugget, and labeled with the same event type. If some of them have not yet been labeled (missed by the nugget classifier), the

method picks up the cases as supplements.

We don't limit our propagation to the reliable nuggets. On the contrary, we take semantically related nuggets into consideration. We name the cases as allied nuggets of the reliable ones. An allied nugget needs to meet the following requirements:

- It is a lexical unit, and it used to be a nugget, or say, either being labeled as a nugget in the training data or in some texts previously dealt with by the nugget classifier, or corresponding to the same frames with those known nuggets.
- It is closely related to the reliable nuggets. The relation can be well assessed by mutual information in a large-scale corpus. When calculating mutual information, we treat both nuggets as general lexical units. Every occurrence is taken into account but not just the cases of nuggets.

Given a text, we detect all the allied nuggets of those reliable ones. If they haven't yet been labeled by the classifier, we pick up them as supplements. In addition, we recognize other lexical units that have the same sense with the allied nuggets. On the basis, we add them to the supplements with the role of new allied nuggets. Similarly, we recognize the homogeneous allied nuggets by using semantic frames.

	plain			*mention*			*realis*			*mention +realis*		
evaMetric	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
Baseline	62.32	**42.20**	50.33	53.49	**36.23**	43.20	48.30	32.71	39.00	41.03	27.79	33.13
System1	57.89	**44.71**	50.45	49.63	**38.33**	43.25	43.87	33.88	38.23	37.25	28.77	32.46
System2	57.56	**45.24**	50.66	49.40	**38.83**	43.49	43.39	34.11	38.19	36.85	28.97	32.43
System3	58.11	**45.17**	50.83	49.92	**38.81**	43.67	43.84	34.08	38.35	37.26	28.97	32.59

Table 4: Performance of nugget detection system (***Micro*** Average)

	plain			*mention*			*realis*			*mention +realis*		
evaMetric	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1	Prec	Rec	F1
Baseline	59.47	**41.03**	48.56	50.43	**34.71**	41.12	45.30	31.78	37.36	38.27	26.68	31.44
System1	55.62	**43.39**	48.75	47.12	**36.67**	41.24	42.10	32.94	36.96	35.55	27.67	31.12
System2	55.29	**43.90**	48.94	46.95	**37.16**	41.48	41.55	33.06	36.82	35.12	27.77	31.02
System3	55.73	**43.85**	49.08	47.37	**37.15**	41.64	41.87	33.05	36.94	35.42	27.77	31.14

Table 5: Performance of nugget detection system (***Macro*** Average)

3.3 Propagation Constraint

In order to avoid unlimited propagation, we use two constraints respectively to filter the weakly-related allied nuggets:

- If an allied nugget has never co-occurred with the reliable in a text window, it will be filtered as a noise.
- If the allied nugget and the reliable one frequently co-occur in the text windows related to the same topics, we determine them as eligible and propagate both of them, otherwise filter the allied.

In the evaluation duration, we set a text window as a full sentence. We use LDA models to generate topics as well as probability distributions of words over them. Probabilistic model is used to measure the topic-level similarity between sentences.

3.4 Evaluation

We submitted 3 nugget detection systems, numbered from 1 to 3. System 1 is the maximum entropy classifier which using frames of the disambiguated triggers for featuring semantics of lexical units. System 2 use the new propagation method to enhance System 1. The propagation takes all allied nuggets into account. System 3 uses topic-level sentence similarity as constraint to refine the propagated allied nuggets by System2.

The improvements are minor (see Tables 4 and 5). Despite all this, it is respected to achieve promising performance in open-domain nugget detection and that in larger-scale set of documents. In the other aspect, however, it illustrates that the recall of reliable nuggets is very important for mining more allied nuggets. Besides, it should be also proved whether the length of documents and medias have some influences on discovery of allied nuggets. A long story may involve more diverse triggers of a type of event, thus if they are all allied to some outputted nuggets by System1, there could be more new nuggets found and propagated. Further a media like twitter may provide documents (threads) that are full of OOVs, idioms and international slangs. This may give a chance to show the ability of allied propagation.

4 Conclusion

We will add frame disambiguation in slot filling task. More important, we will improve IR technique to collect truly related documents, especially the ones that do contain a reliable provenance. Sentence-level embedding may be used for provenance verification.

Similarly, we will use sentence-level embedding for event mention recognition. The trigger words are not enough to cover all language phenomena of event mentions.

For example, the sentence "*Because of her, he eventually owns a happy family*" contains a marrying event though doesn't an exact trigger word. It also gives the filler of slot spouse implicitly. The traditional lexicon based and word embedding based methods will encounter considerable difficulty in treating with such cases.

Acknowledgement

This research is supported by the National NSFC projects No.61672368, 61373097, and 61672367. The authors would like to thank the anonymous reviewers for their insightful comments and suggestions. 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 CHN Government. The CHN Government are authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

Reference

Steven Bird. 2006. NLTK: the natural language toolkit. In Proceedings of the COLING/ACL on Interactive presentation sessions, COLING-ACL'06, pages 69–72. Association for Computational Linguistics.

Collin F. Baker, Charles J. Fillmore, and John B. Cronin, 2003. The Structure of the Framenet Database, International Journal of Lexicography, Volume 16.3, 281-296.

Sanjoy Dasgupta and Philip M. Long. 2002. Performance guarantees for hierarchical clustering. Journal of Computer and System Sciences, 70(4), 555-569

Yu Hong, Di Lu, Dian Yu, Xiaoman Pan, Xiaobin Wang, Yadong Chen, Lifu Huang, Heng Ji. 2015. RPI BLENDER TAC-KBP2015 System Description. Proc. Text Analysis Conference (TAC2015), Gaithersburg, Maryland USA..