Enhancing Named Entity Recognition in Twitter Messages Using Entity Linking

Ikuya Yamada[1 2 3] **Hideaki Takeda**[2] **Yoshiyasu Takefuji**[3]
ikuya@ousia.jp takeda@nii.ac.jp takefuji@sfc.keio.ac.jp
[1]Studio Ousia, 4489-105-221 Endo, Fujisawa, Kanagawa, Japan
[2]National Institute of Informatics, 2-1-2 Hitotsubashi, Chiyoda, Tokyo, Japan
[3]Keio University, 5322 Endo, Fujisawa, Kanagawa, Japan

Abstract

In this paper, we describe our approach for *Named Entity Recognition in Twitter*, a shared task for ACL 2015 Workshop on Noisy User-generated Text (Baldwin et al., 2015). Because of the noisy, short, and colloquial nature of Twitter, the performance of Named Entity Recognition (NER) degrades significantly. To address this problem, we propose a novel method to enhance the performance of the Twitter NER task by using *Entity Linking* which is a method for detecting entity mentions in text and resolving them to corresponding entries in knowledge bases such as Wikipedia. Our method is based on supervised machine-learning and uses the high-quality knowledge obtained from several open knowledge bases. In comparison with the other systems proposed for this shared task, our method achieved the best performance.

1 Introduction

Named Entity Recognition (NER) refers to the task of identifying mentions of entities (e.g., persons, locations, organizations) within text. Because of the noisy, short, and colloquial nature of Twitter messages (or *tweets*), the performance of standard NER software significantly suffers. For example, Derczynski et al. (Derczynski et al., 2015) recently demonstrated that the performance of various state-of-the-art NER software (e.g., Stanford NER and ANNIE) is typically lower than 50% F1[1] for tweets.

Entity Linking (EL) refers to the task of detecting textual entity mentions and linking them to corresponding entries within knowledge bases (e.g., Wikipedia, DBpedia (Auer et al., 2007), Freebase (Bollacker et al., 2008)). Because of the recent emergence of large online knowledge bases (KB), EL has recently gained significant attention. It is evident that the performance of EL also degrades when analyzing tweets (Derczynski et al., 2015; Meij et al., 2012). However, Guo et al. (Guo et al., 2013) recently revealed that the main failures of Twitter EL are caused while detecting entity mentions from text, because existing EL methods usually address the mention detection task by using external NER software whose performance is unreliable when processing tweets. Consequently, several approaches (Guo et al., 2013; Yamada et al., 2015) have been proposed with enhanced abilities that address the task in an *end-to-end* manner without completely depending on NER software.

The main objective of this study is to investigate the possibility of enhancing the performance of Twitter NER by using an end-to-end EL. Although EL is typically performed *after* NER in most of the existing methods, our approach performs EL *before* NER and uses the EL results to enhance the NER performance. Resolving the entity mentions to the KB entries enables us to use the high-quality knowledge in KB for enhancing the NER performance. This knowledge includes things such as the popularity of the entity, the classes of the entity, and the likelihood that the entity appears in the given context.

We begin by briefly introducing our end-to-end EL method that specifically focuses on tweets. Our EL method is based on supervised machine-learning and addresses the task in an end-to-end manner. It considers every possible n-gram as a candidate entity mention and detects the mention with a corresponding link to a KB entry if the mention exists in the KB. Furthermore, it can handle mentions that appear as irregular forms (e.g., *misspellings*, *abbreviations*, *acronyms)* using several approximate string matching algorithms.

[1] The harmonic mean of precision and recall.

The NER task is split into two separate subtasks: *segmentation* and *classification*. During segmentation, the entity mentions are detected from tweets. Then, the entity mentions are classified into the predefined entity types. Both tasks involve supervised machine-learning with various features.

For the segmentation task, we use data obtained from the KB of the corresponding entity mention detected by the EL and the output of a NER software as the main machine-learning features. Furthermore, we include several common features used in traditional NER methods.

For the classification task, the following three types of features are used as primary features: 1) the KB types of the entity detected by the EL, 2) the entity types detected by the NER software, and 3) the vector representation of the entity mention derived from word embeddings. The entity's KB types are extracted from the corresponding entries in DBpedia and Freebase. Furthermore, the vector representation of the entity mention is derived using GloVe word embeddings (Pennington et al., 2014).

To train and evaluate our system, we used the dataset given by the *Named Entity Recognition in Twitter* shared task. Our proposed method significantly outperformed the second ranked system by a wide margin; *10.3% F1* at the segmentation task, and *5.0% F1* at the end-to-end (both the segmentation and the classification) task.

2 The Proposed System

2.1 Preprocessing

The system first assigns part-of-speech tags to the resulting tokens using ARK Twitter Part-of-Speech Tagger (Gimpel et al., 2011). It also tokenizes Twitter hashtags using our enhanced implementation of the hashtag tokenization.

2.2 Entity Linking

We formalize our EL task as follows: Given a tweet, our goal is to recognize a set of entity mentions (e.g., *Obama*, *President Obama*, *Barack Obama*) that appear in a tweet, and then resolve the mentions into entities (e.g., Barack Obama) in Wikipedia if they exist.

Our EL system addresses the task using the following two steps; *mention candidate generation* and *mention detection and disambiguation*.

2.2.1 Mention Candidate Generation

Our system first generates a set of candidate entity mentions with the set of corresponding referent entities. The system takes all the n-grams of $n \leq 10$ and looks up each n-gram in a dictionary, treats an n-gram as a candidate mention if it exists in the dictionary, and finally, generates an output of pairs of mentions and their associated possible referent entities.

Mention-Entity Dictionary: The system uses a *mention-entity* dictionary that maps a mention surface (e.g., *apple*) to the possible referent entities (e.g., Apple Inc., Apple (food)). The possible mention surfaces of an entity are extracted from the corresponding Wikipedia page title, the page titles of the Wikipedia pages that redirect to the page of the entity, and anchor texts in Wikipedia articles that point to the page of the entity. We constructed this dictionary using the January 2015 dump of Wikipedia.

Approximate Candidate Generation: One major problem of the mention candidate generation task is that many entity mentions in tweets cannot be detected because they appear as irregular forms (e.g., *misspellings*, *abbreviations*). In order to address this problem, we introduce the following three approximate string-matching methods to improve the ability of this task:

- *Fuzzy match* searches the mention candidates that have text surfaces within a certain distance of the surface of the n-gram measured by edit distance.
- *Approximate token search* obtains mention candidates whose text surfaces have a significant ratio of words in common with the surface of the n-gram.
- *Acronym search* retrieves mention candidates with possible acronyms[2] that include the surface of the n-gram.

When using the above methods, we observed that the number of mention candidates becomes very large. To deal with this, we use a simple filtering method based on *soft tf-idf* (Cohen et al., 2003); we simply use only the mention candidates that have a similarity greater than a threshold measured by the soft tf-idf. We use 0.9 as the threshold

[2] We generate acronyms by tokenizing the mention surface and simply taking the first characters of the resulting tokens.

because this achieves the best performance in our experiments of EL.

2.2.2 Mention Detection and Disambiguation

Given a pair of a mention and its possible referent entity, it needs to be determined if the possible referent entity is indeed the correct one for its associated mention.

In this system, we use a supervised machine-learning algorithm to assign a relevance score to each of the pairs and select the entity mention with the highest score. We use *random forest* as the machine-learning algorithm.

Here, we use machine-learning features that are mostly identical to the method proposed previously (Yamada et al., 2015). Basically, we use various features that are commonly observed in EL studies and enhance the performance further by introducing two new features: 1) the entity popularity knowledge extracted from Wikipedia page views[3], and 2) the contextual similarity between the entity and the tweet measured by word embeddings.

2.3 Named Entity Recognition

We address the NER task by performing two subtasks: *segmentation* and *classification*.

2.3.1 Segmentation of Named Entities

In this step, entity mentions are detected from tweets. We formalize this task as follows. Given an n-gram in a tweet, the goal of this task is assigning a binary label that represents whether the n-gram should be detected as an entity mention. Note that in order to enable the straightforward integration of EL and this task, we formalize this task as simply classifying n-grams instead of the commonly-used IOB labeling approach (Ramshaw and Marcus, 1995).

The basic strategy that we adopt here is to combine the output of NER software and the KB knowledge of the corresponding entity mention detected by the EL using supervised machine-learning. We again use *random forest* as the machine-learning algorithm.

We use Stanford NER[4] as the NER software that achieves relatively better performance in the Twitter NER task in a recent study (Derczynski et al., 2015). Here, we adopt two models of Stanford NER to enhance the performance: 1) the standard three-class model which is included in the software and 2) a model that does not use capitalization as a feature, in order to deal with the unreliability of capitalization in tweets.

The results of the NER and the KB knowledge of the corresponding entity mention detected by the EL are used as the primary machine-learning features. We also include features that are traditionally used in NER such as part-of-speech tags and the capitalization features. Furthermore, the ratio of the capitalized words in the tweet is also used as an indicator of the reliability of the capitalization.

The machine-learning features for this step include:

- *EL relevance score**: The relevance score of the entity mention assigned by the previous EL step.
- *Link probability**: The probability of the entity mention appearing as an anchor text in Wikipedia.
- *Capitalization probability**: The probability of the entity mention being capitalized in Wikipedia.
- *The number of inbound links**: The number of inbound links of the corresponding entity in Wikipedia.
- *The average page view**: The average page view of the corresponding entity in Wikipedia.
- *NER span match*: Binary values that represent whether the n-gram is detected by NER models.
- *Part-of-speech tags*: Part-of-speech tags of the previous, first, last, and next words of the n-gram.
- *Context capitalization*: Binary values that represent whether the previous, first, last, and next words of the n-gram are capitalized.
- *Character length*: The number of characters read in the surface of the n-gram.
- *Token length*: The number of tokens read in the n-gram.

Note that some features (marked with *) are based on an entity mention detected by EL, thus

[3] http://dumps.wikimedia.org/other/pagecounts-raw/

[4] http://nlp.stanford.edu/software/CRF-NER.shtml

these features can be missing if there is no corresponding entity mention detected by the EL.

We also resolve overlaps of mentions by iteratively selecting the longest entity mention from the beginning of a tweet.

2.3.2 Classification of Named Entities

In this step, detected entity mentions are classified into the predefined types (i.e., person, geo-loc, facility, product, company, movie, sportsteam, musicartist, tvshow, and other) using supervised machine-learning. Here, linear support vector machine is used as the machine-learning model.

One main machine-learning feature of this step is the corresponding entity types retrieved from KBs. We obtain KB entity types from the corresponding entries in DBpedia[5] and Freebase[6].

One problem in this step is that there are several entity mentions that cannot be detected by EL because of various reasons (e.g., a non-existent entity in the KB, an error performing EL). In addition, some minor entities might not have entity types in the KBs. In order to deal with this problem, we first include the entity types predicted by Stanford NER as features. However, because the target entity types of our task do not directly correspond to the ones given in Stanford NER (i.e., location, person, and organization), the effectiveness of these features is obviously limited. Therefore, we introduce another type of feature based on word embeddings. For this, we use GloVe word embeddings[7] to calculate an average vector of vectors of words in n-gram text.

We also include the relevance score assigned by the previous EL step that indicates the reliability of the KB entity types to the model. The number of words and the number of characters in the n-gram text are also included as features to enhance the expressiveness of our model even further.

The machine-learning features for this step include:

- *KB entity types*: The entity types in KBs. The KBs used include DBpedia and Freebase.
- *NER detected type*: The detected entity types of the NER model. As mentioned in Section 2.3.1, we use two different models of Stanford NER.
- *N-gram vector*: The vector representation of the n-gram derived using the method explained above and includes each dimension of the vector as a separate feature.
- *EL relevance score*: The relevance score assigned by the previous EL step.
- *Character length*: The number of characters read in the n-gram text.
- *Token length*: The number of tokens read in the n-gram.

System Name	Precision	Recall	F1
Our Method	**72.20%**	**69.14%**	**70.63%**
NLANGP	67.74%	54.31%	60.29%
USFD	63.81%	56.28%	59.81%
multimedialab	62.93%	55.22%	58.82%
nrc	62.13%	54.61%	58.13%

Table 1: Performances of the proposed systems at segmenting entities

3 Experiments

3.1 Experimental Setup

To train our proposed EL method, we used the #Microposts 2015 EL dataset (Rizzo et al., 2015) that contains *3,998* tweets and *3,993* annotations of entities.[8] The performance of our EL method using this particular dataset is reported in (Yamada et al., 2015).

For this shared task, we trained and evaluated our proposed Twitter NER using the dataset provided by the workshop.[9]

3.2 Results

Table 1 shows the results of the segmentation task of the five top-ranking systems. Our proposed method significantly outperforms the second ranked method by 10.3% F1.

The end-to-end results (both segmentation and classification tasks) of the five top-ranking systems are shown in Table 2. Here, our method significantly outperforms the second ranked method by 5.0% F1. Table 3 also presents detailed scores broken down by entity types.

[5] http://mappings.dbpedia.org/server/ontology/classes/

[6] http://wiki.freebase.com/wiki/Type

[7] We use the 300-dimensional model generated using 840B tokens obtained from CommonCrawl corpus. http://nlp.stanford.edu/projects/glove/

[8] We use the *training* and the *dev* set of the #Microposts 2015 dataset as the training data.

[9] We use the *train*, the *dev*, and the *dev_2015* set for training the NER model.

System Name	Precision	Recall	F1
Our Method	57.66%	**55.22%**	**56.41%**
NLANGP	**63.62%**	41.12%	51.40%
nrc	53.24%	38.58%	44.74%
multimedialab	49.52%	39.18%	43.75%
USFD	45.72%	39.64%	42.46%

Table 2: Performances of the proposed systems at both segmentation and classification tasks

Entity Type	Precision	Recall	F1
company	41.82%	58.97%	48.94%
facility	50.00%	26.32%	34.48%
geo-loc	57.59%	78.45%	66.42%
movie	66.67%	40.00%	50.00%
musicartist	70.00%	34.15%	45.90%
other	47.06%	42.42%	44.62%
person	70.97%	77.19%	73.95%
product	34.78%	21.62%	26.67%
sportsteam	66.67%	34.29%	45.28%
tvshow	14.29%	50.00%	22.22%

Table 3: Performance of our system at both segmentation and classification tasks broken down by entity types

4 Conclusions

In this paper, we proposed a novel method for the Twitter NER task. We showed that the data retrieved from open knowledge bases (i.e., Wikipedia, DBpedia, Freebase) can be naturally leveraged to enhance NER using *entity linking*. Furthermore, this data appears to be highly effective for both the *segmentation* and the *classification* tasks.

References

Sören Auer, Christian Bizer, Georgi Kobilarov, Jens Lehmann, Richard Cyganiak, and Zachary Ives. 2007. DBpedia: a nucleus for a web of open data. *The Semantic Web*, pages 722–735.

Timothy Baldwin, Marie Catherine de Marneffe, Bo Han, Young-Bum Kim, Alan Ritter, and Wei Xu. 2015. Shared tasks of the 2015 workshop on noisy user-generated text: Twitter lexical normalization and named entity recognition. In *Proceedings of the Workshop on Noisy User-generated Text (WNUT 2015)*.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD International Conference on Management of Data (SIGMOD '08)*, pages 1247–1250.

William W. Cohen, Pradeep D. Ravikumar, and Stephen E. Fienberg. 2003. A comparison of string distance metrics for name-matching tasks. In *Proceedings of IJCAI-03 Workshop on Information Integration on the Web*, pages 73–78.

Leon Derczynski, Diana Maynard, Giuseppe Rizzo, Marieke van Erp, Genevieve Gorrell, Raphaël Troncy, Johann Petrak, and Kalina Bontcheva. 2015. Analysis of named entity recognition and linking for tweets. *Information Processing & Management*, 51(2):32–49.

Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A Smith. 2011. Part-of-speech tagging for Twitter: annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (HLT '11)*, pages 42–47.

Stephen Guo, Ming-Wei Chang, and Emre Kiciman. 2013. To link or not to link? a study on end-to-end Tweet entity linking. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT '13)*, pages 1020–1030.

Edgar Meij, Wouter Weerkamp, and Maarten de Rijke. 2012. Adding semantics to microblog posts. In *Proceedings of the Fifth ACM International Conference on Web Search and Data Mining (WSDM '12)*, pages 563–572.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP '14)*, pages 1532–1543.

Lance A. Ramshaw and Mitchell P. Marcus. 1995. Text chunking using transformation-based learning. In *Proceedings of ACL Third Workshop on Very Large Corpora*, pages 82–94.

Giuseppe Rizzo, Amparo Elizabeth Cano Basave, Bianca Pereira, Andrea Varga, Matthew Rowe, Milan Stankovic, and Aba-Sah Dadzie. 2015. Making sense of microposts (#Microposts2015) named entity recognition and linking (NEEL) challenge. In *5th Workshop on Making Sense of Microposts (#Microposts2015)*.

Ikuya Yamada, Hideaki Takeda, and Yoshiyasu Takefuji. 2015. An end-to-end entity linking approach for Tweets. In *5th Workshop on Making Sense of Microposts (#Microposts 2015)*.