MSM2013 IE Challenge: Annotowatch

Štefan Dlugolinský, Peter Krammer, Marek Ciglan and Michal Laclavík
Institute of Informatics, Slovak Academy of Sciences
{stefan.dlugolinsky, peter.krammer,marek.ciglan, michal.laclavik}@savba.sk

ABSTRACT

In this paper, we describe our approach taken in the MSM2013 IE Challenge, which was aimed at concept extraction from microposts. The goal of the approach was to combine several existing NER tools which use different classification methods and benefit from their combination. Several NER tools have been chosen and individually evaluated on the challenge training set. We observed that some of these tools performed better on different entity types than other tools. In addition, different tools produced diverse results which brought a higher recall when combined than that of the best individual tool. As expected, the precision significantly decreased. The main challenge was in combining annotations extracted by diverse tools. Our approach was to exploit machine-learning methods. We have constructed feature vectors from the annotations yielded by different extraction tools and various text characteristics, and we have used several supervised classifiers to train the classification models. The results showed that several classification models have achieved better results than the best individual extractor.

Categories and Subject Descriptors

I.2.7 [**Natural language processing**]: Language parsing and understanding, Text.

General Terms

Measurement, Performance, Design, Experimentation.

Keywords

Information extraction, machine-learning, named entity recognition, microposts.

1. INTRODUCTION

Most of the current Named Entity Recognition (NER) methods have been designed for concept extraction from relatively long and grammatically correct texts, such as newswire texts or biomedical texts. More and more user-generated content on the Web consists of a relatively short text which is often grammatically incorrect (e.g., microposts, on which these methods perform worse). The goal of the approach proposed in this paper is to combine several different information extraction methods in order to reach a more precise concept extraction on relatively short texts. We hypothesized that if these methods were combined properly, they would perform better than the best individual method from the pool. This assumption was partially proven through the evaluation of several available and well-known NER tools that use different entity extraction methods. The merged results of these tools showed a higher recall than that of the best tool but with a very low precision. The goal was to reduce or eliminate this tradeoff. Higher recall indicates that different methods complement each other and that there is room for improvement. We have tried various machine-learning algorithms and built several models capable of producing results based on concepts extracted by yielded tools. The goal was to produce a model with the highest possible precision approximating the recall measured for unified extracted concepts. In the following sections, we describe the NER tools that have been used and how they individually performed on the MSM2013 IE Challenge (from here on referenced as "challenge") training set (version 1.5). We briefly describe the methodology of our investigation (i.e., how our solution was built).

2. TOOLS USED

Our solution incorporates several available well-known NER tools: *Annie Named Entity Recognizer [1], Apache OpenNLP*[1]*, Illinois Named Entity Tagger (with 4-label type model) [2], Illinois Wikifier [3], LingPipe (with English News - MUC-6 model)*[2]*, Open Calais*[3]*, Stanford Named Entity Recognizer (with 4 class caseless model) [4], WikiMiner*[4]. This list is complemented by the *Miscinator*, a tool specifically designed for the challenge. The Miscinator detects MISC concepts (i.e., entertainment/award event, sports event, movies, TV shows, political event, and programming languages). One of the tools' evaluation conclusions was that they were not performing well in detecting entertainment, award, and sports events. Therefore, we built a specialized gazetteer annotation tool for this task. The gazetteer has been constructed from the events annotations found in the challenge training set extended by Google Sets service (a method trained on web crawls) which generates list of items based on several examples. The only customization made to listed tools was the mapping of their annotation types to match target entity types (i.e., Location - LOC, Person - PER and Organization – ORG) as well as filtering unimportant ones (e.g., Token). Relevant OpenCalais entities to target entities were similarly mapped. Illinois Wikifier was treated a bit differently, as it provided annotations with Wikipedia concepts and the yielded output did not comprise the type classification for the annotations. To overcome this drawback, we mapped the annotations to the DBPedia knowledge base and used DBPedia types associated with the given concepts to derive target entity types. WikiMiner annotations were mapped the same way.

3. EVALUATION OF USED TOOLS

All of the tools used were evaluated on the challenge training set. There were three ways of computing the Precision, Recall, and F_1 metrics used. The first method was *strict* (P_S, R_S and F_{1S}), which considered partially correct responses as incorrect; however, the second, *lenient*, considered them as correct (P_L, R_L and F_{1L}). The third method was an *average* of the previous two (P_A, R_A, and $F1_A$). The evaluation results are shown in Figure 1. We also evaluated the unified responses of all of the tools. Results showed that the recall was much higher ($R_S = 90\%$) than the best individual tool (Illinois NER got $R_S = 60\%$), but the precision was very poor ($P_S = 18\%$), hence the F_1 score ($F_{1S} = 30\%$). The best performing tool on microposts was OpenCalais, which scored $P_S = 70\%$, $R_S = 58\%$ and $F_{1S} = 64\%$.

[1] http://opennlp.apache.org
[2] http://alias-i.com/lingpipe
[3] http://www.opencalais.com/about
[4] http://wikipedia-miner.cms.waikato.ac.nz

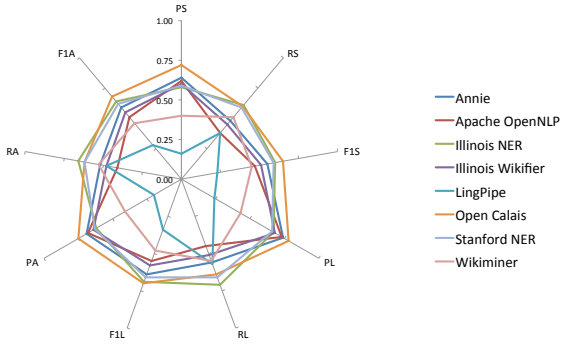

Figure 1. Micro summary of NER tools over training set v1.5

4. MACHINE-LEARNING

Our goal was to create a model that would take the most relevant results detected by each tool and perform better than the best tool did individually. We have used statistical classifiers to achieve this goal.

4.1 Input Features

We have taken the approach of describing how particular extractors performed on different entity types compared to the response of other extractors. Used as a training vector, this description was an input for training a classification model. A vector of input training features was generated for each annotation found by integrated NER tools. We called this annotation a reference annotation. The vector of each reference annotation consisted of several sub-vectors. The first sub-vector of the training vector was an annotation vector. The annotation vector described the reference annotation--whether it was uppercase or lowercase, used a capital first letter or capitalized all of its words, the word count, and the type of the detected annotation (LOC, MISC, ORG, PER, NP – noun phrase, VP – verb phrase, OTHER). The second sub-vector described microposts as a whole. It contained features describing whether all words longer than four characters were capitalized, uppercase, or lowercase. The rest of the sub-vectors were computed according to the overlap of the reference annotation with annotations produced by other NER tools. Such sub-vector (termed a method vector by us) was computed for each extractor and contained four other vectors (average scores per named entity type) for each target entity type (LOC, MISC, ORG, PER). The average score vector consisted of five components--*ail*: the average intersection length of a reference annotation with annotations produced by other extractors (from here on referenced as other's annotations), *aiia*: the average percentage intersection of other's annotations with reference annotation, *aiir*: the average percentage intersection of a reference annotation with other's annotations, *average confidence* (if the underlying extractors return such value), and *variance of the average confidence*. The last component in the training vector was the correct answer (i.e., the correct annotation type taken from manual annotation).

4.2 Model Training

Several types of classification models were considered, especially tree-models which allow the use of numerical and discrete attributes. Due to its large number of trees, Random Forests looked very advisable and reliable during the first round of testing. However, the increasing number of input attributes caused the performance of Random Forests to degrade. Therefore, we used a single decision tree generated by the C4.5 algorithm [5] as a simple alternative. The set of training vectors was preprocessed before the model training. Duplicate rows were removed from the training set and a randomize filter was applied to shuffle the training vectors. The preprocessed training set contained approximately 35,000 vectors, each consisting of 105 attributes. The trained model was represented by a classification tree built by the J48 algorithm in Weka[5]. J48 is also known as an open-source implementation of the C4.5 algorithm with pruning. A Tenfold Fold Cross Validation was used. This model provided classification into five discrete classes (NULL, ORG, LOC, MISC, PER) for each record.

4.3 Estimated Evaluation of the Model

To get an idea of our model performance, we have trained the model on an 80% split of the challenge training set cleaned from duplicate records and have evaluated it on the remaining 20% split. The evaluation results are displayed in Table 1. We included the results from the best individually performing tools for each entity type.

Table 1 Evaluation on the 20% training set split

	Illinois NER			Illinois Wikifier			Stanford NER			Miscinator			Annotowatch		
	P_S	R_S	F_{1S}	P_S	R_S	F_{1S}	P_S	R_S	F_{1S}	P_S	R_S	F_{1S}	P_S	R_S	F_{1S}
LOC	54%	56%	55%	36%	44%	40%	55%	54%	55%	-	-	-	57%	56%	57%
MISC	4%	7%	5%	10%	18%	13%	2%	2%	2%	87%	44%	59%	55%	58%	57%
ORG	31%	35%	33%	60%	41%	49%	23%	28%	25%	-	-	-	64%	49%	56%
PER	86%	84%	85%	89%	56%	69%	83%	78%	81%	-	-	-	85%	87%	86%
ALL	62%	66%	64%	63%	49%	55%	60%	60%	60%	87%	4%	7%	77%	75%	76%

5. Runs Submitted

Three runs were submitted for evaluation in the challenge. The first run was generated by the C4.5 algorithm trained model with parameter M denoting the minimum number of instances per leaf set to 2. The second run was generated by the model trained with parameter M set to 3. The third run was based on the first run and involved specific post-processing. If a micropost identical to one in the training set was annotated, we extended the detected concepts by those from manually annotated training data (affecting three microposts). A gazetteer built from a list of organizations found in the training set has been used to extend the ORG annotations of the model (affecting 69 microposts). The models producing the submission results were trained on a full challenge training set.

6. ACKNOWLEDGMENTS

This work is supported by projects VEGA 2/0185/13, VENIS FP7-284984, CLAN APVV-0809-11 and ITMS: 26240220072

7. REFERENCES

[1] H. Cunningham, D. Maynard, K. Bontcheva, V. Tablan. GATE: A Framework and Graphical Development Environment for Robust NLP Tools and Applications. Proceedings of ACL'02. Philadelphia, July 2002.

[2] L. Ratinov and D. Roth, Design Challenges and Misconceptions in Named Entity Recognition. CoNLL (2009).

[3] L. Ratinov and D. Roth and D. Downey and M. Anderson, Local and Global Algorithms for Disambiguation to Wikipedia. ACL (2011).

[4] Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In ACL 2005, pp. 363-370.

[5] Ross Quinlan (1993). C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Mateo, CA.

[5] http://www.cs.waikato.ac.nz/ml/weka/