FBK-IRST: Semantic Relation Extraction using Cyc

Kateryna Tymoshenko and **Claudio Giuliano**
FBK-IRST
I-38050, Povo (TN), Italy
tymoshenko@fbk.eu, giuliano@fbk.eu

Abstract

We present an approach for semantic relation extraction between nominals that combines semantic information with shallow syntactic processing. We propose to use the ResearchCyc knowledge base as a source of semantic information about nominals. Each source of information is represented by a specific kernel function. The experiments were carried out using support vector machines as a classifier. The system achieves an overall F_1 of 77.62% on the "Multi-Way Classification of Semantic Relations Between Pairs of Nominals" task at SemEval-2010.

1 Introduction

The SemEval-2010 Task 8 "Multi-Way Classification of Semantic Relations Between Pairs of Nominals" consists in identifying which semantic relation holds between two nominals in a sentence (Hendrickx et al., 2010). The set of relations is composed of nine mutually exclusive semantic relations and the *Other* relation. Specifically, the task requires to return the most informative relation between the specified pair of nominals e_1 and e_2 taking into account their order. Annotation guidelines show that semantic knowledge about e_1 and e_2 plays a very important role in distinguishing among different relations. For example, relations *Cause-Effect* and *Product-Producer* are closely related. One of the restrictions which might help to distinguish between them is that products must be concrete physical entities, while effects must not.

Recently, there has emerged a large number of freely available large-scale knowledge bases. The ground idea of our research is to use them as source of semantic information. Among such resources there are DBpedia,[1] YAGO,[2] and OpenCyc.[3] On the one hand, DBpedia and YAGO have been automatically extracted from Wikipedia. They have a good coverage of named entities, but their coverage of common nouns is poorer. They seem to be more suitable for relation extraction between named entities. On the other hand, Cyc is a manually designed knowledge base, which describes actions and entities both in common life and in specific domains (Lenat, 1995). Cyc has a good coverage of common nouns, making it interesting for our task. The full version of Cyc is freely available to the research community as ResearchCyc.[4]

We approached the task using the system introduced by Giuliano et al. (2007) as a basis. They exploited two information sources: the whole sentence where the relation appears, and WordNet synonymy and hyperonymy information. In this paper, we (i) investigate usage of Cyc as a source of semantic knowledge and (ii) linguistic information, which give useful clues to semantic relation extraction. From Cyc, we obtain information about super-classes (in the Cyc terminology *generalizations*) of the classes which correspond to nominals in a sentence. The sentence itself provides linguistic information, such as local contexts of entities, bag of verbs and distance between nominals in the context.

The different sources of information are represented by kernel functions. The final system is based on four kernels (i.e., local context kernel, distance kernel, verbs kernel and generalization kernel). The experiments were carried out using support vector machines (Vapnik, 1998) as a classifier. The system achieves an overall F_1 of

[1] http://dbpedia.org/
[2] http://www.mpi-inf.mpg.de/yago-naga/yago/
[3] http://www.cyc.com/opencyc
[4] http://research.cyc.com/

Proceedings of the 5th International Workshop on Semantic Evaluation, ACL 2010, pages 214–217,
Uppsala, Sweden, 15-16 July 2010.

77.62%.

2 Kernel Methods for Relation Extraction

In order to implement the approach based on shallow syntactic and semantic information, we employed a linear combination of kernels, using the support vector machines as a classifier. We developed two types of basic kernels: syntactic and semantic kernels. They were combined by exploiting the closure properties of kernels. We define the composite kernel $K_C(x_1, x_2)$ as follows.

$$\sum_{i=1}^{n} \frac{K_i(x_1, x_2)}{\sqrt{K_i(x_1, x_1)K_i(x_2, x_2)}}. \quad (1)$$

Each basic kernel K_i is normalized.

All the basic kernels are explicitly calculated as follows

$$K_i(x_1, x_2) = \langle \varphi(x_1), \varphi(x_2) \rangle, \quad (2)$$

where $\varphi(\cdot)$ is the embedding vector. The resulting feature space has high dimensionality. However, Equation 2 can be efficiently computed explicitly because the representations of input are extremely sparse.

2.1 Local context kernel

Local context is represented by terms, lemmata, PoS tags, and orthographic features extracted from a window around the nominals considering the token order. Formally, given a relation example R, we represent a local context $LC = t_{-w}, ..., t_{-1}, t_0, t_{+1}, ..., t_{+w}$ as a row vector

$$\psi_{LC}(R) = (tf_1(LC), tf_2(LC), ..., tf_m(LC)) \in \{0, 1\}^m, \quad (3)$$

where tf_i is a feature function which returns 1 if the feature is active in the specified position of LC; 0 otherwise. The local context kernel $K_{LC}(R_1, R_2)$ is defined as

$$K_{LC\_e1}(R_1, R_2) + K_{LC\_e2}(R_1, R_2), \quad (4)$$

where K_{LC_e1} and K_{LC_e2} are defined by substituting the embedding of the local contexts of e_1 and e_2 into Equation 2, respectively.

2.2 Verb kernel

The verb kernel operates on the verbs present in the sentence,[5] representing it as a *bag-of-verbs*. More formally, given a relation example R, we represent the verbs from it as a row vector

$$\psi_V(R) = (vf(v_1, R), ..., vf(v_l, R)) \in \{0, 1\}^l, \quad (5)$$

where the binary function $vf(v_i, R)$ shows if a particular verb is used in R. By substituting $\psi_V(R)$ into Equation 2 we obtain the bag-of-verbs kernel K_V.

2.3 Distance kernel

Given a relation example $R(e_1, e_2)$, we represent the distance between the nominals as a one-dimensional vector

$$\psi_D(R) = \frac{1}{dist(e_1, e_2)} \in \Re^1, \quad (6)$$

where $dist(e_1, e_2)$ is number of tokens between the nominals e_1 and e_2 in a sentence. By substituting $\psi_D(R)$ into Equation 2 we obtain the distance kernel K_D.

2.4 Cyc-based kernel

Cyc is a comprehensive, manually-build knowledge base developed since 1984 by CycCorp. According to Lenat (1995) it can be considered as an expert system with domain spanning all everyday actions and entities, like *Fish live in water*. The open-source version of Cyc named OpenCyc, which contains the full Cyc ontology and restricted number of assertions, is freely available on the web. Also the full power of Cyc has been made available to the research community via ResearchCyc. Cyc knowledge base contains more than 500,000 concepts and more than 5 million assertions about them. They may refer both to common human knowledge like food or drinks and to specialized knowledge in domains like physics or chemistry. The knowledge base has been formulated using CycL language. A Cyc constant represents a thing or a concept in the world. It may be an individual, e.g. *BarackObama*, or a collection, e.g. *Gun*, *Screaming*.

2.4.1 Generalization kernel

Given a nominal e, we map it to a set of Cyc constants $EC = \{c_i\}$, using the Cyc function *denotation-mapper*. Nominals in Cyc usually denote constants-collections. Notice that we do not perform word sense disambiguation. For each $c_i \in EC$, we query Cyc for collections which generalize it. In Cyc collection X generalizes collection

[5] On average there are 2.65 verbs per sentence

Y if each element of Y is also an element of collection X. For instance, collection *Gun* is generalized by *Weapon, ConventionalWeapon, MechanicalDevice* and others.

The semantic kernel incorporates the data from Cyc described above. More formally, given a relation example R each nominal e is represented as

$$\psi_{EC}(R) = (fc(c_1, e), ..., fc(c_k, e)) \in \{0, 1\}^k, \quad (7)$$

where the binary function $fc(c_i, e)$ shows if a particular Cyc collection c_i is a generalization of e.

The *bag-of-generalizations* kernel $K_{genls}(R_1, R_2)$ is defined as

$$K_{genls\_e1}(R_1, R_2) + K_{genls\_e2}(R_1, R_2), \quad (8)$$

where K_{genls_e1} and K_{genls_e2} are defined by substituting the embedding of generalizations e_1 and e_2 into Equation 2 respectively.

3 Experimental setup and Results

Sentences have been tokenized, lemmatized and PoS tagged with TextPro.[6] Information for generalization kernel has been obtained from ResearchCyc. All the experiments were performed using jSRE customized to embed our kernels.[7] jSRE uses the SVM package LIBSVM (Chang and Lin, 2001). The task is casted as multi-class classification problem with 19 classes (2 classes for each relation to encode the directionality and 1 class to encode *Other*). The multiple classification task is handled with One-Against-One technique. The SVM parameters have been set as follows. The cost-factor W_i for a given class i is set to be the ratio between the number of negative and positive examples. We used two values of regularization parameter C: (i) $C_{def} = \frac{1}{\sum K(x,x)}$ where x are all examples from the training set, (ii) optimized C_{grid} value obtained by brute-force grid search method. The default value is used for the other parameters.

Table 1 shows the performance of different kernel combinations, trained on 8000 training examples, on the test set. The system achieves the best overall macro-average F_1 of 77.62% using $K_{LC} + K_V + K_D + K_{genls}$. Figure 1 shows the learning curves on the test set. Our experimental study has shown that the size of the training set influences the performance of the system. We observe that when the system is trained on 8000 examples the overall F_1 increases for 14.01% as compared to the case of 1000 examples.

[6] http://textpro.fbk.eu/

[7] jSRE is a Java tool for relation extraction available at http://tcc.itc.it/research/textec/tools-resources/jsre.html.

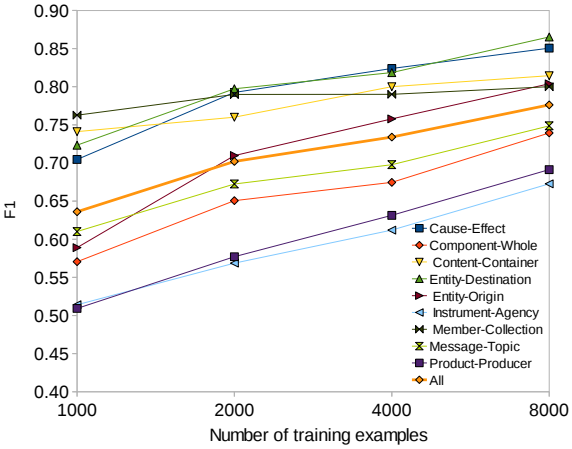

Figure 1: Learning curves on the test set per relation

Kernels	P	R	F_1
$K_{LC} + K_V + K_D + K_{genls}$	74.98	80.69	77.62
$K_{LC} + K_V + K_D + K_{genls}$*	78.51	76.03	77.11
$K_{LC} + K_D + K_{genls}$*	78.14	75.93	76.91
$K_{LC} + K_{genls}$*	78.19	75.70	76.81
$K_{LC} + K_D + K_{genls}$	72.98	80.28	76.39
$K_{LC} + K_{genls}$	73.05	79.98	76.28

Table 1: Performance on the test set. Combinations marked with * were run with C_{grid}, others with C_{def}.

4 Discussion and error analysis

The experiments have shown that K_{LC} is the core kernel of our approach. It has good performance on its own. For instance, it achieves precision of 66.16%, recall 72.67% and F_1 of 69.13% evaluated using 10-fold cross-validation on the training set.

Relation	K_{LC}	$K_{LC} + K_{genls}$	ΔF_1
Cause-Effect	74.29	76.41	2.12
Component-Whole	61.24	66.13	4.89
Content-Container	76.36	79.12	2.76
Entity-Destination	82.85	83.95	1.10
Entity-Origin	72.09	74.13	2.04
Instrument-Agency	57.71	65.51	7.80
Member-Collection	81.30	83.40	2.10
Message-Topic	60.41	69.09	8.68
Product-Producer	55.95	63.52	7.57

Table 2: The contribution of Cyc evaluated on the training set.

Generalization kernel combined with local context kernel gives precision of 70.38%, recall of 76.96%, and F_1 73.47% with the same experimental setting. The increase of F_1 per relation is shown in the Table 2 in the column ΔF_1. The largest F_1 increase is observed for *Instrument-Agency* (+7.80%), *Message-Topic* (+8.68%) and *Product-Producer* (+7.57%). K_{genls} reduces the number of misclassifications between the two directions of the same relation, like *Product-Producer(artist,design)*. It also captures the differences among relations, specified in the annotation guidelines. For instance, the system based only on K_{LC} misclassfied "The <e1>species</e1> makes a squelching <e2>noise</e2>" as *Product-Producer(e2,e1)*. Generalizations for *<e2>noise</e2>* provided by Cyc include *Event, MovementEvent, Sound*. According to the annotation guidelines a product must not be an event. A system based on the combination of K_{LC} and K_{genls} correctly labels this example as *Cause-Effect(e1,e2)*.

K_{genls} improves the performance in general. However, in some cases using Cyc as a source of semantic information is a source of errors. Firstly, sometimes the set of constants for a given nominal is empty (e.g., *disassembler*, *babel*) or does not include the correct one (noun *surge* is mapped to the constant *IncreaseEvent*). In other cases, an ambiguous nominal is mapped to many constants at once. For instance, *notes* is mapped to a set of constants, which includes *MusicalNote*, *Note-Document* and *InformationRecordingProcess*. Word sense disambiguation should help to solve this problem. Other knowledge bases like DBpedia and FreeBase[8] can be used to overcome the problem of lack of coverage.

Bag-of-word kernel with all words from the sentence did not impact the final result.[9] However, the information about verbs present in the sentence represented by K_V helped to improve the performance. A preliminary error analysis shows that a deeper syntactic analysis could help to further improve the performance.

For comparison purposes, we also exploited WordNet information by means of the supersense kernel K_{SS} (Giuliano et al., 2007). In all experiments, K_{SS} was outperformed by K_{genls}. For instance, $K_{LC} + K_{SS}$ gives overall F_1 measure of 70.29% with the same experimental setting as described in the beginning of this section.

[8] http://www.freebase.com/

[9] This kernel has been evaluated only on the training data.

5 Conclusion

The paper describes a system for semantic relations extraction, based on the usage of semantic information provided by ResearchCyc and shallow syntactic features. The experiments have shown that the external knowledge, encoded as super-class information from ResearchCyc without any word sense disambiguation, significantly contributes to improve overall performance of the system. The problem of the lack of coverage may be overcome by the usage of other large-scale knowledge bases, such as DBpedia. For future work, we will try to use the Cyc inference engine to obtain implicit information about nominals in addition to the information about their superclasses and perform word sense disambiguation.

Acknowledgments

The research leading to these results has received funding from the ITCH project (http://itch.fbk.eu), sponsored by the Italian Ministry of University and Research and by the Autonomous Province of Trento and the Copilosk project (http://copilosk.fbk.eu), a Joint Research Project under Future Internet - Internet of Content program of the Information Technology Center, Fondazione Bruno Kessler.

References

Chih-Chung Chang and Chih-Jen Lin, 2001. *LIBSVM: a library for support vector machines*. Software available at http://www.csie.ntu.edu.tw/~cjlin/libsvm.

Claudio Giuliano, Alberto Lavelli, Daniele Pighin, and Lorenza Romano. 2007. Fbk-irst: Kernel methods for semantic relation extraction. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, Prague, Czech Republic, June. Association for Computational Linguistics.

Iris Hendrickx, Su Nam Kim, Zornitsa Kozareva, Preslav Nakov, Diarmuid Ó Séaghdha, Sebastian Padó, Marco Pennacchiotti, Lorenza Romano, and Stan Szpakowicz. 2010. Semeval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. In *Proceedings of the 5th SIGLEX Workshop on Semantic Evaluation*, Uppsala, Sweden.

Douglas B. Lenat. 1995. CYC: A large-scale investment in knowledge infrastructure. *Communications of the ACM*, 38(11):33–38.

Vladimir N. Vapnik. 1998. *Statistical Learning Theory*. Wiley-Interscience, September.