Improving Twitter Named Entity Recognition using Word Representations

Zhiqiang Toh, Bin Chen and Jian Su
Institute for Infocomm Research
1 Fusionopolis Way
Singapore 138632
{ztoh,bchen,sujian}@i2r.a-star.edu.sg

Abstract

This paper describes our system used in the ACL 2015 Workshop on Noisy User-generated Text Shared Task for Named Entity Recognition (NER) in Twitter. Our system uses Conditional Random Fields to train two separate classifiers for the two evaluations: predicting 10 fine-grained types, and segmenting named entities. We focus our efforts on generating word representations from large amount of unlabeled newswire data and tweets. Our experiment results show that cluster features derived from word representations significantly improve Twitter NER performances. Our system is ranked 2nd for both evaluations.

1 Introduction

Named Entity Recognition (NER) is the task of identifying and categorizing the various mentions of people, organizations and other named entities within the text. NER has been an essential analysis component in many Natural Language Processing (NLP) systems, especially information extraction and question answering.

Traditionally, the NER system is trained and applied on long and formal text such as the newswire. From the beginning of the new millennium, user-generated content from the social media websites such as Twitter and Weibo presents a huge compilation of informative but noisy and informal text. This rapidly growing text collection becomes more and more important for NLP tasks such as sentiment analysis and emerging topic detection.

However, standard NER system trained on formal text does not work well on this new and challenging style of text. Therefore, adapting the NER system to the new and challenging Twitter domain has attracted increasing attention of researchers. The ACL 2015 Workshop on Noisy User-generated Text (W-NUT) Shared Task for NER in Twitter is organized in response to these new changes (Tim Baldwin, 2015).

We participated in the above Shared Task, which consists of two separate evaluations: one where the task is to predict 10 fine-grained types (10types) and the other in which only named entity segments are predicted (notypes).

For both evaluations, we model the problem as a sequential labeling task, using Conditional Random Fields (CRF) as the training algorithm. An additional postprocessing step is applied to further refine the system output.

The remainder of this paper is structured as follows. In Section 2, we report on the external resources used by our system and how they are obtained and processed. In Section 3, the features used are described in details. In Section 4, the experiment and official results are presented. Finally, Section 5 summarizes our work.

2 External Resources

External resources have shown to improve the performances of Twitter NER (Ritter et al., 2011). Our system uses a variety of external resources, either publicly available, or collected and preprocessed by us.

2.1 Freebase Entity Lists

We use the Freebase entity lists provided by the task organizers. For some of the lists that are not provided (e.g. a list of sports facilities), we manually collect them by calling the appropriate Freebase API.

2.2 Unlabeled Corpora

We gather unlabeled corpora from three different sources: (1) Pre-trained word vectors generated using the GloVe tool (Pennington et al.,

Proceedings of the ACL 2015 Workshop on Noisy User-generated Text, pages 141–145,
Beijing, China, July 31, 2015. ©2015 Association for Computational Linguistics

2014)[1], (2) English Gigaword Fifth Edition[2], and (3) raw tweets collected between the period of March 2015 and April 2015.

For English Gigaword, all articles of *story* type are collected and tokenized. Further preprocessing is performed by following the cleaning step described in Turian et al. (2010). This results in a corpus consisting of 76 million sentences.

The collected raw tweets are tokenized[3] and non-English tweets are removed using `langid.py` (Lui and Baldwin, 2012), resulting in a total of 14 million tweets.

3 Features

This section briefly describes the features used in our system. Besides the features commonly used in traditional NER systems, we focus on the use of word cluster features that have shown to be effective in previous work (Ratinov and Roth, 2009; Turian et al., 2010; Cherry and Guo, 2015).

3.1 Word Feature

The current word and its lowercase format are used as features. To provide additional context information, the previous word and next word (in original format) are also used.

3.2 Orthographic Features

Orthographic features based on regular expressions are often used in NER systems. We only use the following two orthographic features: InitialCap (`[A-Z][a-z].*`) and AllCaps (`[A-Z]+`). In addition, the first character and last two characters of each word are used as features.

3.3 Gazetteer Feature

The current word is matched with entries in the Freebase entity lists and the feature value is the type of entity list matched.

3.4 Word Cluster Features

Unsupervised word representations (e.g. Brown clustering) have shown to improve the performance of NER. Besides brown clusters, we also use clusters generated using the K-means algorithm. These two kinds of clusters are generated from the processed Gigaword and tweet corpora (Section 2.2).

Brown clusters are generated using the implementation by Percy Liang[4]. We experiment with different cluster sizes ($\{100, 200, 500, 1000\}$), resulting in different cluster files for each of the corpora. For each cluster file, different minimum occurrences ($\{5, 10, 20\}$) and binary prefix lengths ($\{4, 6, \cdots, 14, 16\}$) are tested. For each word in the tweet, its corresponding binary prefix string representation is used as the feature value.

K-means clusters are generated using two different methods. The first method uses the `word2vec` tool (Mikolov et al., 2013)[5]. By varying the minimum occurrences ($\{5, 10, 20\}$), word vector size ($\{50, 100, 200, 500, 1000\}$), cluster size ($\{50, 100, 200, 500, 1000\}$) and sub-sampling threshold ($\{0.00001, 0.001\}$), different cluster files are generated and tested. Similar to the Brown cluster feature, the name of the cluster that each word belongs to is used as the feature value.

The second method uses the GloVe tool to generate global vectors for word representation[6]. As the GloVe tool does not output any form of clusters, K-mean clusters are generated from the global vectors using the K-means implementation from Apache Spark MLlib[7]. Similarly, by varying the minimum count ($\{5, 10, 20, 50, 100\}$), window size ($\{5, 10, 15, 20\}$), vector size ($\{50, 100, 200, 500, 1000\}$), and cluster size ($\{50, 100, 200, 500, 1000\}$), different cluster files are generated and tested.

We also generate K-mean cluster files using the pre-trained GloVe word vectors (trained from Wikipedia 2014 and Gigaword Fifth Edition, Common Crawl and Twitter data) in the same manner.

We create a cluster feature for each cluster file that is found to improve the 5-fold cross validation performance. As there are over 800 cluster files, we only test a random subset of cluster files each time and select the best cluster file from the subset to create a new cluster feature. The procedure is repeated for a new subset of cluster files, until no (or negligible) improvement is obtained. Our final settings use one Brown cluster feature and six K-means cluster features (for both 10types and notypes settings).

[1] http://nlp.stanford.edu/projects/glove/

[2] https://catalog.ldc.upenn.edu/LDC2011T07

[3] The tweet tokenization script can be found at https://github.com/myleott/ark-twokenize-py

[4] https://github.com/percyliang/brown-cluster/

[5] https://code.google.com/p/word2vec/

[6] Due to memory constraints, only the tweet corpus is used to generate global vectors.

[7] https://spark.apache.org/mllib/

	10types					
Feature	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Overall
Word Feature	25.76	23.93	27.59	28.01	9.69	23.94
+ Orthographic Features	36.48	35.64	41.20	43.27	25.34	37.03
+ Gazetteer Feature	44.36	43.94	48.22	44.84	30.35	42.94
+ Word Cluster Features	55.85	57.49	60.07	58.35	44.99	55.95
+ Postprocessing	56.09	57.82	60.07	58.88	45.78	56.31

Table 1: 5-fold cross-validation F1 performances for the 10types evaluation. Each row uses all features added in the previous rows.

	notypes					
Feature	Fold 1	Fold 2	Fold 3	Fold 4	Fold 5	Overall
Word Feature	30.91	30.08	33.41	36.99	20.69	31.09
+ Orthographic Features	52.06	54.29	52.22	53.11	44.49	51.62
+ Gazetteer Feature	52.26	56.70	58.78	56.74	47.45	54.77
+ Word Cluster Features	65.14	65.57	66.77	68.13	55.31	64.66
+ Postprocessing	65.44	65.85	67.30	68.70	56.00	65.13

Table 2: 5-fold cross-validation F1 performances for the notypes evaluation. Each row uses all features added in the previous rows.

4 Experiments and Results

Our system is trained using the CRF++ tool[8]. We trained separate classifiers for the two different evaluations (10types and notypes).

To select the optimum settings, we make use of all available training data (`train`, `dev`, `dev_2015`) and conduct 5-fold cross validation experiments. For easier comparisons with other systems, the 5 folds are split such that `dev` is the test set for Fold 1, while `dev_2015` is the test set for Fold 5.

4.1 Preliminary Results on Training Data

Table 1 and Table 2 shows the 5-fold cross validation performances after adding each feature group for the 10types and notypes evaluations respectively. The use of word clusters significantly improves the performances for both evaluations. There is an overall improvement of 13% and 9% for the 10types and notypes evaluation respectively when word cluster features are added. This demonstrates the usefulness of word vectors in improving the accuracy of a Twitter NER system.

Comparing the performances of Fold 1 (tested on `dev`) and Fold 5 (tested on `dev_2015`), we observe a significant performance difference. Similar observations can also be seen for the other three folds (tested on a subset of `train`) when compared with Fold 5. This suggests that there are notable differences between the data provided during the training period (`train` and `dev`) and evaluation period (`dev_2015`), probably because the two sets of data are collected in different time periods.

4.2 Postprocessing

We also experiment with a postprocessing step based on heuristic rules to further refine the system output (last row of Table 1 and Table 2). The heuristic rules are based on string matching of words with name list entries. To prevent false positives, we require entries in some of the name lists to contain at least two words and should not contain common words/stop words. For certain name lists where single-word entries are common but ambiguous (e.g. name of sports clubs), we check for the presence of cue words in the tweet before matching. For example, for single-word sport team names that are common in tweets, we check for the presence of cue words such as “vs”. Examples of name lists used include names of professional athletes, music composers and sport facilities.

[8] http://taku910.github.io/crfpp/

	10types				notypes			
System	Rank	Precision	Recall	F1	Rank	Precision	Recall	F1
NLANGP	2	63.62	43.12	51.40	2	67.74	54.31	60.29
1st	1	57.66	55.22	56.41	1	72.20	69.14	70.63
2nd	2	63.62	43.12	51.40	2	67.74	54.31	60.29
3rd	3	53.24	38.58	44.74	3	63.81	56.28	59.81
Baseline	–	35.56	29.05	31.97	–	53.86	46.44	49.88

Table 3: Comparison of our system (NLANGP) with the top three participating systems and official baselines for the 10types and notypes evaluations.

	10types			notypes		
System	Precision	Recall	F1	Precision	Recall	F1
NLANGP	63.62	43.12	51.40	67.74	54.31	60.29
- Word Cluster Features	57.99	25.26	35.19	62.56	38.43	47.61

Table 4: System performances on the test data when word cluster features are not used.

4.3 Evaluation Results

Table 3 presents the official results of our 10types and notypes submissions. We also include the results of the top three participating systems and official baselines for comparison.

As shown from the table, our system (NLANGP) is ranked 2nd for both evaluations. Based on our preliminary Fold 5 performances, our system performances on the test data (`test_2015`, collected in the same period as `dev_2015`) are within expectation. In general, the fine-grained evaluation is a more challenging task, as seen from the huge performance difference between the F1 score of 10types and notypes.

Type	Precision	Recall	F1
COMPANY	80.00	41.03	54.24
FACILITY	52.17	31.58	39.34
GEO-LOC	63.81	57.76	60.63
MOVIE	100.00	33.33	50.00
MUSICARTIST	50.00	9.76	16.33
OTHER	50.00	30.30	37.74
PERSON	70.70	64.91	67.68
PRODUCT	20.00	8.11	11.54
SPORTSTEAM	79.41	38.57	51.92
TVSHOW	0.00	0.00	0.00
Overall	63.62	43.12	51.40

Table 5: Performance of each fine-grained type of our system.

Table 5 shows the performance of each fine-grained type of our system. Unlike traditional NER where state-of-the-art systems can achieve performances over 90 F1 for the 3 MUC types (PERSON, LOCATION and ORGANIZATION), Twitter NER poses new challenges in accurately extracting entity information in such genre that does not exist in the past.

We are interested to know the performance contribution of the word clusters on the test data. Table 4 shows the performances on the test data when word cluster features are not used. Similarly to the observations observed in the training data, word clusters are important features for our system: a performance drop greater than 16% and 12% is observed for the 10types and notypes evaluation respectively.

5 Conclusion

In this paper, we describe our system used in the W-NUT Shared Task for NER in Twitter. We focus our efforts on improving Twitter NER using word representations, namely, Brown clusters and K-means clusters, that are generated from large amount of unlabeled newswire data and tweets. Our experiments and evaluation results show that cluster features derived from word representations are effective in improving Twitter NER performances. In future, we hope to investigate on the use of distant supervision learning technique to build better system that can perform more robustly across tweets from different time periods. We also like to perform an error analysis to help us understand which other problems persist so as to address them in future.

References

Colin Cherry and Hongyu Guo. 2015. The Unreasonable Effectiveness of Word Representations for Twitter Named Entity Recognition. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 735–745, Denver, Colorado, May–June. Association for Computational Linguistics.

Marco Lui and Timothy Baldwin. 2012. langid.py: An Off-the-shelf Language Identification Tool. In *Proceedings of the ACL 2012 System Demonstrations*, pages 25–30, Jeju Island, Korea, July. Association for Computational Linguistics.

Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. 2013. Linguistic Regularities in Continuous Space Word Representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia, June. Association for Computational Linguistics.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, October. Association for Computational Linguistics.

Lev Ratinov and Dan Roth. 2009. Design Challenges and Misconceptions in Named Entity Recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155, Boulder, Colorado, June. Association for Computational Linguistics.

Alan Ritter, Sam Clark, Mausam, and Oren Etzioni. 2011. Named Entity Recognition in Tweets: An Experimental Study. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 1524–1534, Edinburgh, Scotland, UK., July. Association for Computational Linguistics.

Marie Marie Catherine de Marneffe Young-Bum Kim Alan Ritter Wei Xu Tim Baldwin, Bo Han. 2015. Findings of the 2015 Workshop on Noisy User-generated Text. In *Proceedings of the Workshop on Noisy User-generated Text (WNUT 2015)*. Association for Computational Linguistics.

Joseph Turian, Lev-Arie Ratinov, and Yoshua Bengio. 2010. Word Representations: A Simple and General Method for Semi-Supervised Learning. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 384–394, Uppsala, Sweden, July. Association for Computational Linguistics.