ISCAS_Sogou at TAC-KBP 2017

Xianpei Han[1], **Xiliang Song**[1,2], **Hongyu Lin**[1,2], **Qichen Zhu**[1,2], **Yaojie Lu**[1,2], **Le Sun**[1]
[1]Institute of Software, Chinese Academy of Sciences
[2]University of Chinese Academy of Sciences
{xianpei,xiliang2014,hongyu2016,qichen2015,yaojie2017,sunle}@iscas.ac.cn

Jingfang Xu[3], **Mingrong Liu**[3], **Ranxu Su**[3], **Sheng Shang**[3], **ChenWei Ran**[3], **FeiFei Xu**[3]
[3]Sogou Inc.
{xujingfang,liumingrong,suranxu,shangsheng,ranchenwei,xufeifei}@sogou-inc.com

Abstract

ISCAS_Sogou participated in the entire Chinese Coldstart task in TAC-KBP 2017. This paper describes our cold start knowledge base population system. Our system consists of four components: Entity Discovery and Coreference, Relation Extraction, Event Extraction and BeSt Detection. We first briefly describe the architecture of our system. And then we introduce each module in detail.

1 Introduction

Knowledge Base (KB) plays a vital role in artificial intelligence. National Institute of Standards and Technology (NIST) have organized the TAC Knowledge Base Population tracks for many years. The goal of TAC Knowledge Base Population (KBP) track is to develop and evaluate technologies for populating knowledge bases (KBs) from unstructured text.

This is the first year that ISCAS_Sogou participates in the TAC-KBP track. We participated in the entire Chinese Coldstart task, which requires a system to build up a Knowledge given a predefined KB schema and a collection of documents. Apart from Slot Filling task, this year coldstart task added new KB schemas, including events, event arguments and sentiment, which make the entire task more challenging.

To handle this problem, ISCAS_Sogou developed a system which includes four modules: Entity Discovery and Coreference, Relation Extraction, Event Extraction and BeSt Detection. The overall system architecture is shown in Figure 1. We first preprocessed all documents using Stanford CoreNLP toolkit (Manning et al., 2014). And then an Entity Discovery and Linking module is used to identify and link entities for downstream modules. After that, Relation Extraction, Event Extraction and Belief and Sentiment Extraction modules are independently applied to extract related assertions. Finally, output assertions of different modules are combined to construct the final knowledge base.

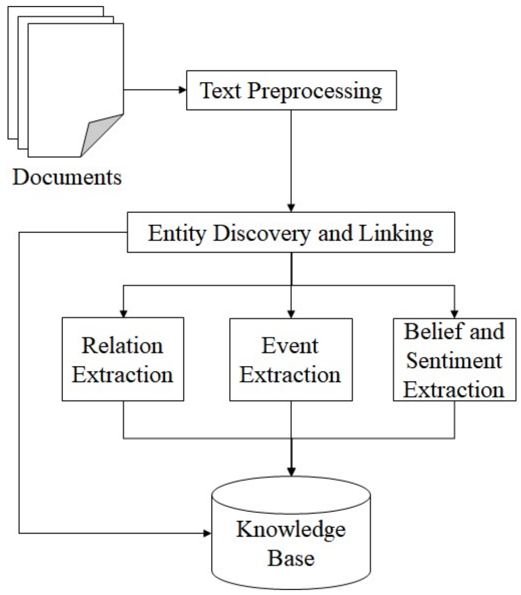

Figure1: Overall system architecture.

The rest of this paper is organized as follows. Section 2 describes our Entity Discovery and Coreference module. From Section 3 to Section 5, we illustrate our Relation Extraction, Event Extraction and Belief and Sentiment Extraction modules, as well as the sub-task evaluation results respectively. Section 6 describes the overall system performance on the KBP Chinese coldstart task using composite evaluation. Finally, we conclude this paper in Section 7.

2 Entity Discovery and Coreference

2.1 Entity Discovery

As in KBP 2016, KBP 2017 participators are required to build a system which can detect all names(NAM) and nominal(NOM) mentions of specific, individual PER, ORG, GPE, LOC, and FAC entities in the corpus.

We first use CRF-based models to extract entity mentions, including named entity mentions and nominal entity mentions. Because downstream components rely heavily on the result of entity discovery, especially the recall of entity mentions, we also use some strategies to boost the recall, including a dictionary based method and a pattern based method.

Named Entity Discovery. To detect named entity mentions, we used two CRF models trained on different datasets. The first one is publicly available Stanford NER model (Finkel et al., 2005), which is trained on Ontonotes 5.0 (Weischedel et al., 2012). The second is the model trained on a combined corpus of Rich ERE dataset, ACE2005 dataset and Ontonotes 5.0, as well as the training and evaluation data of trilingual EDL in 2015 and 2016. We use some heuristic rules to combine the outputs of different models.

Nominal Entity Discovery. Similar to Named Entity Discovery, we retrained a CRF model to extract Nominal Entity using Stanford NER tool. The segmentation and training dataset are the same as the above, but use NOM mentions.

Mention Dictionaries. In order to increase the recall of entity mentions, we construct one NAM dictionary and one NOM dictionary, and recognize more entity mentions through dictionary-based matching under some constraints. Specifically, our NAM dictionary mainly contains GPE and OGR mentions mined from web. The NOM dictionary mainly contains head words of FAC, GPE, LOC, ORG, and PER types, which are frequently used in nominal mentions to refer to entities of specific type.

Based on these dictionaries, candidate entity mentions are recognized by matching tokens sentences with words in dictionaries. Then we simply applied some heuristic rules to filter out noisy entity mentions.

Besides, in Chinese, many abbreviations often appear together as whole, such as "亚欧/Asia and Europe", "中美/China and the United States", "港澳台/Hongkong, Macao and Taiwan", "京津冀/Beijing Tianjin and Hebei". As Tan et al., 2016, we also create a dictionary for these entity mentions. We first detect these words as a whole, and then mapping parts of these words into entities according to the dictionary.

Pattern-based Extraction. Furthermore, we also employ patterns to extract extra mentions. For example, pattern "{([{ tag:/JJ|N.*/ }]* /俱乐部/) => "ORG" }" means that a sequence ended with "俱乐部/club" and preceded by a noun or adjective words, e.g. "罗马俱乐部/Club of Rome", will be recognized as an ORG entity mention.

Besides, for forum documents, we use regular expressions to extract post authors as persons. And mentions which have similar string to the post authors are identified as PER too.

We also use some combination rules to merge small-grained entity mentions to a larger one. For example, if we extract two mentions "莫斯科/Moscow _GPE" "大学/ university _ORG" from string "莫斯科大学/Moscow State University", we then merge these two mentions according to the combination rule "GPE + ORG -> ORG" and get a new mention "莫斯科大学_ORG".

2.2 Entity Coreference

For coreference resolution, we first use the Stanford's Multi-Pass Sieve Coreference Resolution System (Raghunathan et al., 2010; Lee et al., 2011) to obtain document-level coreference result, and then we refine the results using some Chinese specific heuristic rules. After that, we perform cross-document coreference resolution according using mention similarities, which are computed using hand-craft features.

Intra-document coreference. We first perform Coreference Resolution using the Stanford's Multi-Pass Sieve Coreference Resolution System. Then some heuristic rules are used to merge different coreference chains if two chains contain similar entity mentions.

Cross-document Coreference. For cross-document coreference resolution, we use a NAM synonym dictionary mined from web and two coreference chains will be combined if they contain synonym names and are of the same entity type.

2.3 Experiment Result

Table 1 shows the performance of Chinese entity discovery, including overall result, NAM result and NOM result. Table 2 shows the performance on pre-defined five types. We can see that our system achieved a high recall rate on both the strong_mention_match and the strong_typed_metion_match metrics. This is because our Entity Discovery module prefers high recall rather than high precision. Besides, the F-Measure results on both NAM and NOM mentions under strong_typed_metion_match metric ranked on the top among all CSKB teams.

3 Relation Extraction

In this component, we extract relation triples from NLP annotated sentences. Due to the lack of labeled Chinese relation instances, we build our Chinese relation extractors mainly depend on manually-crafted patterns. That is, relation candidates are first extracted using hand-crafted patterns. Then these candidates are validated and scored by training classifiers using heuristically labeled corpus.

3.1 Relation Extractors

The goal of a relation extractor is to extract relation triples from sentences. Unfortunately, traditional relation extraction approaches are mainly supervised (Kambhatla 2004; Zhang et al., 2006) and require expensive labeled data, therefore cannot be used in this task. Furthermore, we found that there were little open released Chinese knowledge bases which contain relations of the same types of KBP relations, therefore it is also hard to apply our previously developed distant supervision techniques (Han and Sun, 2016; Han and Sun, 2017) for this task.

Based on the above observations, we build our Chinese relation extractors manly depend on pattern-based extractors. In order to improve the recall of relation triples, we also employ several heuristically labeling techniques to train our relation extractors. Figure 2 shows the framework of our relation extraction component. In following we describe each component of this framework in detail.

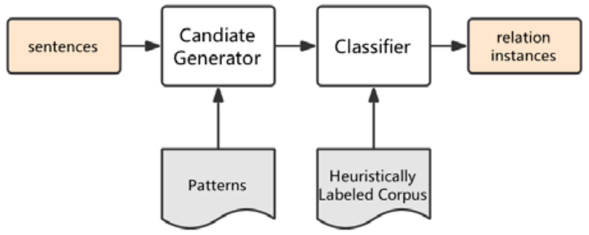

Figure 2: Our Relation Extraction Framework

Pattern-based Relation Candidate Generation. In the first step of our RE component, we generate relation candidates using manually-crafted patterns. For instance, our system will generate one candidate *per_title(章鹏/Zhang Peng, 董事长/CEO)* from the sentence "*联合集团/United Group 董事长/CEO 章鹏/ Zhang Peng*".

To develop patterns for relation extraction, we observed that there are roughly three types of expressions for relation instances in KBP SF 2016, including *modifier expression*, *SVO expression* and *trigger-headed expression*. For modifier expression, a relation instance is a noun phrase and one argument is the modifier of another argument. For example, "*美国/the United States 总统/President 特朗普/ Donald Trump*" is a modifier expression for relation *President-of(美国/the United States, 特朗普/ Donald Trump)*. For SVO expression, a relation instance is expressed in a simple "subject verb object" format, where the verb phrase expresses the relation between the subject and the object. For example, "*川普/Donald Trump 是/is 美国总统/the President of the United States*". For the trigger-headed pattern, the relation between two entities are captured by the dependency path between them, and the dependency path are headed in a relation-specific word such as 去世/die for *per:date_of_death*, 出生/born for *per:date_of_birth*, etc. Table 3 shows several examples of the above three types of expressions. And we found that the above three types of expressions can cover >90% of relation justification instances of the KBP CS 2016 SF queries.

Types of Relation Expression	Examples
Modifier Expression	[苹果公司董事长] [乔布斯] [美国总统][特朗普]
SVO Expression	[特朗普]当选[美国总统] 袁世凯出生于 1859 年
Trigger-Headed Expression	南非前总统曼德拉因病医治无效，于当地时间５日２０时５０分去世，享年９５岁。

Table 3. Three types of relation expressions

Based on the above observations, we use two types of extraction patterns for candidate extraction. The first is token sequence pattern using the Stanford TokenRegex package(Chang and Manning, 2014), which captures the sequence patterns between two entities. For example, we design a "*[ner:PERSON] / 出生于 /born/ [ner:DATE]*" pattern to match the sentence "袁世凯出生于 1859". And we found the TokenRegex patterns work well for the modifier expressions and the SVO expressions. The second kind of pattern is dependency tree based pattern, which is triggered by a head word (e.g., 出生/born and 诞生 /born for per:date_of_birth relation) and arguments are extracted by argument-headword dependency path patterns (e.g., arg1 of *per:date_of_birth* relation must be *nsubj* of 出生).We found this trigger-pattern works well for the trigger-headed expression.

Classifier-based Candidate Validator. The above relation pattern can generate many relation instances. However, we found that precise patterns will often result in a low recall and high-recall patterns will often result in a low precision, i.e., the precision-recall tradeoff problem. In this paper, we address this problem by using high-recall relation extraction patterns, but filter out wrong relation instances using relation classifiers.

Specifically, we train a SVM classifier for each relation type, using features include word features, word sequence features, and dependency path features. Because KBP CS task only provides little labelled instances, we heuristically construct training corpus using two strategies. The first is the distant supervision strategy(Mintz et al., 2009), which matches relations instances in training corpus(i.e., extracted from KBP 2016 corpus) with relations in Wikidata (https://www.wikidata.org). However, we found the distant supervision strategy can only match very little instances. To get more training instances, we employ a new strategy, i.e., the candidates extracted by high-confident patterns are also used as positive instances. Using the heuristically labeled corpus, we found the trained classifier can give a reasonable confidence score to each relation candidate.

3.2 Relation Extraction Performance

The SF results on KBP 2017 coldstart task are shown in Table 4, where Run-1 is the output using both pattern-based candidate generator and classifier-based validator, and Run-2 is the output which only uses pattern-based candidate generator for higher recall than Run-1.

RunID	Precision	Recall	F1
		SF-ALL-Micro	
Run-1	0.2045	0.0798	0.1148
Run-2	0.2175	0.0773	0.1141
		LDC-MEAN-ALL-Macro	
Run-1	0.0827	0.0842	0.0703
Run-2	0.0811	0.0826	0.0692

Table 4. The SF Performance on KBP 2017 CS task.

From the results in Table 4, we can see that:

1) The pattern-based relation extractor can only find a limited amount of relation instances: its recall is below 10%.

2) Even a heuristically labeled corpus can be used to enhance the performance of relation extractors: by applying classifier-based candidate validator, our system can get some performance improvement.

3) Although there exist a lot of weak-supervised relation extraction techniques, the lack of labeled data and the lack of open released Chinese knowledge base limit the use of these techniques.

4 Event Extraction

We applied a two-step, neural network based method in event extraction module. The extraction procedure is split into two parts. First we identify event nuggets from each sentence and recognize its event type and realis value. After that we extract event arguments by classifying the relation between event nuggets and all detected entities in the sentence. We used similar architecture of Dynamic Multi-pooling Convolutional neural networks (DMCNN) proposed by Chen et al. (2015) as our basic model, but do some improvement according to the characteristics of Chinese.

4.1 Dynamic Multi-pooling Convolutional neural networks

Our network architecture is similar to the DMCNN model proposed by Chen et al. (2015), which takes words in the sentence, as well as their relative positions to the concerning token as inputs, and uses a convolutional neural network with a softmax layer at the top to classify the property of the concerning token. Figure 3 shows the overall architecture of DMCNN used in Event Nugget Detection. Specifically, given input tokens $\mathbf{t} = \{t_1, t_2, \cdots, t_n\}$ and their relative positions to concerning token $\mathbf{p} = \{p_1, p_2, \cdots, p_n\}$, $\mathbf{x_i}$ is defined as a d-dimensional representation which is a concatenation of word embedding of t_i and its positional embedding of p_i. A convolutional layer is then applied to capture the sentence-level semantics of the entire sentence. Specifically, k convolutional filters **w** with window size of h words are used to generate the convolutional feature maps:

$$r_{ij} = tanh(\mathbf{w_i} \cdot \mathbf{x_{j:j+h-1}} + b_i)$$

Here $\mathbf{x_{i:j}}$ is the concatenation of embeddings from $\mathbf{x_i}$ to $\mathbf{x_j}$, $\mathbf{w_i}$ is i-th row of w, b_i is a bias term. Then we follow Chen et al. (2015) to divide feature maps into two different parts according to the position of concerning token t_c and do max-pooling at each part:

$$r_i^{left} = max_{j<c} r_{ij}$$
$$r_i^{right} = max_{j\geq c} r_{ij}$$

After we obtain $\mathrm{r_i^{left}}$ and $\mathrm{r_i^{right}}$ from all feature maps, we simply concatenate all of them, as well as the embedding of tokens near to $\mathrm{t_c}$ as lexical features to form an overall feature vector f, then a linear layer is applied to obtain the scores for each event types (or realis types):

$$O = Uf + b$$

Here U is weight matrix and b is the bias term. Then a softmax layer is used to normalize the scores into probabilities:

$$P(y_i|x;\theta) = \frac{e^{O_i}}{\sum_{j=1}^{d} e^{O_j}}$$

where $\mathrm{O_i}$ is the i-th element in **O**.

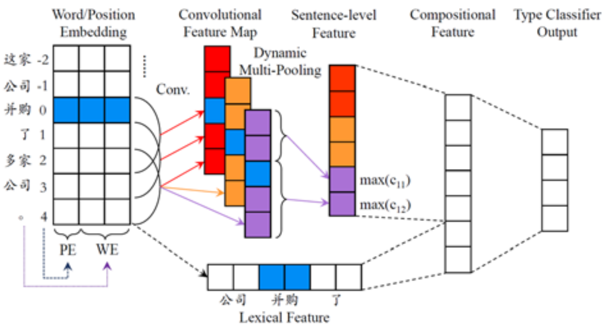

Figure 3: The architecture of Dynamic Multi-pooling Convolutional neural networks

The model applied to Event Argument Extraction is quite similar to the one in Event Nugget Detection, except two differences. First, the feature maps are split into three parts according to the position of the candidate entity headword and the position of trigger word. Second, the lexical features contain not only word embeddings of surrounding words of trigger, but also include the surrounding words of candidate entity head word. Apart from these two differences, models applied in two different steps are the same.

4.2 Errata Table

We found that in Chinese, there exist severe mismatches between word spans and event nuggets, and the trigger nugget is part of a word in the majority of those mismatches.

To handle this, in event nugget detection procedure, we introduce an errata table extracted from the training data and replaced those words that part of whom was a trigger nugget with that trigger directly. If one word in the training data never serves as an event trigger but part of it in some occurrence is annotated as trigger nugget, then there will be a map from this word to its trigger part added to the errata table. In this way we are able to alleviate the trigger-word mismatch problem raising from word segmentation in Chinese.

Besides, in order to boost the recall of our system, we also tried to replace words that not in the training data with its synonym which appears as a trigger in the training data. We found this have no significant effect on the development set, but can make a little improvement on the final composite evaluation results.

4.3 Event Coreference

For event coreference, we used a simple heuristic method. We regarded two events whose triggers are synonyms and shares at least one argument in one document as the same event. And we do not participate in cross-document event coreference task. However, we are surprisingly to find that our systems are still very competitive in the final evaluation using coreference-related evaluation metrics.

4.4 Model Training and Datasets

Since all parameters in DMCNN are differentiable, it can be trained by maximizing the log-likelihood using any gradient-based method. And this paper uses minibatch stochastic gradient descent method with the Adadelta update rule (Zeiler 2012) to train our models. We randomly sampled 20 documents from the Rich ERE 2016 Evaluation corpus as our development set. The leftover of it, as well as previous Rich ERE data and ACE2005 dataset, were used as the training set.

4.5 Event Extraction Performance

We reported both composite and component performance of our Event Extraction systems on KBP 2017 Coldstart task. There are two different systems we submitted, where Run-1 is the system without synonym expansion of trigger words and Run-2 is the system with it.

4.5.1 Component Performance

Table 5 shows the component performance on Event Nugget Detection and coreference task. We reported F1 scores under different evaluation metrics. We can see that our system achieved very competitive performance. The performance on Realis and Type+ Realis tasks ranked 1st in all CSKB teams and 2nd in all Event Nugget teams. Besides, our systems ranked 2nd in all CSKB teams in Plain, and Type tasks, as well as CONLL metric that considers event coreference. But notice that in our KB submission we removed all event nuggets without any argument detected, which will not be considered in composite evaluation. This significant dropped our system recall and thus led to negative effects on Event Nugget component evolution. However, our systems still achieved very competitive results, which demonstrate the effectiveness of our system.

Table 6 and Table 7 show the performance of our systems on Event Argument and Linking Evaluation. Our system performance on this component evaluation surpasses all other CSKB teams by a large margin in all evaluation metrics. Also we can see that synonym expansion do have a little influence on the final results, but the difference is not significant.

	Run-1	**Run-2**
Plain	43.99	43.73
Type	40.01	39.76
Realis	35.35	35.03
Type + Realis	32.43	32.16
Coref CONLL	20.80	20.32

Table 5: Component evaluation results of Event Nugget Detection and Coreference.

	Run-1	**Run-2**
ArgScore_95	11.26	11.32
Link_95	7.46	7.45
ArgF1	22.77	22.80

Table 6: Component evaluation results of Event Argument and Linking (WithRealis Scores)

	Run-1	**Run-2**
ArgScore_95	14.66	14.67
Link_95	8.80	8.79
ArgF1	27.20	27.27

Table 7: Component evaluation results of Event Argument and Linking (NoRealis Scores)

4.5.2 Composite Performance

Table 8 shows the composite evaluation results of the entire event extraction system. From this table we can see that synonym expansion slightly improved the results of composite evaluation.

RunID	**Precision**	**Recall**	**F1**
	SF-ALL-Micro		
Run-1	0.5938	0.1699	0.2643
Run-2	0.6025	0.1735	0.2694
	LDC-MEAN-ALL-Macro		
Run-1	0.2497	0.1324	0.1605
Run-2	0.2503	0.1329	0.1612

Table 8: Composite evaluation results of Event Module

5 BeSt Detection

In this section, we introduce the architecture of our sentiment system. Our system consists of four modules: Preprocess, Subjectivity Classification, Polarity Classification and Source/Target Extraction. The overall framework is shown in Figure 4.

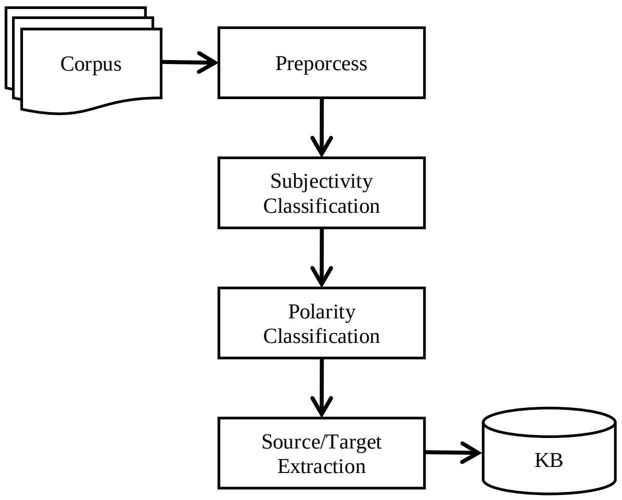

Figure 4: The architecture of our BeSt Detection System.

5.1 Preprocess

In this module, we use HIT LTP toolkit (Che et al., 2010) to preprocess original documents. The preprocessing pipeline includes sentence segmentation, word segmentation and POS tagging. For forum data, we also extract the initial poster and the poster of each post according to the XML tagging.

5.2 Subjectivity Classification

In the analysis of the 2016 Golden dataset, we found that only 10% of forum posts have subjective sentiment or belief, and the percentage is even lower in the news dataset. Based on the above observation, we improve accuracy and efficiency by first conducting subjective classification to identify subjective sentences.

We use the following methods to identify subjective sentences:

Sentimental word dictionary. We construct a sentimental word dictionary from the training corpus. The confidence of each word being a sentimental word is calculated using a label propagation algorithm. And we classify a sentence into subjective if it contains sentimental words.

Pattern based method. To capture more sentimental expressions, we also propose some pattern based methods. We first manually write several patterns, and then a bootstrapping method is used to expand patterns. The overall extraction procedure is shown in Figure 5. Finally we build 176 patterns for subjective classification.

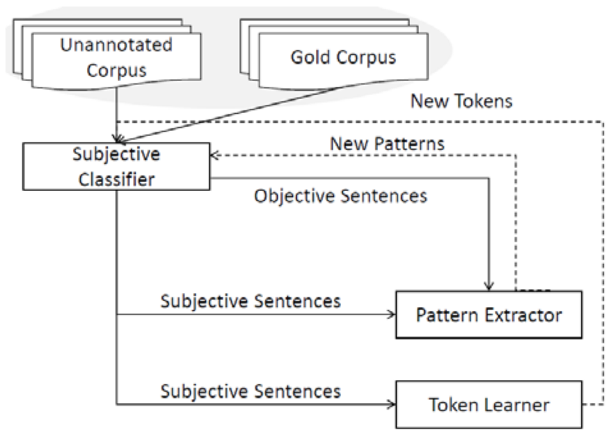

Figure 5: The architecture of our Subjectivity Classification System.

5.3 Polarity Classification

For sentimental polarity classification, we propose a rule based system which using patterns, sentimental word dictionary, and some heuristic rules developed from training dataset.

The classification is as follows. Firstly, if a sentence matches one polarity pattern, then its polarity is the same as the polarity of pattern. If no pattern is matched, a sentence will be further classified according to the sentimental words in the sentence. Finally, if a sentence cannot be classified by previous two steps, we will classify it into negative if it appears in a forum document and into positive if it appears in a news document. This is because we observed that almost all subjective sentences in forum documents are negative, and a majority of subjective sentences in news documents are positive.

5.4 Source and Target Extraction

After classifying the sentimental polarity of a sentence, we need to extract the source and the target of this sentimental tendency. To achieve this goal, we modify our patterns can also match the source and the target. To boost the recall, we further write some rules to deal with the conjunctions and the juxtaposition structure in the Chinese. Finally, we also refine our results based on the entity coreference results.

5.5 BeSt Results

The composite evaluation results of sentiment slots are shown in Table 9. We submit two run: a balanced system (Run-1) and a high recall run (Run-2). We get the high recall run using a simple match patterns, instead of strict patterns. We can see that Run-1 have a better performance than Run-2, which means that our patterns is effective for the BeSt task.

RunID	Hop	Precision	Recall	F1
	SF-ALL-Micro			
Run-1	0	0.2828	0.1253	0.1736
	1	0.0313	0.0743	0.0441
	ALL	0.0908	0.1061	0.0979
Run-2	0	0.1348	0.1655	0.1486
	1	0.0094	0.0967	0.0171
	ALL	0.0301	0.1397	0.0496
	LDC-MEAN-ALL-Macro			
Run-1	0	0.1657	0.1411	0.1107
	1	0.0216	0.0315	0.0228
	ALL	0.0999	0.0911	0.0706
Run-2	0	0.0955	0.1758	0.0932
	1	0.0102	0.0363	0.0136
	ALL	0.0565	0.1121	0.0569

Table 9: Composite evaluation results of sentiment slots.

6 Overall Performance

RunID	Precision	Recall	F1
	SF-ALL-Micro		
Run-1	0.1461	0.1020	0.1201
Run-2	0.0602	0.1122	0.0783
LDC-Manual	0.8692	0.2362	0.3715
	LDC-MEAN-ALL-Macro		
Run-1	0.1211	0.0960	0.0896
Run-2	0.1112	0.0998	0.0863
LDC-Manual	0.6964	0.5680	0.5923

Table 10: Composite evaluation results of all slots types (All Hops).

RunID	Precision	Recall	F1
	SF-ALL-Micro		
Run-1	0.3610	0.1229	0.1834
Run-2	0.2439	0.1346	0.1735
LDC-Manual	0.8841	0.2766	0.4213
	LDC-MEAN-ALL-Macro		
Run-1	0.1696	0.1265	0.1207
Run-2	0.1570	0.1320	0.1175
LDC-Manual	0.7765	0.6035	0.6376

Table 11: Composite evaluation results of all slots types (Hop 0).

RunID	Precision	Recall	F1
	SF-ALL-Micro		
Run-1	0.0374	0.0558	0.0448
Run-2	0.0131	0.0626	0.0217
LDC-Manual	0.8120	0.1469	0.2488
	LDC-MEAN-ALL-Macro		
Run-1	0.0308	0.0392	0.0319
Run-2	0.0259	0.0399	0.0283
LDC-Manual	0.5472	0.5019	0.5079

Table 12: Composite evaluation results of all slots types (Hop 1).

Table 10 to Table 12 shows the performances of our systems and LDC manual run in composite evaluation of all slots. From these tables we can see that there still exists a large margin between our system performances and manual results. This demonstrates that coldstart knowledge base population is still a very challenging task.

7 Conclusion

In this year ISCAS_Sogou participates in the entire Chinese Coldstart task of TAC-KBP 2017, which requires a system to populate a Knowledge base given a predefined KB schema and a collection of documents. Evaluation results demonstrate that our system can achieve state-of-the-art performance in some tasks on some evaluation metrics. However, the large performance margin between automatic IE techniques and manual results shows that coldstart knowledge base population is still a very challenging task, and more techniques and resources are need to be developed.

References

Angel X. Chang and Christopher D. Manning. 2014. TokensRegex: Defining cascaded regular expressionsover tokens. Technical Report CSTR 2014-02, Department of Computer Science, Stanford University.

Finkel, Jenny Rose, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In: Proceedings of the 43rd annual meeting on association for computational linguistics. Association for Computational Linguistics,pp. 363-370.

Yongmei Tan, Xiaoguang Li and Di zheng. BUPTTeam Participation at TAC 2016 Knowledge Base Population. 2016. In:2016 Text Analysis Conference (TAC) Workshop.

Lee, Heeyoung, et al. 2011. Stanford's multi-pass sieve coreference resolution system at the CoNLL-2011 shared task. In:Proceedings of the fifteenth conference on computational natural language learning: Shared task. Association for Computational Linguistics,pp. 28-34.

Raghunathan, Karthik, et al. 2010. A multi-pass sieve for coreference resolution. In:Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing. Association for Computational Linguistics,pp. 492-501.

Chen, Yubo, et al. 2015. Event Extraction via Dynamic Multi-Pooling Convolutional Neural Networks. In: Proceedings of ACL 2015,pp 167-176.

Christopher D. Manning, Mihai Surdeanu, John Bauer, J enny Finkel, Steven J. Bethard, and David McClosky. The Stanford CoreNLP Natural Language Processing Toolkit. In ACL 2014.

Mike Mintz, Steven Bills, Rion Snow, and Dan Juraf- sky. 2009. Distant supervision for relation extraction without labeled data. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2, pages 1003–1011.

Kambhatla, N. 2004. *Combining lexical, syntactic, and semantic features with maximum entropy models for extracting relations*. In: Proceedings of ACL 2004, pp. 178–181.

Ralph Weischedel, Martha Palmer, Mitchell Marcus, Ed uard Hovy, Sameer Pradhan, Lance Ramshaw, Nianw en Xue, Ann Taylor, Jeff Kaufman, Michelle Franchini , Mohammed ElBachouti, Robert Belvin, and Ann Ho uston. OntoNotes Release 5.0. LDC2011T03, Philadel phia, Penn.: Linguistic Data Consortium.

Wanxiang Che, Zhenghua Li, and Ting Liu. LTP: A Chines e Language Technology Platform. In Proceedings of t he Coling 2010.

Xianpei Han and Le Sun. 2017. Distant Supervision via Prototype-based Global Representation Learning. In: the Thirty-First AAAI Conference.

Xianpei Han and Le Sun. 2016. Global Distant Supervision for Relation Extraction. In: The 30th AAAI Conference on Artificial Intelligence.

Zhang, M., Zhang, J., and Su, J. 2006. *Exploring syntactic features for relation extraction using a convolution tree kernel*. In: Proceedings of NAACL-HLT 2006, pp. 288–295.

	strong_mention_match			strong_typed_metion_match			mention_ceaf			b_cubed		
	P	R	F	P	R	F	P	R	F	P	R	F
All	0.562	0.715	0.630	0.517	0.659	0.579	0.340	0.433	0.381	0.507	0.325	0.396
NAM	0.782	0.780	0.781	0.767	0.765	0.766	0.497	0.496	0.497	0.734	0.393	0.512
NOM	0.155	0.340	0.212	0.136	0.299	0.187	0.089	0.196	0.123	0.138	0.135	0.136

Table 1: The overall Entity Discovery and Linking Result.

	strong_mention_match			strong_typed_metion_match			mention_ceaf			b_cubed		
	P	R	F	P	R	F	P	R	F	P	R	F
PER	0.377	0.751	0.502	0.360	0.717	0.479	0.294	0.585	0.391	0.328	0.505	0.397
ORG	0.495	0.513	0.504	0.429	0.444	0.436	0.313	0.324	0.319	0.442	0.225	0.299
LOC	0.584	0.390	0.468	0.532	0.355	0.426	0.302	0.202	0.242	0.559	0.128	0.208
GPE	0.761	0.819	0.789	0.754	0.812	0.782	0.394	0.424	0.408	0.702	0.284	0.404
FAC	0.189	0.118	0.145	0.122	0.076	0.094	0.159	0.099	0.122	0.143	0.069	0.093

Table 2: The Entity Discovery and Linking results on the five pre-defined types.