Recognizing Named Entities in Tweets

Xiaohua Liu [‡] [†], **Shaodian Zhang**[*] [§], **Furu Wei** [†], **Ming Zhou** [†]
[‡]School of Computer Science and Technology
Harbin Institute of Technology, Harbin, 150001, China
[§]Department of Computer Science and Engineering
Shanghai Jiao Tong University, Shanghai, 200240, China
[†]Microsoft Research Asia
Beijing, 100190, China
[†]{xiaoliu, fuwei, mingzhou}@microsoft.com
[§] zhangsd.sjtu@gmail.com

Abstract

The challenges of Named Entities Recognition (NER) for tweets lie in the insufficient information in a tweet and the unavailability of training data. We propose to combine a K-Nearest Neighbors (KNN) classifier with a linear Conditional Random Fields (CRF) model under a semi-supervised learning framework to tackle these challenges. The KNN based classifier conducts pre-labeling to collect global coarse evidence across tweets while the CRF model conducts sequential labeling to capture fine-grained information encoded in a tweet. The semi-supervised learning plus the gazetteers alleviate the lack of training data. Extensive experiments show the advantages of our method over the baselines as well as the effectiveness of KNN and semi-supervised learning.

1 Introduction

Named Entities Recognition (NER) is generally understood as the task of identifying mentions of rigid designators from text belonging to named-entity types such as persons, organizations and locations (Nadeau and Sekine, 2007). Proposed solutions to NER fall into three categories: 1) The rule-based (Krupka and Hausman, 1998); 2) the machine learning based (Finkel and Manning, 2009; Singh et al., 2010) ; and 3) hybrid methods (Jansche and Abney, 2002). With the availability of annotated corpora, such as ACE05, Enron (Minkov et al., 2005) and CoNLL03 (Tjong Kim Sang and De Meulder, 2003), the data driven methods now become the dominating methods.

However, current NER mainly focuses on formal text such as news articles (Mccallum and Li, 2003; Etzioni et al., 2005). Exceptions include studies on informal text such as emails, blogs, clinical notes (Wang, 2009). Because of the domain mismatch, current systems trained on non-tweets perform poorly on tweets, a new genre of text, which are short, informal, ungrammatical and noise prone. For example, the average F1 of the Stanford NER (Finkel et al., 2005) , which is trained on the CoNLL03 shared task data set and achieves state-of-the-art performance on that task, drops from 90.8% (Ratinov and Roth, 2009) to 45.8% on tweets.

Thus, building a domain specific NER for tweets is necessary, which requires a lot of annotated tweets or rules. However, manually creating them is tedious and prohibitively unaffordable. Proposed solutions to alleviate this issue include: 1) Domain adaption, which aims to reuse the knowledge of the source domain in a target domain. Two recent examples are Wu et al. (2009), which uses data that is informative about the target domain and also easy to be labeled to bridge the two domains, and Chiticariu et al. (2010), which introduces a high-level rule language, called NERL, to build the general and domain specific NER systems; and 2) semi-supervised learning, which aims to use the abundant unlabeled data to compensate for the lack of annotated data. Suzuki and Isozaki (2008) is one such example.

Another challenge is the limited information in tweet. Two factors contribute to this difficulty. One

[*] This work has been done while the author was visiting Microsoft Research Asia.

is the tweet's informal nature, making conventional features such as part-of-speech (POS) and capitalization not reliable. The performance of current NLP tools drops sharply on tweets. For example, OpenNLP [1], the state-of-the-art POS tagger, gets only an accuracy of 74.0% on our test data set. The other is the tweet's short nature, leading to the excessive abbreviations or shorthand in tweets, and the availability of very limited context information. Tackling this challenge, ideally, requires adapting related NLP tools to fit tweets, or normalizing tweets to accommodate existing tools, both of which are hard tasks.

We propose a novel NER system to address these challenges. Firstly, a K-Nearest Neighbors (KNN) based classifier is adopted to conduct word level classification, leveraging the similar and recently labeled tweets. Following the two-stage prediction aggregation methods (Krishnan and Manning, 2006), such pre-labeled results, together with other conventional features used by the state-of-the-art NER systems, are fed into a linear Conditional Random Fields (CRF) (Lafferty et al., 2001) model, which conducts fine-grained tweet level NER. Furthermore, the KNN and CRF model are repeatedly retrained with an incrementally augmented training set, into which high confidently labeled tweets are added. Indeed, it is the combination of KNN and CRF under a semi-supervised learning framework that differentiates ours from the existing. Finally, following Lev Ratinov and Dan Roth (2009), 30 gazetteers are used, which cover common names, countries, locations, temporal expressions, etc. These gazetteers represent general knowledge across domains. The underlying idea of our method is to combine global evidence from KNN and the gazetteers with local contextual information, and to use common knowledge and unlabeled tweets to make up for the lack of training data.

12,245 tweets are manually annotated as the test data set. Experimental results show that our method outperforms the baselines. It is also demonstrated that integrating KNN classified results into the CRF model and semi-supervised learning considerably boost the performance.

Our contributions are summarized as follows.

1. We propose to a novel method that combines a KNN classifier with a conventional CRF based labeler under a semi-supervised learning framework to combat the lack of information in tweet and the unavailability of training data.

2. We evaluate our method on a human annotated data set, and show that our method outperforms the baselines and that both the combination with KNN and the semi-supervised learning strategy are effective.

The rest of our paper is organized as follows. In the next section, we introduce related work. In Section 3, we formally define the task and present the challenges. In Section 4, we detail our method. In Section 5, we evaluate our method. Finally, Section 6 concludes our work.

2 Related Work

Related work can be roughly divided into three categories: NER on tweets, NER on non-tweets (e.g., news, bio-logical medicine, and clinical notes), and semi-supervised learning for NER.

2.1 NER on Tweets

Finin et al. (2010) use Amazons Mechanical Turk service [2] and CrowdFlower [3] to annotate named entities in tweets and train a CRF model to evaluate the effectiveness of human labeling. In contrast, our work aims to build a system that can automatically identify named entities in tweets. To achieve this, a KNN classifier with a CRF model is combined to leverage cross tweets information, and the semi-supervised learning is adopted to leverage unlabeled tweets.

2.2 NER on Non-Tweets

NER has been extensively studied on formal text, such as news, and various approaches have been proposed. For example, Krupka and Hausman (1998) use manual rules to extract entities of predefined types; Zhou and Ju (2002) adopt Hidden Markov Models (HMM) while Finkel et al. (2005) use CRF to train a sequential NE labeler, in which the BIO (meaning Beginning, the Inside and the Outside of

[1] http://sourceforge.net/projects/opennlp/

[2] https://www.mturk.com/mturk/

[3] http://crowdflower.com/

an entity, respectively) schema is applied. Other methods, such as classification based on Maximum Entropy models and sequential application of Perceptron or Winnow (Collins, 2002), are also practiced. The state-of-the-art system, e.g., the Stanford NER, can achieve an F1 score of over 92.0% on its test set.

Biomedical NER represents another line of active research. Machine learning based systems are commonly used and outperform the rule based systems. A state-of-the-art biomedical NER system (Yoshida and Tsujii, 2007) uses lexical features, orthographic features, semantic features and syntactic features, such as part-of-speech (POS) and shallow parsing.

A handful of work on other domains exists. For example, Wang (2009) introduces NER on clinical notes. A data set is manually annotated and a linear CRF model is trained, which achieves an F-score of 81.48% on their test data set; Downey et al. (2007) employ capitalization cues and n-gram statistics to locate names of a variety of classes in web text; most recently, Chiticariu et al. (2010) design and implement a high-level language NERL that is tuned to simplify the process of building, understanding, and customizing complex rule-based named-entity annotators for different domains.

Ratinov and Roth (2009) systematically study the challenges in NER, compare several solutions and report some interesting findings. For example, they show that a conditional model that does not consider interactions at the output level performs comparably to beam search or Viterbi, and that the BILOU (Beginning, the Inside and the Last tokens of multi-token chunks as well as Unit-length chunks) encoding scheme significantly outperforms the BIO schema (Beginning, the Inside and Outside of a chunk).

In contrast to the above work, our study focuses on NER for tweets, a new genre of texts, which are short, noise prone and ungrammatical.

2.3 Semi-supervised Learning for NER

Semi-supervised learning exploits both labeled and un-labeled data. It proves useful when labeled data is scarce and hard to construct while unlabeled data is abundant and easy to access.

Bootstrapping is a typical semi-supervised learning method. It iteratively adds data that has been confidently labeled but is also informative to its training set, which is used to re-train its model. Jiang and Zhai (2007) propose a balanced bootstrapping algorithm and successfully apply it to NER. Their method is based on instance re-weighting, which allows the small amount of the bootstrapped training sets to have an equal weight to the large source domain training set. Wu et al. (2009) propose another bootstrapping algorithm that selects bridging instances from an unlabeled target domain, which are informative about the target domain and are also easy to be correctly labeled. We adopt bootstrapping as well, but use human labeled tweets as seeds.

Another representative of semi-supervised learning is learning a robust representation of the input from unlabeled data. Miller et al. (2004) use word clusters (Brown et al., 1992) learned from unlabeled text, resulting in a performance improvement of NER. Guo et al. (2009) introduce Latent Semantic Association (LSA) for NER. In our pilot study of NER for tweets, we adopt bag-of-words models to represent a word in tweet, to concentrate our efforts on combining global evidence with local information and semi-supervised learning. We leave it to our future work to explore which is the best input representation for our task.

3 Task Definition

We first introduce some background about tweets, then give a formal definition of the task.

3.1 The Tweets

A tweet is a short text message containing no more than 140 characters in Twitter, the biggest micro-blog service. Here is an example of tweets: “mycraftingworld: #Win Microsoft Office 2010 Home and Student *2Winners* #Contest from @office and @momtobedby8 #Giveaway http://bit.ly/bCsLOr ends 11/14”, where ”mycraftingworld” is the name of the user who published this tweet. Words beginning with the “#” character, like “”#Win”, “#Contest” and “#Giveaway”, are hash tags, usually indicating the topics of the tweet; words starting with “@”, like “@office” and “@momtobedby8”, represent user names, and “http://bit.ly/bCsLOr” is a shortened link.

Twitter users are interested in named entities, such

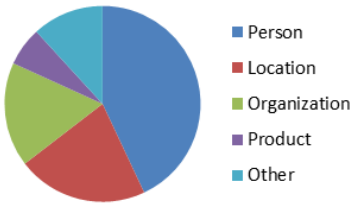

Figure 1: Portion of different types of named entities in tweets. This is based on an investigation of 12,245 randomly sampled tweets, which are manually labeled.

as person names, organization names and product names, as evidenced by the abundant named entities in tweets. According to our investigation on 12,245 randomly sampled tweets that are manually labeled, about 46.8% have at least one named entity. Figure 1 shows the portion of named entities of different types.

3.2 The Task

Given a tweet as input, our task is to identify both the boundary and the class of each mention of entities of predefined types. We focus on four types of entities in our study, i.e., persons, organizations, products, and locations, which, according to our investigation as shown in Figure 1, account for 89.0% of all the named entities.

Here is an example illustrating our task. The input is "...Me without you is like an iphone without apps, Justin Bieber without his hair, Lady gaga without her telephone, it just wouldn..." The expected output is as follows:"...Me without you is like an <PRODUCT >iphone</PRODUCT>without apps, <PERSON>Justin Bieber</PERSON>without his hair,<PERSON>Lady gaga</PERSON> without her telephone, it just wouldn...", meaning that "iphone" is a product, while "Justin Bieber" and "Lady gaga" are persons.

4 Our Method

Now we present our solution to the challenging task of NER for tweets. An overview of our method is first given, followed by detailed discussion of its core components.

4.1 Method Overview

NER task can be naturally divided into two subtasks, i.e., boundary detection and type classification. Following the common practice , we adopt a sequential labeling approach to jointly resolve these sub-tasks, i.e., for each word in the input tweet, a label is assigned to it, indicating both the boundary and entity type. Inspired by Ratinov and Roth (2009), we use the BILOU schema.

Algorithm 1 outlines our method, where: $train_s$ and $train_k$ denote two machine learning processes to get the CRF labeler and the KNN classifier, respectively; $repr_w$ converts a word in a tweet into a bag-of-words vector; the $repr_t$ function transforms a tweet into a feature matrix that is later fed into the CRF model; the knn function predicts the class of a word; the $update$ function applies the predicted class by KNN to the inputted tweet; the crf function conducts word level NE labeling;τ and γ represent the minimum labeling confidence of KNN and CRF, respectively, which are experimentally set to 0.1 and 0.001; N (1,000 in our work) denotes the maximum number of new accumulated training data.

Algorithm 1 NER for Tweets.

Require: Tweet stream i; output stream o.
Require: Training tweets ts; gazetteers ga.

1: Initialize l_s, the CRF labeler: $l_s = train_s(ts)$.
2: Initialize l_k, the KNN classifier: $l_k = train_k(ts)$.
3: Initialize n, the # of new training tweets: $n = 0$.
4: **while** Pop a tweet t from i and $t \neq null$ **do**
5: **for** Each word $w \in t$ **do**
6: Get the feature vector $\vec{w}$: $\vec{w} = repr_w(w, t)$.
7: Classify $\vec{w}$ with knn: $(c, cf) = knn(l_k, \vec{w})$.
8: **if** $cf > \tau$ **then**
9: Pre-label: $t = update(t, w, c)$.
10: **end if**
11: **end for**
12: Get the feature vector $\vec{t}$: $\vec{t} = repr_t(t, ga)$.
13: Label $\vec{t}$ with crf: $(t, cf) = crf(l_s, \vec{t})$.
14: Put labeled result (t, cf) into o.
15: **if** $cf > \gamma$ **then**
16: Add labeled result t to ts , $n = n + 1$.
17: **end if**
18: **if** $n > N$ **then**
19: Retrain l_s: $l_s = train_s(ts)$.
20: Retrain l_k: $l_k = train_k(ts)$.
21: $n = 0$.
22: **end if**
23: **end while**
24: **return** o.

Our method, as illustrated in Algorithm 1, repeatedly adds the new confidently labeled tweets to the training set [4] and retrains itself once the number of new accumulated training data goes above the threshold N. Algorithm 1 also demonstrates one striking characteristic of our method: A KNN classifier is applied to determine the label of the current word before the CRF model. The labels of the words that confidently assigned by the KNN classifier are treated as visible variables for the CRF model.

4.2 Model

Our model is hybrid in the sense that a KNN classifier and a CRF model are sequentially applied to the target tweet, with the goal that the KNN classifier captures global coarse evidence while the CRF model fine-grained information encoded in a single tweet and in the gazetteers. Algorithm 2 outlines the training process of KNN, which records the labeled word vector for every type of label.

Algorithm 2 KNN Training.

Require: Training tweets ts.

1: Initialize the classifier l_k:$l_k = \varnothing$.
2: **for** Each tweet $t \in ts$ **do**
3: **for** Each word,label pair $(w, c) \in t$ **do**
4: Get the feature vector $\vec{w}$: $\vec{w} = repr_w(w, t)$.
5: Add the $\vec{w}$ and c pair to the classifier: $l_k = l_k \cup \{(\vec{w}, c)\}$.
6: **end for**
7: **end for**
8: **return** KNN classifier l_k.

Algorithm 3 describes how the KNN classifier predicts the label of the word. In our work, K is experimentally set to 20, which yields the best performance.

Two desirable properties of KNN make it stand out from its alternatives: 1) It can straightforwardly incorporate evidence from new labeled tweets and retraining is fast; and 2) combining with a CRF model, which is good at encoding the subtle interactions between words and their labels, compensates for KNN's incapability to capture fine-grained evidence involving multiple decision points.

[4]The training set ts has a maximum allowable number of items, which is 10,000 in our work. Adding an item into it will cause the oldest one being removed if it is full.

Algorithm 3 KNN predication.

Require: KNN classifier l_k ;word vector $\vec{w}$.

1: Initialize nb, the neighbors of $\vec{w}$: $nb = neigbors(l_k, \vec{w})$.
2: Calculate the predicted class c^*: $c^* = argmax_c \sum_{(\vec{w}',c') \in nb} \delta(c, c') \cdot cos(\vec{w}, \vec{w}')$.
3: Calculate the labeling confidence cf: $cf = \frac{\sum_{(\vec{w}',c') \in nb} \delta(c,c') \cdot cos(\vec{w},\vec{w}')}{\sum_{(\vec{w}',c') \in nb} cos(\vec{w},\vec{w}')}$.
4: **return** The predicted label c^* and its confidence cf.

The Linear CRF model is used as the fine model, with the following considerations: 1) It is well-studied and has been successfully used in state-of-the-art NER systems (Finkel et al., 2005; Wang, 2009); 2) it can output the probability of a label sequence, which can be used as the labeling confidence that is necessary for the semi-supervised learning framework.

In our experiments, the CRF++ [5] toolkit is used to train a linear CRF model. We have written a Viterbi decoder that can incorporate partially observed labels to implement the crf function in Algorithm 1.

4.3 Features

Given a word in a tweet, the KNN classifier considers a text window of size 5 with the word in the middle (Zhang and Johnson, 2003), and extracts bag-of-word features from the window as features. For each word, our CRF model extracts similar features as Wang (2009) and Ratinov and Roth (2009), namely, orthographic features, lexical features and gazetteers related features. In our work, we use the gazetteers provided by Ratinov and Roth (2009).

Two points are worth noting here. One is that before feature extraction for either the KNN or the CRF, stop words are removed. The stop words used here are mainly from a set of frequently-used words [6]. The other is that tweet meta data is normalized, that is, every link becomes *LINK* and every account name becomes *ACCOUNT*. Hash tags are treated as common words.

[5]http://crfpp.sourceforge.net/

[6]http://www.textfixer.com/resources/common-english-words.txt

4.4 Discussion

We now discuss several design considerations related to the performance of our method, i.e., additional features, gazetteers and alternative models.

Additional Features. Features related to chunking and parsing are not adopted in our final system, because they give only a slight performance improvement while a lot of computing resources are required to extract such features. The ineffectiveness of these features is linked to the noisy and informal nature of tweets. Word class (Brown et al., 1992) features are not used either, which prove to be unhelpful for our system. We are interested in exploring other tweet representations, which may fit our NER task, for example the LSA models (Guo et al., 2009).

Gazetteers. In our work, gazetteers prove to be substantially useful, which is consistent with the observation of Ratinov and Roth (2009). However, the gazetteers used in our work contain noise, which hurts the performance. Moreover, they are static, directly from Ratinov and Roth (2009), thus with a relatively lower coverage, especially for person names and product names in tweets. We are developing tools to clean the gazetteers. In future, we plan to feed the fresh entities correctly identified from tweets back into the gazetteers. The correctness of an entity can rely on its frequency or other evidence.

Alternative Models. We have replaced KNN by other classifiers, such as those based on Maximum Entropy and Support Vector Machines, respectively. KNN consistently yields comparable performance, while enjoying a faster retraining speed. Similarly, to study the effectiveness of the CRF model, it is replaced by its alternations, such as the HMM labeler and a beam search plus a maximum entropy based classifier. In contrast to what is reported by Ratinov and Roth (2009), it turns out that the CRF model gives remarkably better results than its competitors. Note that all these evaluations are on the same training and testing data sets as described in Section 5.1.

5 Experiments

In this section, we evaluate our method on a manually annotated data set and show that our system outperforms the baselines. The contributions of the combination of KNN and CRF as well as the semi-supervised learning are studied, respectively.

5.1 Data Preparation

We use the Twigg SDK [7] to crawl all tweets from April 20^{th} 2010 to April 25^{th} 2010, then drop non-English tweets and get about 11,371,389, from which 15,800 tweets are randomly sampled, and are then labeled by two independent annotators, so that the beginning and the end of each named entity are marked with <TYPE> and </TYPE>, respectively. Here TYPE is PERSON, PRODUCT, ORGANIZATION or LOCATION. 3555 tweets are dropped because of inconsistent annotation. Finally we get 12,245 tweets, forming the gold-standard data set. Figure 1 shows the portion of named entities of different types. On average, a named entity has 1.2 words. The gold-standard data set is evenly split into two parts: One for training and the other for testing.

5.2 Evaluation Metrics

For every type of named entity, Precision (Pre.), recall (Rec.) and F1 are used as the evaluation metrics. Precision is a measure of what percentage the output labels are correct, and recall tells us to what percentage the labels in the gold-standard data set are correctly labeled, while F1 is the harmonic mean of precision and recall. For the overall performance, we use the average Precision, Recall and F1, where the weight of each name entity type is proportional to the number of entities of that type. These metrics are widely used by existing NER systems to evaluate their performance.

5.3 Baselines

Two systems are used as baselines: One is the dictionary look-up system based on the gazetteers; the other is the modified version of our system without KNN and semi-supervised learning. Hereafter these two baselines are called NER_{DIC} and NER_{BA}, respectively. The OpenNLP and the Stanford parser (Klein and Manning, 2003) are used to extract linguistic features for the baselines and our method.

5.4 Basic Results

Table 1 shows the overall results for the baselines and ours with the name NER_{CB}. Here our system is trained as described in Algorithm 1, combin-

[7] It is developed by the Bing social search team, and currently is only internally available.

Table 1: Overall experimental results.

System	Pre.(%)	Rec.(%)	F1(%)
NER_{CB}	81.6	78.8	80.2
NER_{BA}	83.6	68.6	75.4
NER_{DIC}	32.6	25.4	28.6

Table 2: Experimental results on PERSON.

System	Pre.(%)	Rec.(%)	F1(%)
NER_{CB}	78.4	74.5	76.4
NER_{BA}	83.6	68.4	75.2
NER_{DIC}	37.1	29.7	33.0

Table 3: Experimental results on PRODUCT.

System	Pre.(%)	Rec.(%)	F1(%)
NER_{CB}	81.3	65.4	72.5
NER_{BA}	82.5	58.4	68.4
NER_{DIC}	8.2	6.1	7.0

Table 4: Experimental results on LOCATION.

System	Pre.(%)	Rec.(%)	F1(%)
NER_{CB}	80.3	77.5	78.9
NER_{BA}	81.6	69.7	75.2
NER_{DIC}	30.2	30.0	30.1

Table 5: Experimental results on ORGANIZATION.

System	Pre.(%)	Rec.(%)	F1(%)
NER_{CB}	83.2	60.4	70.0
NER_{BA}	87.6	52.5	65.7
NER_{DIC}	54.5	11.8	19.4

ing a KNN classifier and a CRF labeler, with semi-supervised learning enabled. As can be seen from Table 1, on the whole, our method significantly outperforms (with $p < 0.001$) the baselines. Tables 2-5 report the results on each entity type, indicating that our method consistently yields better results on all entity types.

5.5 Effects of KNN Classifier

Table 6 shows the performance of our method without combining the KNN classifier, denoted by NER_{CB-KNN}. A drop in performance is observed then. We further check the confidently predicted labels of the KNN classifier, which account for about 22.2% of all predications, and find that its F1 is as high as 80.2% while the baseline system based on the CRF model achieves only an F1 of 75.4%. This largely explains why the KNN classifier helps the CRF labeler. The KNN classifier is replaced with its competitors, and only a slight difference in performance is observed. We do observe that retraining KNN is obviously faster.

5.6 Effects of the CRF Labeler

Similarly, the CRF model is replaced by its alternatives. As is opposite to the finding of Ratinov and Roth (2009), the CRF model gives remarkably better results, i.e., 2.1% higher in F1 than its best followers (with $p < 0.001$). Table 7 shows the overall performance of the CRF labeler with various feature set combinations, where F_o, F_l and F_g denote the orthographic features, the lexical features and the gazetteers related features, respectively. It can be seen from Table 7 that the lexical and gazetteer related features are helpful. Other advanced features such as chunking are also explored but with no significant improvement.

5.7 Effects of Semi-supervised Learning

Table 8 compares our method with its modified version without semi-supervised learning, suggesting that semi-supervised learning considerably boosts the performance. To get more details about self-training, we evenly divide the test data into 10 parts and feed them into our method sequentially; we record the average F1 score on each part, as shown in Figure 2.

5.8 Error Analysis

Errors made by our system on the test set fall into three categories. The first kind of error, accounting for 35.5% of all errors, is largely related to slang expressions and informal abbreviations. For example, our method identifies "Cali", which actually means "California", as a PERSON in the tweet "i love Cali so much". In future, we can design a normalization component to handle such slang expressions and informal abbreviations.

The second kind of error, accounting for 37.2% of all errors, is mainly attributed to the data sparseness. For example, for this tweet "come to see jax-

Table 6: Overall performance of our system with and without the KNN classifier, respectively.

System	Pre.(%)	Rec.(%)	F1(%)
NER_{CB}	81.6	78.8	80.2
NER_{CB-KNN}	82.6	74.8	78.5

Table 7: Overview performance of the CRF labeler (combined with KNN) with different feature sets.

Features	Pre.(%)	Rec.(%)	F1(%)
F_o	71.3	42.8	53.5
$F_o + F_l$	76.2	44.2	55.9
$F_o + F_g$	80.5	66.2	72.7
$F_o + F_l + F_g$	82.6	74.8	78.5

on someday", our method mistakenly labels "jaxon" as a LOCATION, which actually denotes a PERSON. This error is understandable somehow, since this tweet is one of the earliest tweets that mention "jaxon", and at that time there was no strong evidence supporting that it represents a person. Possible solutions to these errors include continually enriching the gazetteers and aggregating additional external knowledge from other channels such as traditional news.

The last kind of error, which represents 27.3% of all errors, somehow links to the noise prone nature of tweets. Consider this tweet "wesley snipes ws cought 4 nt payin tax coz ths celebz dnt take it cirus.", in which "wesley snipes" is not identified as a PERSON but simply ignored by our method, because this tweet is too noisy to provide effective features. Tweet normalization technology seems a possible solution to alleviate this kind of error.

6 Conclusions and Future work

We propose a novel NER system for tweets, which combines a KNN classifier with a CRF labeler under a semi-supervised learning framework. The KNN classifier collects global information across recently labeled tweets while the CRF labeler exploits information from a single tweet and from the gazetteers. A serials of experiments show the effectiveness of our method, and particularly, show the positive effects of KNN and semi-supervised learning.

In future, we plan to further improve the performance of our method through two directions. Firstly, we hope to develop tweet normalization technology to make tweets friendlier to the NER task. Secondly, we are interested in integrating new entities from tweets or other channels into the gazetteers.

Table 8: Performance of our system with and without semi-supervised learning, respectively.

Features	Pre.(%)	Rec.(%)	F1(%)
NER_{CB}	81.6	78.8	80.2
NER'_{CB}	82.1	71.9	76.7

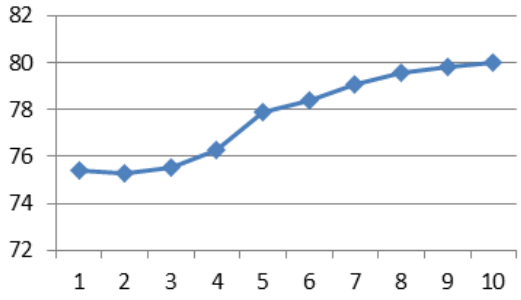

Figure 2: F1 score on 10 test data sets sequentially fed into the system, each with 600 instances. Horizontal and vertical axes represent the sequential number of the test data set and the averaged F1 score (%), respectively.

Acknowledgments

We thank Long Jiang, Changning Huang, Yunbo Cao, Dongdong Zhang for helpful discussions, and the anonymous reviewers for their valuable comments. We also thank Matt Callcut for his careful proofreading of an early draft of this paper.

References

Peter F. Brown, Peter V. deSouza, Robert L. Mercer, Vincent J. Della Pietra, and Jenifer C. Lai. 1992. Class-based n-gram models of natural language. *Comput. Linguist.*, 18:467–479.

Laura Chiticariu, Rajasekar Krishnamurthy, Yunyao Li, Frederick Reiss, and Shivakumar Vaithyanathan. 2010. Domain adaptation of rule-based annotators for named-entity recognition tasks. In *EMNLP*, pages 1002–1012.

Michael Collins. 2002. Discriminative training methods for hidden markov models: theory and experiments with perceptron algorithms. In *EMNLP*, pages 1–8.

Doug Downey, Matthew Broadhead, and Oren Etzioni.

2007. Locating Complex Named Entities in Web Text. In *IJCAI*.

Oren Etzioni, Michael Cafarella, Doug Downey, Ana-Maria Popescu, Tal Shaked, Stephen Soderland, Daniel S. Weld, and Alexander Yates. 2005. Unsupervised named-entity extraction from the web: an experimental study. *Artif. Intell.*, 165(1):91–134.

Tim Finin, Will Murnane, Anand Karandikar, Nicholas Keller, Justin Martineau, and Mark Dredze. 2010. Annotating named entities in twitter data with crowdsourcing. In *CSLDAMT*, pages 80–88.

Jenny Rose Finkel and Christopher D. Manning. 2009. Nested named entity recognition. In *EMNLP*, pages 141–150.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL*, pages 363–370.

Honglei Guo, Huijia Zhu, Zhili Guo, Xiaoxun Zhang, Xian Wu, and Zhong Su. 2009. Domain adaptation with latent semantic association for named entity recognition. In *NAACL*, pages 281–289.

Martin Jansche and Steven P. Abney. 2002. Information extraction from voicemail transcripts. In *EMNLP*, pages 320–327.

Jing Jiang and ChengXiang Zhai. 2007. Instance weighting for domain adaptation in nlp. In *ACL*, pages 264–271.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *ACL*, pages 423–430.

Vijay Krishnan and Christopher D. Manning. 2006. An effective two-stage model for exploiting non-local dependencies in named entity recognition. In *ACL*, pages 1121–1128.

George R. Krupka and Kevin Hausman. 1998. Isoquest: Description of the netowl^{TM} extractor system as used in muc-7. In *MUC-7*.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289.

Andrew Mccallum and Wei Li. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *HLT-NAACL*, pages 188–191.

Scott Miller, Jethran Guinness, and Alex Zamanian. 2004. Name tagging with word clusters and discriminative training. In *HLT-NAACL*, pages 337–342.

Einat Minkov, Richard C. Wang, and William W. Cohen. 2005. Extracting personal names from email: applying named entity recognition to informal text. In *HLT*, pages 443–450.

David Nadeau and Satoshi Sekine. 2007. A survey of named entity recognition and classification. *Linguisticae Investigationes*, 30:3–26.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *CoNLL*, pages 147–155.

Sameer Singh, Dustin Hillard, and Chris Leggetter. 2010. Minimally-supervised extraction of entities from text advertisements. In *HLT-NAACL*, pages 73–81.

Jun Suzuki and Hideki Isozaki. 2008. Semi-supervised sequential labeling and segmentation using giga-word scale unlabeled data. In *ACL*, pages 665–673.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: language-independent named entity recognition. In *HLT-NAACL*, pages 142–147.

Yefeng Wang. 2009. Annotating and recognising named entities in clinical notes. In *ACL-IJCNLP*, pages 18–26.

Dan Wu, Wee Sun Lee, Nan Ye, and Hai Leong Chieu. 2009. Domain adaptive bootstrapping for named entity recognition. In *EMNLP*, pages 1523–1532.

Kazuhiro Yoshida and Jun'ichi Tsujii. 2007. Reranking for biomedical named-entity recognition. In *BioNLP*, pages 209–216.

Tong Zhang and David Johnson. 2003. A robust risk minimization based named entity recognition system. In *HLT-NAACL*, pages 204–207.

GuoDong Zhou and Jian Su. 2002. Named entity recognition using an hmm-based chunk tagger. In *ACL*, pages 473–480.