The Open Knowledge System for TAC KBP 2017

Zixuan Li, Yunqi Qiu, Fan Yang, Xiaolong Jin, Yuanzhuo Wang, Yantao Jia, Haoran Yan, Kailin Zhao, and Jialin Su

CAS Key Laboratory of Network Data Science and Technology,
Institute of Computing Technology,
Chinese Academy of Science
School of Computer and Control Engineering,
University of Chinese Academy of Sciences
lizixuan@software.ict.ac.cn

Abstract

This paper presents the Open Knowledge System (OKS) developed for the Cold Start KB task in TAC KBP 2017. In order to complete this task, we developed seven modules, namely, the English entity discovery module, the relation extraction module, the event nugget detection module, the entity linking and clustering module, the standalone BeSt module, inference module and post processing module. Particularly, the relation extraction module combines three existing methods: RNN-based relation extraction, OpenIE and Implicit Relation Extraction.

1 Introduction

The goal of TAC KBP 2017 is to develop and evaluate technologies for building and populating knowledge bases (KBs) from unstructured text. It contains several tracks, and we participated in the Cold Start Track (KB variant, English) this year, which contains five components: Entity Discovery and Linking (EDL), Slot Filling (SL), Event Nugget Detection and Coreference (EN), Event Argument Extraction and Linking (EAL), and Sentiment.

The Cold Start KBP track builds a knowledge base from scratch using a given document collection and a predefined schema for the entities and relations that will compose the KB. The KB schema for Cold Start 2017 consists of:

- Entities: entities and entity mentions as defined in the main task of the EDL track;
- SF Relations: entity attributes ("slots") as defined in the SF track;
- Events: events (hoppers) and event nuggets as defined in the EN track;
- Event Arguments: event arguments as defined in the EAL track;
- Sentiment: Sentiment from a source entity toward a target entity as defined in the Belief and Sentiment (Best) track.

For this purpose, we proposed a system consisting of seven modules to finish this task.

The paper is organized as follows. Section 2 describes the architecture of the developed system. Document-processing and entity discovery are explained in Section 3, including Pre-processing, entity discovery and intra-document coreference resolution. Section 4 describes the details of event nugget extractor. Section 5 presents the three methods and the strategy for combining them together. The sentiment module is introduced in Section 6, the entity linking and clustering module are explained in Section 7 as well as the inference step are explained in Section 8 and the post-processing module in Section 9. Finally, we conclude the paper in Section 10 and present related references.

2 The System Architecture

Our proposed system starts with the entity discovery module which extracts all named

mentions and nominal mentions from the corpus and saves their types and offsets, as well as carries out intra-document coreference resolution. Then the CNN-based event nugget detection module is used to discover event nuggets, and the relations between entities are extracted from the corpus (i.e. slot filling) in three ways: OpenIE-based method, RNN-based relation extraction and implicit relation extraction. After this step, the sentiment from a source to a target entity is extracted using an SGD classifier. Next, the sentiment module is employed to extract sentiments. After that, the task gets into the entity linking and clustering module. It is implemented by linking entities to Wikipedia and nil clustering. Finally, we utilized the post-processing component to remove wrong results and format final results. In the following sections, each step will be described in detail.

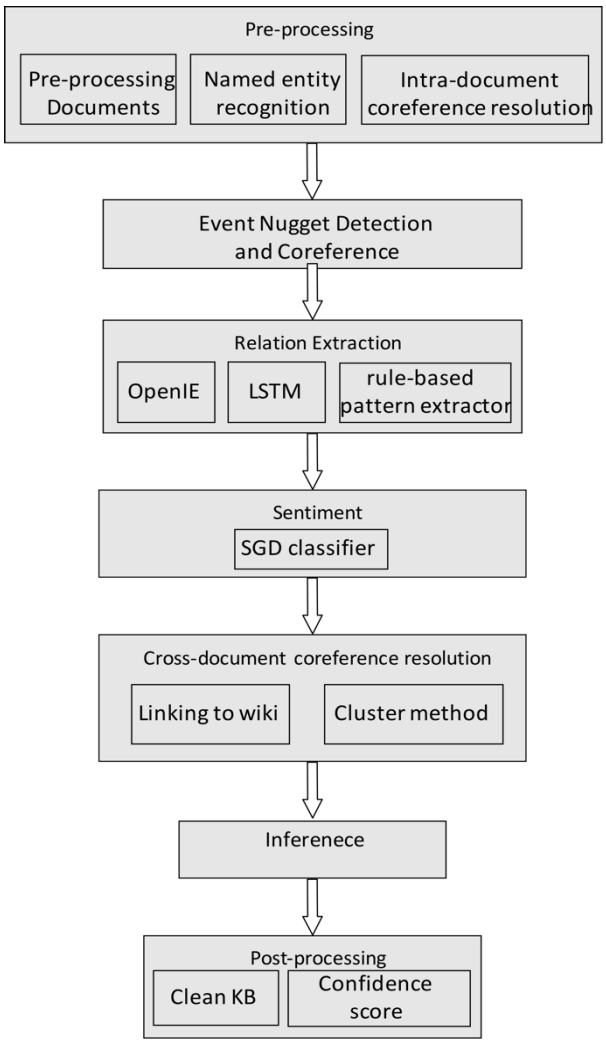

Figure 1 The system architecture.

3 The Entity Discovery Module

Entity discovery module consists of the pre-process of the corpus, entity discovery and intra-document coreference resolution as following.

3.1 Pre-processing

The documents are pre-processed in three aspects:

1) It should be noted that according to the Entity Linking Query Development Guidelines, when we detect discussion forum threads or web documents, the entity mentions occurring between <quote></quote> tags are ignored. As a result, we simply replaced these pieces of content by whitespaces to avoid the errors of offsets.
2) As the publish dates of documents are helpful to infer the slots concerning date (in most cases is death date, see details in section 2.4), we extracted the date information of each document. For newswires, the date is article's publish date; For forums, the date is post's date.
3) The length of each document is validated according to the given doc-length file.

3.2 Entity Discovery

We applied our precious work (Manling Li et al., 2016) to the entity discovery module. There are three steps to detect all entity mentions from the target document. First, we used Stanford NER (Finkel et al.,2005) to extract person (PER), organization (ORG), geo-political entity (GEP), facility (FAC) and location (LOC). Second, we used data provided by the reference KB (LDC2014T16 and LDC2015E42) to supplement the results of NER. Finally, we used the regular expression to extract post authors as persons. We combined these results as the final entity mentions. To detect nominal entity mentions, we trained a model by the Stanford NER tool as well, and the data were same as those used to train the named entity recognizer, but only extracting the NOM mentions which have <head> information.

3.3 Intra-document co-reference resolution ery

There are five steps in our intra-document co-reference resolution: a) the mentions whose name are the same are linked together; b) a co-reference

chain is generated by Stanford CoreNLP (Manning et al., 2014); c) we combined the two chains as the final co-reference chain; d) the entity types of mentions in one chain should be unified into the entity type same as the majority; e) the canonical mention is selected by following standards:

1) the canonical mention should appear in the main body of the document, i.e. in the content between <text> tags, to prevent from lack of contextual information.
2) the start offset of the canonical mention should be as small as possible.

4 The Event Nugget Detection and Co-reference Module

The event nugget detection and coreference module includes three parts, i.e., Event Detection and Classification, Realis Classification and Event Coreference Resolution. All models below are trained on the LDC2015 DEFT Rich ERE English Training Annotation_R2_V2 dataset.

4.1 Event Detection and Classification

Our event extractor employed rich features based on lexical and semantic resources. The features we used are list below.

1) word embedding in the context of trigger mention.
2) lemmas of the words in the context of trigger mention.
3) whether the word is a number in the context of trigger mention.
4) whether the word is upper or lower in the context of trigger mention.

We developed a CNN-based classification to judge whether a word is a trigger or not. We used P-O-S tag for the word we predicted. As for the type of event, we developed a logistic regression classification.

4.2 Realis Classification

We reused the CNN-base classification in section4.1 to predict the Realis value for the event nuggets (ACTUAL,GENERIC,OTHER). Except for the features above, we added some features such as the tense of the word in the context of the word.

4.3 Event Coreference Resolution

We employed a simple heuristic method for event coreference resolution. We just merged the event nuggets with the same trigger mention in a document.

5 The Relation Extraction Module

The relation extraction step consists of three methods, namely, the RNN-based relation extraction, the OpenIE-based relation extraction and Implicit Relation Extraction.

5.1 RNN-based Relation Extraction

Our neural network based extractor is a Bidirectional GRU model with both word level and sentence level attention. The model takes the original sentence as input, and generates word embedding vectors for each word. Afterwards the embedding vector sequence are fed into a bi-GRU module. We used tanh as the nonlinearity function. Then output representations from the bi-GRU component are fed into word-level attention layer and sentence-level attention layer to generate the final sentence representation, which is then passed into the softmax layer for the final prediction.

5.2 OpenIE-based Relation Extraction

This year we still use the Open IE relation extraction tool (we used in the last year of Open IE V4), the relation extraction tool proved to be effective in extracting relations quite completely. Further more, we processed the extracted results according to the initial output to get the relation tuples needed.

Open Information Extraction (Open IE) system is from the University of Washington (UW). An Open IE system runs over sentences and creates extractions that represent relations in text, and it produces tuples of the form (arg1, relation, arg2) for the given sentence. Because Open IE is the successor to Ollie. Whereas Ollie used bootstrapped dependency parse paths to extract relation. If we can deal with the relations returned successfully to obtain tuples we need, then we can get a good recall. We have to deal with them because there are conflicts between the direct results and the TAC demands. First let's see an example, if we consider this sentence: "The U.S. president Barack Obama gave his speech on

Tuesday to thousands of people." we may get several tuples: (Barack Obama, is the president of, the U.S.) and (Barack Obama, gave, his speech) and (Barack Obama, gave his speech, on Tuesday) and (Barack Obama, gave his speech, to thousands of people). This is not the result listed by the TAC, so we need to handle this. The output is expressed as a triple (A, B, C) where A and B are arguments, C is the relation between those arguments, and Open IE is not aligned with an ontology, the relation is a phrase of text. Next, we will accept it only if A and C are extracted in our Entity Discovery module. If it is demonstrative pro-noun, we also need to find the real argument and judge. Beside, we need to assign text B to the 41 exact relation if any relation be assigned to, then we will accept this result, otherwise abandon.

This year, we initially got 1230864 extraction results. After a series of screening, we retained 16792 results. Despite this low proportion, we still felt that Open IE is an effective tool.

5.3 Implicit Relation Extraction

Another relation information extractor we adopted is Implicit Relation Information Extractor, which had been proved in last year. The corresponding tool we adopted is IMPLIE (Soderland et al, 2015), rom which, we get the relation not expressed as text, but noun phrases or adjective phrases.

Compared with Open IE, we could not get all the possible tuples from the sentence, actually, just get limit relations. Next these phrases in the tuples will be aligned to the 41 relations by rules. For example, "jobTitle" will be transformed into "per:title", and it's accuracy is pretty high.

Eventually, we only obtain a part of the 41 relations we need, but the accuracy of some relation is much higher than Open IE. Another advantage is that some tuples related to location can't be extracted effectively by Open IE while IMPLIE could. Therefore, we think IMPLIE is a good supplement.

5.4 The Combination Strategy

The results of four methods are combined by simply taking the union. If these systems had different outputs for a functional relation, we calculated the confidence score by a linear-weighted method.

To be more precise, each method has different weight designed by the performance conducted on evaluation data of previous years, and the method with better performance has higher weight. For each triple (i.e. entity1, slot, entity2) we extracted, the confidence score is the normalization result based on the sum of the weights of the methods related to the triple.

For slots that admit only a single value (e.g., country_of_birth), we selected the triple with highest confidence. For slots that can have more than one value (e.g, per:parents), we selected the top 10 triples as the best set of values for the slot to avoid noises.

6 The Sentiment Module

We use a SGD classifier and the model input is word embeddings. First, we generate word embeddings with the help of the supervised IMDB data. This dataset consists of both positive and negative comments. Then we put the embeddings of each sentence into a SGD classifier for the final prediction. Specially, we make an assumption that source entities are the authors of comments in the Forums dataset and target entities are those appearing in comments. For the Newswire dataset, our method is same as that for relation extraction. Also we tried Logistic Regression classifier and SVC to get the final prediction.

7 The Entity Linking and Clustering Module

To address the tagging of entities, the system employs two steps to cluster the cross-document entities across target documents. Firstly, it employs entity linker to link entities to Wikipedia. Then it employs the single pass clustering method to cluster the entities in terms of the similarity between entity mentions.

7.1 Query Expansion

In the first step, we use acronym expansion matching in the document text. The rules we use for the query expansion are list as following:

1) Search the context of the entity. Add the mention into query list if the mention contains the whole name of the entity.
2) Search the Wikipedia to obtain alias name or acronym of an entity. Add it to the query list of the entity list.

3) Search the Wikipedia to obtain alias name or acronym of a entity. Add it to the query list of the entity list.
4) Calculate the string similarity between the entity and the entities in its context. Add the entities to the query list if the similarity exceed the threshold value.

7.2 Candidate Entity Generation

In order to reduce the time complexity of the linking process, a small set of candidate entities that may link to an entity mention detected from target documents should be generated in an appropriate manner. Namely, we regard every string in query list which is generated in Section 7.1 as a query to obtain the candidate set from the reference KB (i.e., Wikipedia).

7.3 Entity Linking

It takes two steps to generate the linking results. Firstly, coreference resolution is used to cluster the entity mentions that are referred as the same entity. Secondly, based on the cluster results, the system employs seven features to measure the similarity between the reference entity and an entity mention. All of the similarities are projected into dense vectors. These features are listed as follows:

1) Embedding similarity. The similarity between a mention embedding and a candidate mention embedding. Use the average of all word in a mention if the mention is a compound.
2) Name similarity. Namely, the string similarity between the entity mention in the document text and the candidate entity in the reference KB.
3) Context similarity. We select K words window surrounding an entity mention as its context, and compute the similarity between the entity mention and the candidate entity in the reference KB using TFIDF similarity.
4) Wikipedia redirect page with identical titles. Entity mention in the document matches the candidate entity with page referred by Wikipedia redirect page with identical titles.
5) Acronym matching, which indicates whether the entity mention is an acronym of the candidate entity and whether the candidate entity appears in the document text.
6) Candidate popularity. A popularity of the entity is represented by the number of links the entity has in wiki.
7) Type. The type of the entity mention and the type of the entity.

For each pair of entity mention and candidate. We calculate the features above, which are fed into a feed forward neural network with one hidden layer, the output of the network measures the similarity of the pair. The model is trained using the EDL2016 training data.

7.4 NIL Entity Clustering

For the NIL entities, three steps are used to cluster the NIL entities across target documents. Firstly, the similarities between entities are measured in terms of embedding similarity, context similarity, name similarity and type indicator between entity mentions, where the embedding similarity, the context similarity and type indicator are defined in the same way as above.

All nil entities are clustered using the single pass method. The method starts with a entity in a cluster by itself. Then compares the similarity of the next entity to centroids. A pre-specified similarity threshold is needed to judge whether add an entity to a cluster or create a new cluster by itself.

For the aim of avoiding too many compares and high time complexity, we improved the cluster method using elastic search. For a given entity, we first generate a candidate set using elastic search. Instead of comparing query with each cluster，we simplify the cluster by using the intersection of the candidate set and the original entity set of the cluster.

8 The Inference Module

The inference module is aimed to infer more triples based on the generated ones in Section 4, and it is conducted by mainly following these rules:

1) Rules for place-related slots. For example, for an entity that has value about slot “city”, we can infer corresponding "stateorprovince" and "country" by Gazetteer. Similarly, "country" can be inferred from "stateorprovince".
2) Rules for date-related slots. For example, for per:date_of_birth, per:date_of_death, per:age, given two of these three slots, the third one can

be inferred (except birth && age -> death, because someone who has birthdate and age may not die yet). For example, if A died in 2010 at age 78, so we can infer that A was born in 1932.

3) Rules for family relationships, which is illustrated in Table 2.

Table 2 Rules for inferring family relationships

A --- B	B --- C	A --- C
children	siblings	children
children	spouse	other_family
children	children	other_family
spouse	children	children
spouse	parents	other_family
spouse	siblings	other_family
parents	parents	other_family
parents	siblings	other_family
parents	spouse	parents

4) Rules for implicit-date results. For results for the slots describing date which doesn't express year/month/day explicitly, such as "died in Tuesday", we transforme it into standard date format according to the calendar.

5) Rules for employee-related slot. For example, for a person entity whose title is CEO, president, vice-president, or other titles which represent top employees, and this person entity has slot "per:employee_or_member_of", we can infer slot "org:top_members_employees".

6) Rules for inverse slot. For every slot which has inverse slot, we add the inverse relation of this slot according to the Slot Description Guideline.

9 The Post-processing Module

To correct the errors in the slots extracted by the Filler component, we introduce the post-processing step. Specifically, we mainly use some rules, which are listed as below.

1) The values of some slots must be of certain particular type. For example, when slots describe relations between people (e.g. spouse, children), the type of the results must be person (PER). This can be examined by means of the Stanford NER tool (Finkel et al., 2005).

2) Standardization of dates. Convert all answers which represent dates to standard date format "XXXX-XX-XX".

3) Delete unreasonable answers. For example, the results for the slots describing age should be a number usually larger than 0 and smaller than 130 respectively.

10 Result Evaluation

This year, each cold start KB undergoes a composite KB evaluation and a set of component KB evaluations. The results of component KB evaluations for our system are presented in Tables 2, 3, 4 and 5.

For the Cold Start KB Task, we submitted four runs, in which the third run performs best in Entity discovery and linking and the first run performs best in Event Nugget Detection.

Tables 2 and 3 are the results of Entity Discovery and Slot Filling dimension respectively.

Table 1 Entity discovery results (English)

	strong_mention_match			strong_typed_metion_match			Type_mention_ceaf		
	P	R	F1	P	R	F1	P	R	F1
1	0.909	0.622	0.738	0.817	0.559	0.664	0.567	0.388	0.461
2	0.913	0.619	0.738	0.848	0.575	0.685	0.565	0.383	0.457
3	0.908	0.625	0.741	0.833	0.573	0.679	0.644	0.444	0.525
4	0.913	0.619	0.738	0.848	0.575	0.685	0.565	0.383	0.457

Table 2 Slot filling results (LDC-MAX, English，K3)

	hop0_P	hop0_R	hop0_F	hop1_P	hop1_R	hop1_F	All_P	All_R	All_F
1	0.3231	0.1066	0.1603	0.0000	0.0000	0.0000	0.2675	0.0761	0.1185
2	0.4184	0.1041	0.1667	0.0000	0.0000	0.0000	0.3306	0.0743	0.1213
3	0.4111	0.0939	0.1529	0.0909	0.0063	0.0118	0.3762	0.0688	0.1164
4	0.2727	0.0228	0.0422	0.0000	0.0000	0.0000	0.2647	0.0163	0.0307

Table 3 Event nugget detection results (Micro Average, English)

	plain			mention_type			realis_status		
	P	R	F1	P	R	F1	P	R	F1
1	50.56	13.87	21.76	32.83	9.01	14.13	35.51	9.74	15.29
2	50.56	13.87	21.76	32.83	9.01	14.13	35.51	9.74	15.29
3	64.32	11.31	19.23	42.05	7.39	12.57	45.36	7.97	13.56

Table 4 Sentiment extraction results (LDC-MAX, English, K3)

	hop0_P	hop0_R	hop0_F	hop1_P	hop1_R	hop1_F	All_P	All_R	All_F
1	0.0545	0.0380	0.0448	0.0000	0.0000	0.0000	0.0040	0.0275	0.0070
3	0.0755	0.0506	0.0606	0.0000	0.0000	0.0000	0.0606	0.0367	0.0114

Conclusion

In this paper, we presented the OKS system developed for the Cold Start KB Track of the KBP 2017. The proposed system contains five modules corresponding to the five tasks, namely, the Entity discovery and linking task, the English slot filling task, the Event Nugget Detection and Coreference task and the sentiment task. The official evaluation results are also provided.

Acknowledgements

This work is supported by National Key Research and Development Program of China under grant 2016YFB1000902, National Grand Fundamental Research 973 Program of China under grant 2014CB340406, and National Natural Science Foundation of China under grants 61772501, 61572473, 61572469, 91646120, and 61402022.

References

Li M, Chen X, et.al. OpenKN at TAC KBP 2016. In Proceedings of TAC-KBP 2016.

Finkel J. Rose, Grenager T. and Manning C. 2005. Incorporating non-local information into information extraction systems by gibbs sampling, Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics, 363-370.

Mausam, Schmitz M., Bart R., Soderland S., and Etzioni O. 2012. Open language learning for information extraction. In Proceedings of EMNLP.

Soderland S., Gilmer J., Robert Bart, Oren Et- zioni, and Daniel S. Weld. 2013. Open information extraction to KBP relations in 3 hours. In Proceedings of TAC-KBP 2013.

Weston J, Bordes A, Yakhnenko O, et al. Connecting Language and Knowledge Bases with Embedding Models for Relation Extraction[J]. 2013.

Lin H, Zhao Z, Jia Y, et.al. OpenKN at TAC KBP 2014. In Proceedings of TAC-KBP 2014.

Chen X, Jia Y, Wang Y, eet al., OpenKN at TAC KBP 2015. In Proceedings of TAC-KBP 2015.

Toutanova K., Klein D., Manning C., and Singer Y. 2003. Feature-rich Part-of-Speech Tagging with a cyclic dependency network. Proceedings of HLT-NAACL 2003, 252-259.

Soderland S., Hawkins N., Kim G. L., and Weld D. S. 2015. University of Washington System for 2015 KBP Cold Start Slot Filling. In Proceedings of TAC-KBP 2015.

Marneffe M. D. and Manning C. 2008. Stanford Dependencies manual.

Wu F. and Weld D. 2007. Autonomously semantifying Wikipedia. Proceedings of the sixteenth ACM conference on information and knowledge management, 41-50.

Manning, Christopher D., Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations, pp. 55-6