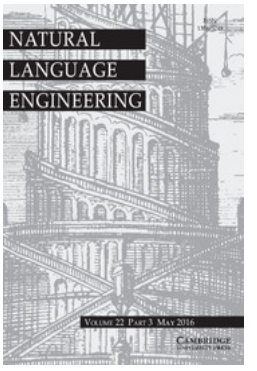

TwitterNEED: A hybrid approach for named entity extraction and disambiguation for tweet

MENA B. HABIB and MAURICE VAN KEULEN

Natural Language Engineering / Volume 22 / Issue 03 / May 2016, pp 423 - 456
DOI: 10.1017/S1351324915000194, Published online: 10 July 2015

Natural Language Engineering **22** (3): 423–456. © Cambridge University Press 2015
doi:10.1017/S1351324915000194

*TwitterNEED: A hybrid approach for named entity extraction and disambiguation for tweet**

MENA B. HABIB and MAURICE VAN KEULEN
Database Chair, University of Twente, Enschede, the Netherlands
e-mail: m.b.habib@ewi.utwente.nl, m.vankeulen@ewi.utwente.nl

(*Received 19 March 2014; revised 22 May 2015; accepted 5 June 2015; first published online 10 July 2015*)

Abstract

Twitter is a rich source of continuously and instantly updated information. Shortness and informality of tweets are challenges for Natural Language Processing tasks. In this paper, we present TwitterNEED, a hybrid approach for Named Entity Extraction and Named Entity Disambiguation for tweets. We believe that disambiguation can help to improve the extraction process. This mimics the way humans understand language and reduces error propagation in the whole system. Our extraction approach aims for high extraction recall first, after which a Support Vector Machine attempts to filter out false positives among the extracted candidates using features derived from the disambiguation phase in addition to other word shape and Knowledge Base features. For Named Entity Disambiguation, we obtain a list of entity candidates from the YAGO Knowledge Base in addition to top-ranked pages from the Google search engine for each extracted mention. We use a Support Vector Machine to rank the candidate pages according to a set of URL and context similarity features. For evaluation, five data sets are used to evaluate the extraction approach, and three of them to evaluate both the disambiguation approach and the combined extraction and disambiguation approach. Experiments show better results compared to our competitors DBpedia Spotlight, Stanford Named Entity Recognition, and the AIDA disambiguation system.

1 Introduction

1.1 Overview

The rapid growth of information technology in the last two decades has led to a growth in the Internet accessibility and usage. The increasing use of the Internet is a contributing factor to the popularity growth of social networks. Social media content represents a big part of all textual content appearing on the Internet. According to an eMarketer report (Winkels 2013), nearly one in four people worldwide used social networks in 2013. By 2017, the global social network audience is estimated to total 2.55 billion. Twitter as an example of a highly active social media network,

* The authors would like to thank Zhemin Zhu for sharing his CRF model (Zhu *et al.* 2013) and assisting us in applying it. This work is supported by the Dutch national research program COMMIT.

has 284 million monthly active users publishing over 500 million tweets every day[1]. These numbers are growing exponentially, and the streams of user-generated content provide an opportunity and challenge for data analysts.

Making use of social media content requires measuring, analyzing, and interpreting interactions and associations between people, topics, and ideas. An example of a main sector for social media analysis is the area of customer feedback through social media. With so many feedback channels, organizations need to mix and match them to best suit corporate needs and customer preferences. Another beneficial sector is safety and security. Communications over social networks have helped put entire nations to action. Social media played a key role in The Arab Spring that started in 2010 in Tunisia (Howard and Hussain 2013). The riots that broke out across England during the summer of 2011 also showed the power of social media. The growing threats associated with social media have been an alarm to government security agencies. There is a growing demand to automatically monitor discussions on social media as a source of intelligence. Increasing resources within investigative agencies are being spent on monitoring social media.

As a concrete example, the research of this paper has been applied in the TEC4SE project[2]. The aim of the project is to improve operational decision making by police and emergency services at massive events by gathering and combining information from different sources such as surveillance cameras, police officers in the field, and — to which we contributed — social media. As a result, signals about relevant events are picked up earlier and information about the event becomes available sooner. Existing tools and technologies for social media monitoring are typically based on keywords only which lead to many false signals reducing the source's effectiveness. They are often also ill equipped for dealing with the informal language used in social media.

Information extraction (IE) is the research field that is concerned with obtaining structured information from unstructured text. IE systems attempt to interpret human language text in order to extract information about different types of events, entities, or relationships. Named Entity Extraction (NEE) is a subtask of IE that aims to locate phrases (mentions) in the text that represent names of persons, organizations, or locations regardless of their type. It differs from the task Named Entity Recognition (NER) (also known as Named Entity Recognition and Classification) which involves both extraction and classification into a set of predefined classes (e.g., person, organization, and location). NED is the task of determining which concrete person, place, event, etc. is referred to by a mention. Wikipedia articles are widely used as an entity's reference. For example, the mention '*Victoria*' may refer to one of many entities such as '`http://en.wikipedia.org/wiki/Victoria_(Australia)`' or '`http://en.wikipedia.org/wiki/Queen_Victoria`'. According to the YAGO Knowledge Base (KB) (Suchanek, Kasneci and Weikum 2007), the mention '*Victoria*' may refer to one of 188 entities in Wikipedia.

[1] `https://about.twitter.com/company`
[2] `http://www.tec4se.nl/`

Twitter messages are noisy and contain a lot of spam and useless contents. According to a deep study made by Pear Analytics[3], 40.55 per cent of Twitter posts are classified as *Pointless Babble*; 37.55 per cent as *Conversational*; 8.7 per cent as *Pass-Along Value*; and 3.75 per cent as *Spam*[4]. Dann (2010) and Sullivan (2012) presented similar analysis on the contents of Twitter messages. Numerous approaches were introduced to measure the information credibility on Twitter (Castillo, Mendoza and Poblete 2011) and to filter spam (Verma, Divya and Sofat 2014). Our focus in this paper is only on the processes of NEE and NED. We assume the existence of other components for message prefiltering and classification prior to the NEE and NED processes.

1.2 Challenges

NEE and NED in informal text are challenging. Here, we summarize the challenges of NEE and NED for tweets as an example of informal text:

- The informal nature of tweets makes the extraction process more difficult. For example, in Table 1 Case 1, it is hard to extract the mentions (i.e., phrases that represent Named entities, NEs) using traditional NEE methods because of the ill-formed sentence structure. Traditional NEE methods might extract '*Grampa*' as a mention because of its capitalization. Furthermore, it is hard to extract the mention '*Speechless*', which is a common English word, but should here be interpreted as the name of a song; it requires further knowledge about '*Lady Gaga*' songs.
- The limited length (140 characters) of tweets forces the senders to provide dense information. Users resort to acronyms to preserve space. Informal language is another way to express more information in less space. All of these problems make both the extraction and the disambiguation processes more complex. For example, in Table 1, Case 2 shows two abbreviations ('*Qld*' and '*Vic*'). It is hard to infer which entities are meant without extra information.
- The limited coverage of a KB is another challenge facing NED for tweets. According to Lin, Mausam and Etzioni (2012), 5 million out of 15 million mentions on the Web cannot be linked to Wikipedia. This means that relying only on a KB for NED leads to around 33 per cent loss in the disambiguated entities. This percentage is higher on Twitter because of its social nature where users also discuss information about non-famous entities. For example, Table 1 Case 3 contains two mentions for two users on the '*My Second Life*' social network. It is unlikely that one can find their entities in a KB. However, their profile pages ('`https://my.secondlife.com/laelith.demonia`' and '`https://my.secondlife.com/liwanu.hird`') can be found by a search engine. Moreover, some NEs do not have home or profile pages at all. Extracting such information could be useful to know the interests of the Twitter

[3] `https://www.pearanalytics.com`

[4] `https://www.pearanalytics.com/wp-content/uploads/2012/12/Twitter-Study-August-2009.pdf`

Table 1. *Examples of challenging cases for NEE and NED in tweets (NE mentions are written in boldface)*

Case #	Tweet content
1	– **Lady Gaga** – **Speechless** live @ **Helsinki** 10/13/2010 http://www.youtube.com/watch?v=yREociHyijk ladygaga also talks about her Grampa who died recently
2	**Qld** flood victims donate to **Vic** bushfire appeal – http://is.gd/iZ4x
3	**Laelith Demonia** has just defeated **liwanu Hird**. Career wins is 575, career losses is 966
4	Adding **Win7Beta**, **Win2008**, and **Vista** x64 and x86 images to munin. #wds
5	history should show that **bush jr** should be in jail or at least never should have been president
6	RT @BBCClick: Joy! **MS Office** now syncs with **Google Docs** (well, in beta anyway). We are soon to be one big happy (cont) http://tl.gd/73t94u
7	'Even Writers Can Help..An Appeal For **Australian** Bushfire Victims' http://cli.gs/Zs8zL2

user and hence providing her/him with the appropriate follow suggestions. The follow suggestions provided by Twitter are mainly based on the follower graph and the interaction graphs (e.g., graphs defined in terms of retweets, favorites, replies, and other user interactions) (Gupta *et al.* 2013). Adding semantics and topic detection can be a good improvement to the Twitter's recommendation algorithm.

- NE representation in KBs poses another NED challenge. The YAGO KB uses the Wikipedia anchor text as a possible mention representation for NEs. However, there may be more representations that do not appear in the Wikipedia anchor text, but are meant to refer to the entity because of a spelling mistake or because of a new abbreviation for the entity. For example, in Table 1 Case 4, the mentions '*Win7Beta*' and '*Win2008*' do not appear in the YAGO mention-entity lookup table, although they refer to the covered entities '`http://en.wikipedia.org/wiki/Windows_7`' and '`http://en.wikipedia.org/wiki/Windows_Server_2008`' respectively.
- The processes of NEE and NED involve degrees of uncertainty. For example, in Table 1 Case 5, for some NER systems, it may be uncertain whether the word '*jr*' should be part of the mention *bush* or not. The same for '*Office*' and '*Docs*' in Case 6, which some extractors may miss. Another example is Case 7 in which it is hard to assess whether '*Australian*' should refer to '`http://en.wikipedia.org/wiki/Australia`' or '`http://en.wikipedia.org/wiki/Australian_people`'[5]. Both might be correct. This is the reason why we believe that it is better to fundamentally consider sets of possible alternatives in the processes of NEE and NED.
- The last challenge we would like to draw attention to, is the recency of the KBs. While the state-of-the-art NED approaches rely mainly on KBs for the

[5] Some NER data sets consider nationalities as NEs (Basave *et al.* 2013).

disambiguation process, the rate of updating the KBs is not as quick as the rising of new famous entities. For example, the page of '*Barack Obama*' on Wikipedia was created on 18 March 2004. Before that date '*Barack Obama*' was a member of the Illinois Senate and you could find his profile page on '`http://www.ilga.gov/senate/Senator.asp?MemberID=747`'. It is common on social networks that users talk about a nonfamous entity who becomes a public figure later.

1.3 Applications

NEE and NED for social media have applications in a wide range of domains. Here, we give some examples for possible applications:

- The business experts always look for the customers' feedback to help their decision making. Social media is an important source of information about customers' attitude and opinion. For example, if the 'Hi' telecommunication company in the Netherlands wants to gather information about its user satisfaction, it is required to find the Tweets which contain the term 'Hi' that refers to the company name but not to the greeting word. Here, the NED module plays the key role.
- Security agencies normally analyze large amounts of text manually to search for information about people involved in criminal or terrorist activities. Providing ways of automatic IE should significantly enhance monitoring and tracking of illegal activities.

1.4 Contributions

The paper makes the following contributions:

- We introduce a combined system for NEE and NED in tweets that uses their interdependency and mimics how humans exploit it in language understanding.
- We propose a generic open world approach for NED in tweets for any NE. Mentions are disambiguated by assigning them to either a Wikipedia article or a home page.
- We handle the uncertainty involved in the disambiguation process by providing a ranked list of possible entity home pages instead of a hard assignment for a mention to one entity page.
- The proposed extraction approach is shown to be robust against the coverage of KBs and the informality of the used language.

1.5 Paper structure

The rest of the paper is organized as follows. Section 2 describes briefly the key aspects of our approach. Section 3 presents related work on NEE and NED in tweets. Sections 4 and 5 present our approach for NED and NEE in tweets respectively along with the experimental results. Finally, conclusions and future work are presented in Section 6.

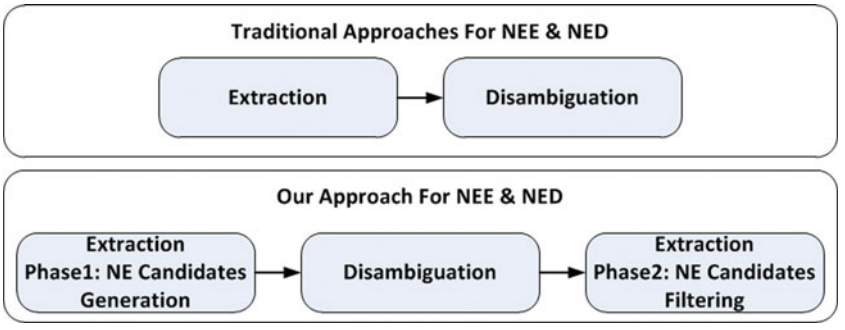

Fig. 1. (Colour online) Traditional approaches versus our approach for NEE and NED.

2 TwitterNEED general approach

Natural language processing (NLP) tasks are commonly composed of a set of chained sub tasks that form the processing pipeline. The residual error produced in these sub tasks propagates, affecting the final process results. In this paper, we focus on NEE and NED which are two common processes in many NLP applications. We claim that feedback derived from disambiguation can improve the extraction, which in turn can improve the disambiguation again. This is the same way, we believe, we as humans understand text. The capability to successfully understand language requires one to acquire a range of skills including syntactic and semantic, and to possess an extensive vocabulary. We try to mimic a human's way of reasoning to solve the NEE and NED problems. Consider the tweet in Table 1 Case 1, one would use regular expressions to recognize '10/13/2010' as a date. Furthermore, prior knowledge enables one to recognize '*Lady Gaga*' and '*Helsinki*' as a singer name and location name respectively or at least as names if one does not know exactly what they refer to. However, the term '*Speechless*' involves some ambiguity as it can be an adjective as well as a name for something. A feedback clue from '*Lady Gaga*' could increase one's confidence that it refers to a song. Even without knowing that '*Lady Gaga*' sings a song called '*Speechless*', there are sufficient clues to guess with quite high probability that it is a song. The pattern 'live @' in association with disambiguating '*Lady Gaga*' as a singer name and '*Helsinki*' as a location name, provides a strong suspicion that '*Speechless*' is a song. We strive for mimicking such inferences in properly interpreting the NEs in a tweet.

Although the logical order for a traditional IE system is to complete the extraction process before commencing with the disambiguation process, we start with an initial extraction-like phase aiming for high recall (i.e., aiming to find as many reasonable mention candidates as possible). We then attempt disambiguation for all the extracted mentions. Finally, we classify extracted mention candidates into true and false NE using features (clues) derived from the results of the disambiguation phase such as KB information and entity coherency. Figure 1 illustrates our general approach contrasted with the traditional process.

Unlike NER systems which extract mentions of entities and assign them to one predefined category such as location, person, or organization, we focus first on

extracting mentions regardless of their categories. We leave this classification to the disambiguation step which links the mentions to their real entities.

The potential of this order is that the disambiguation step gives extra clues (such as entity-tweet context similarity) about each NE candidate. This information can help in the decision whether the candidate is a true NE or not. For example, consider the tweet in Case 1 in Table 1. It is uncertain, even to a human, to recognize '*Speechless*' as a song name without prior information about the songs of '*Lady Gaga*'. Our approach is able to capture such problematic cases of NEs.

2.1 Named entity disambiguation approach

For NED, almost all researchers use entities in KBs as references for the extracted mentions (see Section 3). Furthermore, researchers who studied NED in tweets are mostly entity oriented, i.e., given entities such as '*Apple Inc*' or '*Microsoft*', they strive to determine whether mentions like '*Apple*' or '*MS*' are representatives for those entities or not.

In our opinion, for the NED task in tweets, it is necessary to have a generic system that does not rely on the closed world of KBs only for disambiguation. In this paper, we propose a generic open world NED approach where mentions are disambiguated by assigning them to either a Wikipedia article or a home page.

Given an extracted mention, we query Google to obtain a set of possible entity candidates' home pages. We then enrich the candidate list with Wikipedia articles obtained by querying the YAGO KB.

For each entity candidate, we extract a set of context and URL features. Context features (such as language model and overlapping terms between tweet and document) measure the context similarity between mention context (the tweet text) and entity candidates' home pages. URL features (such as path length and mention-URL string similarity) measure the likelihood of the candidate URL being a representative of the entity home page. In addition, we use the prior probability from the YAGO KB for the entity being referred to by a certain mention. A Support Vector Machine (SVM) with Radial Basis Functions kernel is trained on the aforementioned features and used to rank the candidate pages.

Although coherence features (Entity–Entity Similarity) could be helpful for disambiguation (Yosef *et al.* 2011) and also for extraction (Habib and van Keulen 2012a; Habib and van Keulen 2013), we avoid using such features. The reliability of those features would be harmed by the high number of false positives among the extracted NE candidates. In order to use coherence features in our combined system, it is required to repeat the extraction and disambiguation in loops as described in (Habib and van Keulen 2012a), but this is beyond the scope of this paper.

We conduct experiments on three different data sets of tweets with different characteristics. Our approach achieves better disambiguation results on both sets compared with the baselines and a competitor.

2.2 Named entity extraction approach

For NEE, we believe uncertainty is inherent to this process. Our approach is based on finding as many NE candidates (mentions) as possible (achieving high recall) and then filtering those candidates. To achieve this high recall, we use a tweet segmentation method to segment the tweet into a sequence of consecutive phrases (Li *et al.* 2012), in addition to phrases that match KB entries using a lookup strategy. Furthermore, we use a Conditional Random Fields (CRF) model to generate a top-k of possible NE annotations for each tweet. We use all those annotations as candidates for NEs. To improve the precision, we apply an SVM model to predict if the candidate is an NE or not according to a set of features. We use word shape features (such as capitalization), Part Of Speech (POS) tags, KB features (such as number of possible entities for the given extracted mention), and features derived from disambiguation process (such as similarity between the mention context and the disambiguated entity page).

We also consider the best annotation set for the tweet given by the CRF model as true positives. Extraction results obtained from both SVM and CRF are combined. The idea behind this combination is that SVM and CRF work in a different way. The former is a distance-based classifier that uses numeric features for classification which CRF cannot handle, while the latter is a probabilistic model that can naturally consider state-to-state dependencies and feature-to-state dependencies. On the other hand, SVM does not consider such dependencies. The hybrid approach makes use of the strength of each.

3 Related work

3.1 Named entity disambiguation

3.1.1 For formal text

NED in Web documents is a topic that is well covered in literature. Several approaches use Wikipedia or a KB derived from Wikipedia (such as DBpedia and YAGO) as entity store to lookup the suitable entity for a mention.

One of the earliest approaches was proposed by Bunescu and Pasca (2006). The authors developed an NE disambiguation system that does disambiguation in two steps. First, it detects whether a proper name refers to a NE in the dictionary (detection). Second, it disambiguates between multiple NEs denoted by the same proper name (disambiguation). Furthermore, the authors defined a similarity measure that compares the context of a mention with the Wikipedia categories of an entity candidate.

Cucerzan (2007) proposes a large-scale system for disambiguating NEs based on information extracted from Wikipedia and Web search results. The system uses the data associated with the known surface forms identified in a document and all their possible entity disambiguations to maximize the agreement between the context data stored for the candidate entities and the contextual information in the document, and also, the agreement among the category tags of the candidate entities.

The importance of an entity–entity coherence measure in disambiguation is emphasized by Kulkarni *et al.* (2009). Similarly, Hoffart *et al.* (2011) combine three measures: the prior probability of an entity being mentioned, the similarity between the context of a mention and of a candidate entity, as well as the coherence among candidate entities for all mentions. AIDA[6] (Yosef *et al.* 2011) is a system that is built on Hoffart *et al.* (2011) approach. We used AIDA as a competitor in our paper.

Ad hoc (targeted-entity) NED represents another direction in NED research. Ad-hoc entities do not exist in a KB such as DBpedia, Freebase, or YAGO. Instead of using a KB giving the candidate mentions of all the target entities, targeted-entity disambiguation approaches determine which ones are true mentions of a target entity. Examples for such an approach are presented in Srinivasan, Chen and Srihari, (2009) and Wang *et al.* (2012). In Srinivasan *et al.* (2009), the authors proposed a cross-document person name disambiguation system that clusters documents so that each cluster contains all and only those documents referring to the same person. They introduced features based on topic models and also document-level entity profiles, sets of information that are collected for each ambiguous person in the entire document. In Wang *et al.* (2012), the authors introduced disambiguation techniques that require no knowledge about the targeted entities except their names. They proposed a graph-based model called MentionRank to leverage the homogeneity constraint and disambiguate the candidate mentions collectively across the document. Leveraging the homogeneity constraint of the entities is done in three ways: context similarity, comentioned entities, and cross-document, cross-entity interdependence.

3.1.2 For informal short text

Researchers have attempted NED for informal short text such as tweets. Most of this research investigates the problem of targeted-entity disambiguation. Within this theme, Spina *et al.* (2011), Delgado *et al.* (2012) and Yerva *et al.* (2012) focus on the task of filtering tweets containing a given company name, depending whether the tweet is actually related to the company or not. They develop a set of features (co-occurrence, Web-based features, collection-based features) to find keywords for positive and negative cases. Similarly, Christoforaki, Erunse and Yu (2011) propose a topic-centric entity extraction system where interesting entities pertaining to a topic are mined and extracted from short messages and returned as search results on the topic.

A supervised approach for real-time NED in tweets is proposed by Davis *et al.* (2012). They focused on the problem of continuously monitoring the Twitter stream and predicting whether an incoming message containing mentions indeed refers to a predefined entity or not. The authors propose a three-stage pipeline technique. In the first stage, filtering rules (collocations, users, hash tags) are used to identify clearly positive examples of messages truly mentioning the real-world entities. These messages are given as input to an Expectation–Maximization method (the second

[6] `https://d5gate.ag5.mpi-sb.mpg.de/webaida/`

stage), which produces training information to be used during the last stage. Finally, in the last stage they use the training set produced by the previous stage to classify unlabeled messages in real time. Another real-time analysis tool is proposed by Steiner *et al.* (2013). The authors provide a browser extension which is based on a combination of several third party NLP APIs in order to add more semantics and annotations to Twitter and Facebook microposts.

Similar to our problem discussed in Section 4, is the problem of entity home page finding, which was part of the TREC Web and entity tracks. The task is to extract a target entity and find its home page given an input entity, the type of the target entity, and the relationship between the input and the target entity. One of the proposed approaches for this task was Westerveld, Kraaij and Hiemstra (2002). The authors combine content information with other sources as diverse as inlinks, URLs, and anchors to find an entry page. Another approach for entity home page recognition was introduced by Li *et al.* (2009). It selects the features of a link or Web page content, and constructs entity home page classifiers by using three kinds of machine learning algorithms, logistic, SVM, and AdaBoost, to discover the optimal entity home page.

Although the TREC problem looks similar to ours, the tweets' short informal nature makes it more tricky to find an entity reference page. Moreover, distinguishing entities that could be linked to Wikipedia pages (Wiki entities) from entities that only have a normal home page or profile page (Non-Wiki entities), adds another challenge to our problem.

3.2 Named entity extraction

Many tools and services have been developed for the NER task (such as Stanford NER, AlchemyAPI, DBpedia Spotlight, OpenCalais, Zemanta, and many more) (Rizzo and Troncy 2011). The state-of-the-art NER systems for English news articles produce near-human performance. For example, the best system entering MUC-7 scored an F-measure of 93.39 per cent while human annotators scored 97.60 per cent and 96.95 per cent (Marsh and Perzanowski 1998).

In spite of this, few research efforts studied NER in tweets. Researchers either used off-the-shelf trained NLP tools known for formal text (such as POS tagging and statistical methods of extraction) or adapted those techniques to suit informal text of tweets.

In Ritter *et al.* (2011), the authors built an NLP pipeline to perform NER. The pipeline involves POS tagging, shallow parsing, and a novel SVM classifier that predicts the informativeness of capitalization in a tweet. It trains a CRF model with all the aforementioned features for NEE. For classification, LabeledLDA is applied where entity types are used as classes. A bag-of-words-based profile is generated for each entity type, and the same is done with each extracted mention. Classification is done based on the comparison of the two.

The contextual relationship between the microposts is considered by Jung (2012). The paper proposes merging the microtexts by discovering contextual relationship between the microtexts. A group of microtexts contextually linked with each other is

regarded as a microtext cluster. Once this microtext cluster is obtained, they expect that the performance of NER can be better. The authors provide some suggestions for Contextual closure, Microtext cluster, Semantic closure, Temporal closure, and Social closure. Those closures are used by Maximum Entropy for the NER task.

Similarly, Li *et al.* (2012) exploits the gregarious property in the local context derived from the Twitter stream in an unsupervised manner. The system first leverages the global context obtained from Wikipedia and Web N-Gram corpus to partition tweets into valid segments (phrases) using a dynamic programming algorithm. Each such tweet segment is a candidate NE. Afterwards, a ranking approach tries to rank segments according to their probability of being an NE. The highly ranked segments have a higher chance of being true NEs. Each segment is represented as a node in a graph, and using the Wikipedia and the context of tweet (adjacent nodes (segments)), a score is assigned to that segment if it is an NE or not.

Bontcheva *et al.* (2013) proposed TwitIE, an open-source NLP pipeline which is built on GATE ANNIE (Cunningham *et al.* 2002) and customized to microblog text. TwitIE is composed of a set of pipelined modules. The normalizer module tries to reduce the linguistic noise on tweets in two stages: first, the identification of orthographic errors in an input discourse, and second, the correction of these errors. TwitIE contains an adapted Stanford POS tagger, trained on tweets. As a result of the adaptation in the earlier components, TwitIE demonstrates a significant F1 performance increase compared to ANNIE. LODIE (Derczynski and Bontcheva 2013) extends the TwitIE system to include entity disambiguation and event extraction.

4 Generic open world named entity disambiguation approach

We conclude from the previous section that almost all NED approaches in tweets are entity oriented. In contrast, we present a generic open world approach for NED for any NE based on the mention context.

First, let us formalize the problem. Given a mention m_i that belongs to a tweet t, the goal is to find a ranked list of entity home pages e_{ij} that represent m_i. We make use of the context of the mention $\{w\} = \{m_i, w_1, w_2, ..w_n\}$ to find the best entity candidate where $\{w\}$ is the set of words in the tweet after stop word removal. A set of features is extracted for each e_{ij} measuring how related it is to m_i and its context. An SVM classifier with Radial Basis Functions kernel is trained on a set of manually disambiguated mentions. The resulting SVM model is used for ranking entity pages for unseen mentions (new mentions which are not used in the training process).

Figure 2 illustrates our process of NED in tweets. The system is composed of three modules: the matcher, the feature extractor, and the SVM ranker. We describe each module separately.

4.1 Matcher

This module contains two submodules: Google API, and YAGO KB. Google API is a service provided by Google to enable developers to use Google products

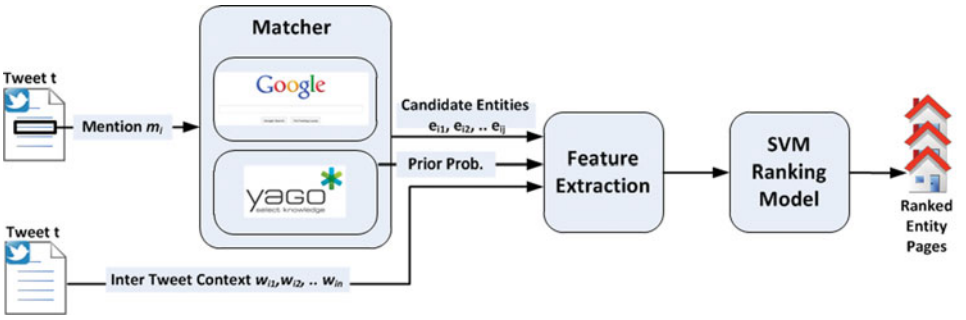

Fig. 2. (Colour online) Disambiguation system architecture.

from their applications. YAGO KB is a knowledge base constructed from Wikipedia, GeoNames[7], and WordNet[8]. It contains more than 447 million facts for 9.8 million entities. A fact is a tuple representing a relation between two entities. YAGO has about 100 relation types, such as `hasWonPrize`, `isKnownFor`, and `isLocatedIn`. Furthermore, it contains relation types connecting mentions to entities such as `hasPreferredName`, `means`, and `isCalled`. The `means` relation represents the relation between the entity and all possible mention representations in Wikipedia. For example, the mentions {'*Chris Ronaldo*', '*Christiano*', '*Golden Boy*', '*Cristiano Ronaldo dos Santos Aveiro*'} and many more are all related to the entity '`http://en.wikipedia.org/wiki/Cristiano_Ronaldo`' through the `means` relation.

We use mention m_i as an input query for the Google API module. The top 18 Web pages retrieved by Google are considered candidate entities for the mention m_i. To enlarge the search space of candidate entity home pages, we query YAGO KB for possible entities for that mention. Instead of taking all candidate entities related to that mention, which will increase our search space dramatically, we just take the set of candidates with top prior probabilities. Prior probability represents the popularity for mapping a name to an entity. YAGO calculates these prior probabilities by counting, for each mention that constitutes an anchor text in Wikipedia, how often it refers to a particular entity. We sort the entity candidates in descending order according to their prior probability. We select the top candidates satisfying the following condition:

$$\frac{Prior(e_{ij})}{Maximum(Prior(e_{ij}))} > 0.2. \tag{1}$$

This rule cuts out the long tail of candidates with low prior probability. It produces a variable size set of most probable entities into our search space instead of just considering a fixed number of top entities or using a fixed cutting threshold. The higher the $Maximum(Prior(e_{ij}))$, the fewer the entities to be included as candidates and vice versa. For example, if the maximum prior probability of an entity candidate given a mention is 0.8, then we consider only those candidates with a prior

[7] `http://www.geonames.org/`
[8] `http://wordnet.princeton.edu/`

Table 2. *URL features*

Feature name	Feature description
URL length	The length of URL.
Mention-URL Similarity	String similarity between the mention and the URL domain name (for non-Wikipedia pages) or the Wikipedia entity name (for Wikipedia pages) based on Dice Coefficient Strategy (Dice 1945).
Is mention contained	Whether or not the mention is contained in the whole URL.
Google page rank	The page order as retrieved by Google. Wikipedia pages from the YAGO KB are assigned a rank higher than the pages retrieved from Google.
Title keywords	Whether or not the page title contains keywords such as ('Official' or 'Home page').
#Slashes	Path length of the page (i.e., number of slashes in the URL).

probability above 0.16; candidates with a prior probability below or equal 0.16 are not considered. According to a statistical evaluation on the three used data sets, we found that the threshold 0.2 ensures that 91 per cent to 97 per cent of the correct Wikipedia entities are included in the matched YAGO candidates. On the other hand, it reduces the search space by 96 per cent on average in case we consider all the matched YAGO candidates. We believe that the obtained results are not sensitive to small changes in this threshold.

For the thus selected entities in YAGO, we add their Wikipedia articles to the set of candidate Web pages retrieved from Google. For each candidate page, we extract its title, description, keywords, and textual content. Title, description and keywords are extracted from the metadata of the HTML page, while the textual content is extracted using the HtmlUnit library.[9] The main limitation of HtmlUnit is that it is compute intensive. More than 90 per cent of the processing time for both the mention extraction and disambiguation tasks, is spent on running HtmlUnit over the candidate pages. Our focus in this paper is on the quality of the results rather than efficiency. It is possible, however, to improve the process in various ways such as replacing it with another HTML rendering component or by distributing processing over multiple nodes.

4.2 Feature extractor

This module is responsible for extracting a set of contextual and URL features that correlate with the likelihood that the candidate entity is a good representative for the mention. The SVM is trained on these features. For each candidate page, we extract the set of URL features shown in Table 2. In addition to these features, the following context features are also extracted:

[9] `http://htmlunit.sourceforge.net/`

- **Language Model (LM):** We use a smoothed unigram LM (Zhai and Lafferty 2001). We treat the mention along with the words of its tweet as a query and the entity pages as documents. The probability of a document being relevant to the query is calculated as follows:

$$\log P(q|d) = \sum_{w \in q,d} \log \frac{P_s(w|d)}{\alpha_d P(w|c)} + \sum_{w \in q} \log P(w|c) + n \log \alpha_d \tag{2}$$

where $q = \{m_i, w_{i1}, ..w_{in}\}$, d is the e_{ij} candidate page, c is the collection of all the candidate pages for m_i, n is the query length and α_d is document length normalization factor, $P(w|c)$ is the collection LM and $P_s(w|d)$ is the Dirichlet conjugate prior (MacKay and Peto 1994). These probabilities can be calculated as follows:

$$P(w|c) = \frac{tf(w,c)}{c_s} \tag{3}$$

$$P_s(w|d) = \frac{tf(w,d) + \mu P(w|c)}{|D| + \mu} \tag{4}$$

where tf is the term frequency of the word w in document d or in the entire collection c, c_s is raw collection size (total number of tokens in the collection), and μ is a smoothing parameter for which we use the average document length in the collection c. We calculated a separate LM for each of the parts of the entity pages (the title, description (page meta data), keywords (page meta data), and content).

- **Tweet-Page Overlap:**
The difference in length between Wikipedia pages and non-Wikipedia pages in addition to the document length normalization in the LM, leads to a system that favors long documents (typically Wikipedia pages) over short documents (typically non-Wikipedia pages). Therefore, we looked for another feature that is indifferent to the length of the pages. The feature Tweet-Page Overlap is inspired by Jaccard distance, with the difference that our version disregards page length. The feature is defined as the count of the words that occur in both the query q (the words of the tweet) and the document d (disregarding stop words):

$$Overlap(q,d) = |q \cap d| \tag{5}$$

Again, calculating it for page title, description, keywords, and content results in 4 versions of the feature.

- **Entity Prior Probability:** This is a value provided by the YAGO KB as described in Section 4.1. Only Wikipedia pages have prior probabilities. Non-Wikipedia pages are just assigned zero for this feature.

4.3 SVM ranker

To rank the candidate entities of a mention, an SVM classifier (Chang and Lin 2011) with a Radial Basis Functions kernel function is trained on the aforementioned set of features. Although our problem is to find the relevant (and irrelevant) entity pages

of a mention, we trained the SVM to distinguish between three types of entity pages: Wikipedia home page (Wiki entity), non-Wikipedia home page (Non-Wiki entity), and nonrelevant page. The reason behind this distinction comes from an observation we made while we manually analyzed our data sets. We found that Wikipedia home pages and non-Wikipedia home pages have quite different characteristics. In one of our test collections (Brian Collection), the mean number of tokens in Wikipedia home pages is about 9434 while it is about 653 for non-Wikipedia home pages. Wikipedia home pages have richer contents and thus context features are likely to correlate better with the relevance of a Wikipedia page to the mention context. On the other side, non-Wikipedia home pages tend to be short and sometimes with almost no content. In this class, URL features are more indicative for the relevancy of an entity page of a mention. The distinction in classes enables the classifier to choose the best features for each class.

In addition, we search the Wikipedia page infobox for a home page URL for the entity. If found, we remove that home page from the candidate list. For example, for the mention '*Barcelona*', if we find among the candidate pages, the Wikipedia page '`http://en.wikipedia.org/wiki/FC_Barcelona`' and the official site for '*Barcelona*' '`http://www.fcbarcelona.com/`', we remove the latter page from the candidates list. The idea behind this action is that our training data is annotated by assigning only one entity page for each mention with a preference for Wikipedia pages. We do not want to confuse the classifier by assigning a nonrelevant class to a home page for one mention and assigning a relevant class for a home page of another mention that does not have a Wikipedia entity.

The SVM classifier provides a probability for how likely the input feature vector represents each of the aforementioned classes. We used the SVM probability estimation approach presented by Wu, Lin and Weng (2004) and implemented in Chang and Lin (2011). Due to the imbalance in the training data between the first two classes and the third (only one page is assigned to the mention and the rest is treated as nonrelevant page), the probabilities of the majority class (nonrelevant) are dominating. We overcome this problem by dealing with the task as a ranking task instead of a hard classification task.

For testing and evaluating, we rank the mention's candidate pages according to the probabilities of the two relevant classes. Evaluation is done by looking at the recall of finding the correct entity page of the mention in the top k results.

4.4 Experiments

4.4.1 Data sets

To validate our approach, we use three collections of tweets[10]. The first two data sets are mainly designed for an NER task. We manually construct the NED ground truth by linking each NE to only one appropriate entity page. We give higher priority

[10] Our data sets are available at `https://github.com/badiehm/TwitterNEED`

Table 3. *Data sets statistics*

	Brian Collection	Mena Collection	#Microposts Collection
#Tweets	1,603	162	2,340
#Mentions	1,585	510	3,819
#Wiki entities	1,233 (78 per cent)	483 (94 per cent)	3,819 (100 per cent)
#Non-Wiki entities	274 (17 per cent)	19 (4 per cent)	0 (0 per cent)
#Mentions with no entity	78 (5 per cent)	8 (2 per cent)	0 (0 per cent)

to Wikipedia pages. When no Wikipedia page exists for a mention, we link it to a non-Wikipedia home page or profile page.

The first data set (Brian collection) is the one used in Locke and Martin (2009). The data set is composed of four subsets of tweets; one public timeline subset and three subsets of tweets revolving around economic recession, Australian Bushfires and a gas explosion in Bozeman, MT.

The second data set (Mena collection) is the one used in Habib and van Keulen (2012b) which is relatively small in the number of tweets but rich in the number of NEs. It is composed mainly from tweeted news about sportsmen, celebrities, politics, etc.

The third data set (#Microposts collection) is the training data set provided by the #Microposts Named Entity Extraction & Linking (NEEL) Challenge (Cano Basave *et al.* 2014). The NEEL Challenge task required participants to build systems to extract entity mentions from a tweet and to link the extracted mentions to the English DBpedia v3.9 resource. Note that this data set does not contain any non-Wikipedia entities. The collection covers event-annotated tweets collected for the period 15 July 2011 to 15 August 2011 (31 days). It extends over multiple notable events, including the death of Amy Winehouse, the London Riots, and the Oslo bombing. We have done the mapping from the YAGO KB to DBpedia by identifying the Wikipedia page as a common property for the identical entities.

Statistics about the three data sets are shown in Table 3. The three collections are representatives for two types of tweets: formal news tweets (Mena collection) and user-generated tweets that discuss events (Brian collection) and a mixture of both types (#Microposts collection).

4.4.2 Experimental setup

As mentioned in Section 4.1, the top eighteen Web pages retrieved by Google are considered candidate entities for each mention m_i. Our choice for eighteen is driven by the distribution analysis of Google's correct entity rank presented in Figure 3. To compute the distribution, we checked the first fifty page results by Google and determined at which rank the correct entity was located (if any). From the graph, the threshold eighteen seems to be a good point to cut out the long tail of Google results for the three data sets. Considering the top eighteen pages is enough to ensure

Table 4. *Candidate pages for the mention 'Houston'*

http://www.houstontx.gov/
http://en.wikipedia.org/wiki/Houston
http://www.visithoustontexas.com/
http://www.chron.com/
http://www.tripadvisor.com/Tourism-g56003-Houston_Texas-Vacations.html
http://www.forbes.com/places/tx/houston/
http://www.nba.com/rockets/
http://www.uh.edu/
http://www.houstontexans.com/
http://www.houston.org/
http://www.citypass.com/houston
http://www.portofhouston.com/
http://www.hillstone.com/
http://wikitravel.org/en/Houston
http://houston.craigslist.org/
http://houston.astros.mlb.com/

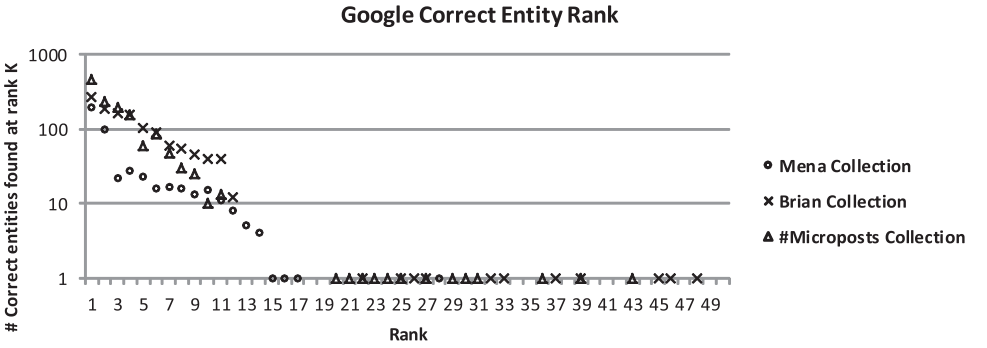

Fig. 3. Distribution analysis of the Google's correct entity rank.

that the correct entity page is covered by the Google candidates without adding unnecessary false candidates.

All our experiments are done with a 4-fold cross validation approach for training and testing the SVM. Our evaluation measure is the recall of finding the correct entity page of a mention in the top-k results. We evaluated rankings up to $k = 5$. The reason behind focusing on recall instead of precision is that our ground truth for the disambiguation task links the mention to only one entity page. At the same time, we cannot completely consider the other retrieved pages as nonrelevant. In some cases, there may exist more than one relevant page among the candidate pages for a given mention. So that, as we link each mention to only one entity page, it is unfair to count the other pages as negatives. For example, Table 4 shows some candidate pages for the mention '*Houston*'. Although we link this mention to the Wikipedia page `http://en.wikipedia.org/wiki/Houston`, we do not consider the other pages (such as `http://www.houstontx.gov/` and `http://wikitravel.org/en/Houston`) that appear in the top-k as nonrelevant. Following this reasoning, we consider results

Table 5. *Baselines and upper bounds*

	Brian Collection	Mena Collection	#Microposts Collection
Prior	846 (53 per cent)	394 (77 per cent)	2124 (55 per cent)
AIDA	766 (48 per cent)	389 (76 per cent)	2145 (56 per cent)
Google 1st rank	269 (17 per cent)	197 (39 per cent)	458 (11 per cent)
YAGO coverage	990 (62 per cent)	449 (88 per cent)	3187 (83 per cent)
Google coverage for:			
All entities	1218 (77 per cent)	476 (93 per cent)	1308 (34 per cent)
Wikipedia entities	1077 (87 per cent)	462 (96 per cent)	1308 (34 per cent)
Non-Wikipedia entities	141 (51 per cent)	14 (74 per cent)	–

at rank 1 to be the measure of the average precision (P@1) of the disambiguation process.

4.4.3 Baselines and upper bounds

Table 5 shows our baselines and upper bounds in terms of the percentage of finding the correct entity page of a mention, in addition to some statistics about the Google page rank of the correct entity pages. Three baselines are defined. The first is the Prior, which represents the disambiguation results if we just pick the YAGO entity with the highest prior for a given mention. The second is the AIDA disambiguation system (Yosef *et al.* 2011). The third is Google 1st rank, which represents the results if we pick the Google 1st ranked page as the correct entity page. It might be surprising that AIDA gives worse results than one of its components which is Prior. The reason behind this is that in AIDA matching of mentions is case sensitive and thus could not find entities for lower case mentions. It was not possible to turn all mentions to upper case initials because some mentions should be in all upper case to be matched (such as 'USA'). For the Prior baseline, we perform case-insensitive matching. AIDA and Prior are upper bounded by the YAGO coverage for the mention-entity pairs. Coverage means how many mention-entity pairs of our ground truth exist in the KB. Note that more mentions might have a Wikipedia entity but it is not covered in YAGO because it does not have the proper surface mention (such as '*Win7Beta*'). For our results, we have an upper bound which we cannot exceed, which is the coverage of the search space of set of candidates retrieved by Google and enriched through the KB. They do not cover our ground truth annotations completely.

4.4.4 Feature evaluation

To evaluate the importance of each of the two feature sets used, we conduct an experiment to measure the effect of each individual feature set on the disambiguation results. Table 6 shows the disambiguation results on our data sets using each of the introduced feature sets. It also shows the effect of each feature set on both types of entities, Wiki and Non-Wiki.

Table 6. *NED results at rank k using different feature sets*

(a) Brian collection

	All entities			Wiki entities			Non-Wiki entities		
Rank	All	Context	URL	All	Context	URL	All	Context	URL
1	65.30 per cent	61.64 per cent	61.83 per cent	78.91 per cent	78.18 per cent	74.70 per cent	22.99 per cent	4.38 per cent	21.17 per cent
2	70.73 per cent	64.98 per cent	67.00 per cent	83.70 per cent	81.27 per cent	79.97 per cent	32.12 per cent	9.85 per cent	27.37 per cent
3	71.86 per cent	67.07 per cent	69.91 per cent	84.51 per cent	82.89 per cent	82.16 per cent	35.04 per cent	14.60 per cent	34.31 per cent
4	73.00 per cent	69.02 per cent	70.54 per cent	84.91 per cent	84.02 per cent	82.24 per cent	39.78 per cent	20.80 per cent	37.59 per cent
5	73.69 per cent	70.66 per cent	70.91 per cent	85.48 per cent	85.16 per cent	82.48 per cent	41.24 per cent	25.18 per cent	38.69 per cent

(b) Mena collection

	All entities			Wiki entities			Non-Wiki entities		
Rank	All	Context	URL	All	Context	URL	All	Context	URL
1	85.10 per cent	82.16per cent	60.59 per cent	88.82 per cent	86.54 per cent	62.73 per cent	26.32per cent	5.26 per cent	31.58 per cent
2	89.22 per cent	86.47 per cent	70.20 per cent	92.96 per cent	90.68 per cent	72.67 per cent	31.58 per cent	15.79 per cent	36.84 per cent
3	90.78 per cent	87.65 per cent	71.96 per cent	94.20 per cent	91.93 per cent	74.53 per cent	42.11 per cent	15.79 per cent	36.84 per cent
4	91.37 per cent	88.63 per cent	74.12 per cent	94.41 per cent	92.75 per cent	76.60 per cent	52.63 per cent	21.05 per cent	42.11 per cent
5	91.96 per cent	89.41 per cent	75.10 per cent	95.03 per cent	93.37 per cent	77.64 per cent	52.63 per cent	26.32 per cent	42.11 per cent

(c) #Microposts collection

	All entities (Wiki entities)		
Rank	All	Context	URL
1	70.98 per cent	66.45 per cent	63.36 per cent
2	76.13 per cent	73.59 per cent	73.15 per cent
3	77.39 per cent	76.48 per cent	76.13 per cent
4	77.76 per cent	77.65 per cent	76.76 per cent
5	77.86 per cent	77.86 per cent	76.95 per cent

By comparing results of Wiki and Non-Wiki entities, we can see that context features are more effective than URL features in finding Wiki entities while URL features are more powerful in finding Non-Wiki entities.

Although Wikipedia URLs are quite informative, the context features have more data to be investigated and used in the selection and ranking of candidate pages than the URL features. Furthermore, some Wiki URLs are not informative for the given mention. For example, the mention '*Qld*' refers to the Wikipedia entity '`http://en.wikipedia.org/wiki/Queensland`' which is not informative regarding the input mention. This is why the context features are more effective than the URL features in finding Wiki entities.

On the other hand, the context features are less effective than the URL features in finding Non-Wiki entities because many home pages are either developed in flash or have some graphics content and hence contain less textual content to be used.

From Table 6, it is also clear that the usage of both sets of features yields better entity disambiguation results. Compared to Table 5, our approach shows improvements on the disambiguation results for all entities by 12 per cent on Brian collection, by 8 per cent on Mena collection, and by 15 per cent on #Microposts collection over the best baseline at rank $k = 1$. At rank $k = 5$, the improvements over the best baseline are 21 per cent, 15 per cent, and 22 per cent, respectively.

For a fair comparison with the AIDA system, we should only compare the AIDA results against our disambiguation results for Wiki entities at rank 1 (P@1) because AIDA is only capable of disambiguating Wiki entities. Furthermore, AIDA only links mentions to one entity. At rank $k = 1$, our approach shows significant improvement on disambiguation results for Wiki entities over AIDA by 30 per cent (from 48 per cent to 78.91 per cent) on Brian collection, by 12 per cent (from 76 per cent to 88.82 per cent) on Mena collection, and by 15 per cent (from 56 per cent to 70.98 per cent) on #Microposts collection.

To evaluate the impact of each distinct feature described in Section 4.2 on the disambiguation results, we measure the information gain ratio (Abeel, Van de Peer and Saeys 2009) of each feature. Table 7 shows the feature list ordered on the calculated gain score. The higher the score, the more important is the feature. The scores are calculated for the Brian collection data set as it has more diversity in entity types than the other two collections.

4.4.5 Error analysis

By going through the output of our approach, we found three main sources of errors. The first source of errors comes from the fact that our search space does not always contain the correct entity among the candidate list of entities. This represents around 50 per cent of erroneous cases. The second source comes from the domain similarity among the candidate entities for a given mention. For example, the mention '*NCAA Basketball*' in the tweet '***Notre Dame*** *vs* ***Marquette*** *Live* ***NCAA Basketball*** *Stream Online TV Link … : Watch basketball match live online on yo…http://bit.ly/gxvDL3*' is linked to the '`http://en.wikipedia.org/wiki/NCAA_Basketball_series`' while

Table 7. *Information gain of the disambiguation features*

Feature name	Feature gain score
Prior	0.1058
Content tweet-page overlap	0.0523
Google page rank	0.0256
Description LM	0.0252
URL length	0.0242
Content LM	0.0216
Mention-URL similarity	0.0214
#Slash	0.0203
KeyWords LM	0.0168
Description tweet-page overlap	0.0127
KeyWords tweet-page overlap	0.0119
Title keywords	0.0104
Title LM	0.0058
Title tweet-page overlap	0.0045
Is mention contained	0.0028

it should be linked to '`http://en.wikipedia.org/wiki/NCAA_Men's_Division_I_Basketball_Championship`'. We can see that the correct entity and the falsely assigned one have the same domain and are similar in their page contents. This source of errors represents around 45 per cent of misclassified cases. Another source of errors is those mentions that should not be linked to any entity (null entity assignment) as our system always assigns an entity to any given mention. For example, in the tweet '*RT @WirelessFest: Congrats to **Greg Snelgrove** who is the winner of our '**Festival Survey**' prize draw! Keep your eyes peeled for more compe …'*. Our ground truth annotators were not able to find any suitable pages for the given mentions. However, our system links each of them to one of the candidate pages retrieved by Google for these mentions. This type of errors represents around 5 per cent of the cases.

5 Hybrid named entity extraction approach

As stated before, we do the extraction in two phases, one phase before the disambiguation and one after. The first phase aims to generate as many NE candidates as possible to achieve a high recall. The second phase that filters false positives from those candidates comes after the disambiguation process. The whole extraction process is shown in Figure 4.

5.1 Named entity candidates generation

The main goal of this phase is to achieve high recall and of course this results in low precision. It is the task of the second phase (NE candidates filtering) to improve precision without risking much damage to recall.

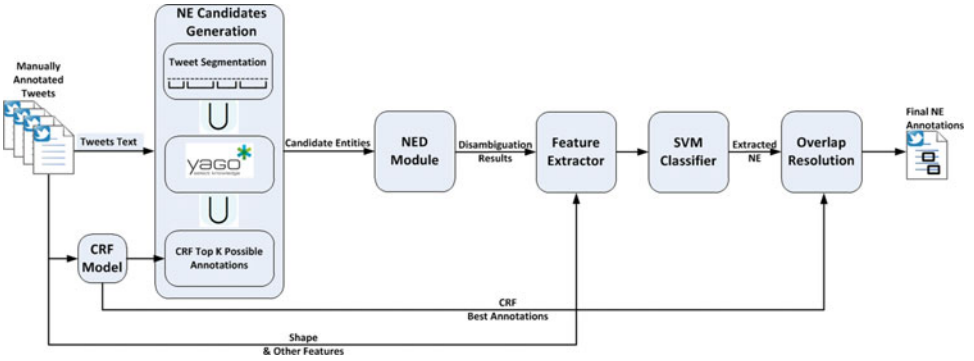

Fig. 4. (Colour online) Extraction system architecture.

For candidate generation phase, we use the following methods:

- **Tweet Segmentation**: Tweet text is segmented using the segmentation algorithm described in Li *et al.* (2012). Each segment is considered a candidate for a NE. The segmentation approach splits the tweet text based on a stickiness function. A high stickiness score of segments indicates that it is not suitable to further split segments, as it breaks the correct word collocation. In other words, a high stickiness value indicates that a segment cannot be further split at any internal position. More formally, given a tweet of four words $w_1w_2w_3w_4$, we segment it as $w_1w_2||w_3w_4$ rather than $w_1||w_2w_3w_4$, if $C(w_1w_2) + C(w_3w_4) > C(w_1) + C(w_2w_3w_4)$, where C(.) basically captures the probability being a valid phrase of a segment. Microsoft Web N-Gram (Wang *et al.* 2010) and YAGO KB (which represents Wikipedia normalization of the stickiness function) are used as described in Li *et al.* (2012). We prefer this tweet segmentation approach over using noun phrases, because NEs are not always noun phrases: Book, movie, and song titles may be verb phrases (for example, the movie title '*Catch me if you can*'). Tweet segmentation has a higher chance for correctly handling those cases.
- **KB Lookup**: List lookup is an established method of performing NEE by matching n-grams of a document's content against the mention-entity table of a KB such as YAGO or DBpedia. Due to the shortness of the messages and the informal nature of the used language, KB lookup is a suitable method for NEE in tweets or other short-message contexts.
- **CRF Alternative Annotations**: CRF is a probabilistic model that is widely used for NER (McCallum and Li 2003). Despite the successes of CRF, the standard training of CRF can be very expensive due to global normalization (Sutton and McCallum 2005). In our approach, we used an alternative method called *empirical training* (Zhu *et al.* 2013) to train a CRF model. The Maximum Likelihood Estimation of the empirical training has a closed form solution and does not need iterative optimization nor global normalization. Therefore, empirical training can be radically faster than standard CRF training. Furthermore, the Maximum Likelihood Estimation of the empirical

training is also a Maximum Likelihood Estimation of the standard training. Hence, it can obtain competitive precision to the standard training.

Tweet text is tokenized using a special tweet tokenizer (Gimpel *et al.* 2011). For each token, the following features are extracted and used to train the CRF:

— The POS tag of the token provided by a special POS tagger designed for tweets (Gimpel *et al.* 2011).
— Whether the token's initial is capitalized.
— Whether the token's characters are all capitalized.
— Whether the token has any capital letters.

We used the IOB representation for the training annotations. Tokens that represent the start of an NE are annotated with *B-NE*. Tokens that represent the inner part of an NE are annotated with *I-NE*. Finally, tokens that do not belong to an NE are annotated with *O*. The CRF model is trained and used to provide not only the best annotation set for the tweet text but also top-k possible annotation sets.

A naive implementation for obtaining the k most probable annotation sets for $p(S|O)$, where O is the observation sequence (tweet tokens) and S is the tag sequence (annotations), is to calculate probabilities for all possible S, sort them by probability and select the k most probable S. Unfortunately, such a naive implementation is space and computation inefficient. To illustrate this, let $O = \{o_1, o_2, \ldots, o_n\}$ and $S = \{s_1, s_2, \ldots, s_m\}$. Suppose the size of the tag space is $|s|$, then there can be as many as $|s|^n$ possible annotation sets. Exhaustive search through the complete annotation set space becomes impractical with growing $|s|$ and n.

In our work, we constrain the complete annotation set space to a promising subspace with unary and pairwise constraints. According to the Co-occurrence Rate Factorization (Zhu *et al.* 2012), $p(S|O)$ can be factorized as follows:

$$p(S|O) = \prod_{i=1}^{n} p(s_i|o_i) \prod_{j=1}^{n-1} \mathtt{CR}(s_{j-1}; s_j|o_{j-1}, o_j).$$

We impose unary constraints to the unary factors $p(s_i|o_i)$, i.e., for an observation o_i, we only consider the top-k most possible s_i. Similarly, for the pairwise factors $\mathtt{CR}(s_{j-1}; s_j|o_{j-1}, o_j)$, we only consider the top-k most probable $(s_{j-1}; s_j)$ for (o_{j-1}, o_j). Hence, there can be at most $k^{\frac{1}{2}n}$ annotation sets in the promising subspace. In practice, this works well. Note that the reduction of the complete path space to a promising subspace may lead to the exclusion of the best annotation set from the promising subspace. This is very rare though on real-world data sets. We remedy it anyway by explicitly adding the best answer set to the top-k paths.

As a post processing step for the candidate generation phase, we remove duplicate candidates. Furthermore, to improve precision, we apply certain filtering heuristics such as removing segments that are composed of stop words or that contain a term with a verb POS tag.

5.2 Named entity candidates filtering

After generating the candidate list of NEs, we apply our disambiguation approach as described in Section 4 to disambiguate each extracted NE candidate. After this disambiguation phase, we use an SVM classifier with an Radial Basis Functions kernel to predict which candidates are true positives and which ones are not. We use the following set of features:

- **Shape Features**: The features used to train the CRF model as presented in Section 5.1.
- **Probabilistic Features**:
  - — The joint and conditional probability of the candidate obtained from the Microsoft Web N-Gram service.
  - — The stickiness of the segment as described in Li *et al.* (2012).
  - — The segment frequency over around 5 million tweets[11].
  - — The extraction confidence for the candidate, if it was extracted by the CRF.
- **KB Features**:
  - — Whether the segment appears in WordNet.
  - — Whether the segment appears in the YAGO mention-entity look-up table.
- **Disambiguation Features**:
  - — All the features described in Section 4.2 derived from the top-ranked entity page selected for the given NE candidate. We consider the features for the top-ranked page only rather than features of pages at lower rank, because those are the most representative and expressive for their entity.
  - — Whether any of the candidate entity pages for the given NE candidate is a Twitter, Facebook, Linkedin, ebay, or IMDB page.

5.3 Final named entity set generation

Finally, we take the union of the best CRF annotation set and SVM results, after removing duplicate extractions, to get the final set of annotations. To resolve overlapping annotations, we select the entity that appears in YAGO; if both entity mentions exist in YAGO, then we select the longest one.

5.4 Experimental results

In this section, we present the results of experiments with the presented extraction methods applied on five different collections of tweets. The goal of the experiments is to investigate effectiveness of our approach in comparison with a state-of-the-art extraction approach (Stanford NER (Finkel, Grenager and Manning 2005)) and a competitor (Ritter_NER approach (Ritter *et al.* 2011)). Furthermore, we present a combined evaluation for both our extraction and disambiguation approaches in comparison with two competitors (AIDA (Yosef *et al.* 2011) and DBpedia Spotlight (Mendes *et al.* 2011)) applied on the three data sets described in Section 4.4.1.

[11] `http://wis.ewi.tudelft.nl/umap2011/` + TREC 2011 Microblog track collection.

5.4.1 Data sets

In addition to the three data sets described in Section 4.4.1, we use two other Twitter data sets named Ritter and #MSM. The Ritter collection[12] (Ritter *et al.* 2011) consists of 2394 tweets with 1495 annotated NE mentions. The #MSM collection[13] is provided by the 'Making Sense of Microposts Challenge' of 2013 (Basave *et al.* 2013). The collection consists of 2815 tweets with 2987 NE mention annotations.

5.4.2 Extraction evaluation

In this experiment, we evaluate several extraction techniques on our data sets:

- **Stanford**: A Stanford NER (Finkel *et al.* 2005) model trained on a normal CoNLL collection. It is based on a CRF model which incorporates long-distance information. It has been shown to obtain good performance consistently across different domains.
- **Stanford_Caseless**: The Stanford NER caseless model which is a NER model that ignores the capitalization feature).
- **Stanford_CRF**: The Stanford CRF model trained and tested on our ground truth collections using 4-fold cross validation.
- **Ritter_NER**: A system that uses a set of features that includes orthographic and dictionary features (Ritter *et al.* 2011).
- **TwitterNEED**: Our method. We distinguish the different phases of the extraction process: the candidate generation phase **(CG)**, the candidate filtering phase **(SVM_CF)**, the best CRF annotation set **(Best_CRF)**, and the final NE set generation **(SVM∪CRF)**.

The SVM is trained and tested using 4-fold cross validation. Three folds are used to train the NED and NEE models while the fourth is used for validation. Since the Ritter and #MSM collections do not have a ground truth for NED to be used for training, we use a disambiguation model trained on the other collections (Mena and Brian) instead. For the #Microposts collection, we also use regular expressions to extract numbers, dates, and URLs from the tweet text as these entity types are among the taxonomy of the #Microposts NEEL challenge.

5.4.3 Important evaluation details

The concept of NE is not an exactly defined commonly agreed upon notion. This has its effect on the ground truth, because annotators may disagree. For example, the tweet '*RT @BBCClick: Joy! MS Office now syncs with Google Docs (well, in beta anyway). We are soon to be one big happy (cont) http://tl.gd/73t94u*'. Annotators may disagree about whether '*Office*' and '*Docs*' are part of the mentions '*MS*' and '*Google*' or not.

[12] `https://github.com/aritter/twitter_nlp`
[13] `http://oak.dcs.shef.ac.uk/msm2013/`

This is why we prefer to use the extraction evaluation strategy introduced by GATE[14] which computes three variants for each of the precision, recall, and F1 measures named *strict*, lenient, and average.

strict Only perfectly matching annotations are counted as correct.
lenient Partially matching annotations are counted as correct as well.
average The average of the strict and the lenient scores. This is equivalent with counting a partially correct annotation as 0.5.

The percentage of the partially overlapping annotations produced by the SVM_CRF method on the #Microposts collection is about 11 per cent of the ground truth annotations.

It is also important to note that we evaluate only the ability of the different systems to extract mentions of NEs, but not classification (into person, organization, location, etc.). We believe that the NED phase already gives a fine-grained classification by linking the mentions to their entities which typically contains type information, instead of just doing a classification for their entity type.

5.4.4 Discussion

Table 8 shows the performance of the TwitterNEED approach in comparison with the other competitors (Stanford and Ritter). From the shown results, we observe the following:

- The results of the Ritter_NER approach on the Ritter collection, shows only the strict precision, recall, and F1. This is because (Ritter *et al.* 2011) used only the strict strategy for extraction evaluation and the shown values are taken from Ritter's own evaluation. The model provided by Ritter[15] has been already trained on the whole Ritter collection, so that applying the provided model on the collection would not be unfair.
- The F1 results of TwitterNEED outperform all the Stanford models including Stanford_CRF trained on our collections. TwitterNEED achieves on average a 10 per cent improvement over Stanford, 18 per cent over Stanford_Caseless, 5 per cent over Stanford_CRF, and 15 per cent over Ritter_NER.
- TwitterNEED outperforms Ritter_NER on four out of five collections. It only fails to outperform Ritter_NER on Ritter's own collection. This is because Ritter_NER is a pipeline of classifiers (POS, Capitalization reliability, Chunker) trained and used on the same collection. In contrast, our NED model trained on the Mena and Brian collections is used on the #MSM collection and contributes in improving the extraction performance. This demonstrates that our models are generic and not restricted to the collection used for training.
- The effect of taking the union of the SVM and CRF outputs can more clearly be seen in the strict results than in the lenient results where the improvement of the SVM over the CRF results is low. The reason is that the SVM is

[14] `http://gate.ac.uk/`
[15] `https://github.com/aritter/twitter_nlp`

Table 8. *Evaluation of NEE*

(a) Brian collection

	Lenient			Average			Strict		
	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1
Stanford	0.7676	0.6877	0.7255	0.6799	0.6038	0.6396	0.5907	0.5199	0.5530
Stanford_Caseless	0.6895	0.6151	0.6502	0.6248	0.5558	0.5883	0.5597	0.4965	0.5262
Stanford_CRF	0.8447	0.7953	0.8190	0.7972	0.7427	0.7688	0.7487	0.6901	0.7180
Ritter_NER	0.6713	0.6005	0.6339	0.6145	0.5414	0.5756	0.5559	0.4824	0.5165
TwitterNEED:									
CG	0.1746	**0.9905**	0.2969	0.1627	**0.9609**	0.2783	0.1517	**0.9312**	0.2609
SVM_CF	**0.9033**	0.7016	0.7898	0.8721	0.6864	0.7682	**0.8418**	0.6713	0.7469
Best_CRF	0.8783	0.8379	0.8576	**0.8308**	0.7855	0.8075	0.7825	0.7331	0.7570
SVM∪CRF	0.8425	0.8738	**0.8579**	0.8056	0.8353	**0.8202**	0.7687	0.7968	**0.7825**

(b) Mena collection

	Lenient			Average			Strict		
	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1
Stanford	0.8941	0.9275	0.9105	0.8235	0.8461	0.8346	0.7514	0.7647	0.7580
Stanford_Caseless	0.7818	0.8078	0.7946	0.7248	0.7461	0.7353	0.6673	0.6843	0.6757
Stanford_CRF	0.8859	0.8863	0.8853	0.8084	0.8084	0.8078	0.7309	0.7306	0.7303
Ritter_NER	0.9066	0.7055	0.7935	**0.8491**	0.6511	0.7370	0.7899	0.5966	0.6797
TwitterNEED:									
CG	0.4683	**0.9980**	0.6374	0.3815	**0.9676**	0.5473	0.3187	**0.9373**	0.4756
SVM_CF	**0.9330**	0.7647	0.8405	0.8659	0.7343	0.7947	**0.8031**	0.7039	0.7503
Best_CRF	0.9279	0.9078	0.9177	0.8333	0.8137	0.8234	0.7384	0.7196	0.7289
SVM∪CRF	0.8994	0.9471	**0.9226**	0.8344	0.8794	**0.8563**	0.7695	0.8118	**0.7901**

(c) #Microposts collection

	Lenient			Average			Strict		
	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1
Stanford	0.8153	0.5894	0.6842	0.7283	0.5251	0.6103	0.6409	0.4609	0.5362
Stanford_Caseless	0.8194	0.5048	0.6248	0.7288	0.4482	0.5550	0.6378	0.3915	0.4852
Stanford_CRF	**0.8677**	0.7252	**0.7900**	**0.7933**	0.6522	0.7158	0.7163	0.5793	0.6405
Ritter_NER	0.8192	0.4496	0.5806	0.7492	0.4070	0.5275	0.6777	0.3645	0.4740
TwitterNEED:									
CG	0.1361	**0.9629**	0.2385	0.1285	**0.9542**	0.2264	0.1209	**0.9455**	0.2144
SVM_CF	0.7788	0.5666	0.6560	0.7507	0.5558	0.6387	**0.7225**	0.5449	0.6213
Best_CRF	0.7448	0.6503	0.6943	0.7026	0.6093	0.6526	0.6605	0.5682	0.6109
SVM∪CRF	0.7660	0.7714	0.7687	0.7376	0.7391	**0.7383**	0.7093	0.7067	**0.7079**

Table 8. *Continued*

(d) Ritter collection

	Lenient			Average			Strict		
	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1
Stanford	0.6442	0.6903	0.6665	0.5684	0.6075	0.5873	0.4922	0.5247	0.5079
Stanford_Caseless	0.4381	0.5621	0.4924	0.3807	0.4877	0.4276	0.3231	0.4132	0.3626
Stanford_CRF	0.7600	0.6015	0.6704	0.7017	0.5528	0.6174	0.6429	0.5041	0.5642
Ritter_NER	–	–	–	–	–	–	**0.7300**	0.6100	**0.6700**
TwitterNEED:									
CG	0.1042	**0.9860**	0.1884	0.0946	**0.9326**	0.1718	0.0858	**0.8792**	0.1563
SVM_CF	**0.8189**	0.4920	0.6147	**0.7738**	0.4693	0.5843	0.7296	0.4466	0.5540
Best_CRF	0.7722	0.6742	0.7199	0.7057	0.6148	0.6572	0.6390	0.5554	0.5943
SVM∪CRF	0.7396	0.7336	**0.7366**	0.6843	0.6792	**0.6817**	0.6290	0.6248	0.6269

(e) #MSM collection

	Lenient			Average			Strict		
	Pre.	Rec.	F1	Pre.	Rec.	F1	Pre.	Rec.	F1
Stanford	0.7341	0.8305	0.7793	0.6728	0.7589	0.7133	0.6112	0.6872	0.6470
Stanford_Caseless	0.7281	0.7788	0.7526	0.6679	0.7122	0.6893	0.6073	0.6456	0.6259
Stanford_CRF	0.8745	0.7615	0.8141	0.8171	0.7090	0.7592	0.7592	0.6565	0.7041
Ritter_NER	0.7195	0.6302	0.6719	0.6982	0.6103	0.6513	0.6767	0.5904	0.6306
TwitterNEED:									
CG	0.2162	**0.9969**	0.3553	0.1936	**0.9652**	0.3225	0.1741	**0.9336**	0.2935
SVM_CF	**0.8840**	0.7358	0.8031	**0.8428**	0.7123	0.7721	**0.8028**	0.6888	0.7414
Best_CRF	0.8252	0.8803	**0.8519**	0.7722	0.8234	0.7970	0.7192	0.7665	0.7421
SVM∪CRF	0.8013	0.9088	0.8517	0.7588	0.8613	**0.8068**	0.7164	0.8139	**0.7620**

more capable of finding the exact annotation than the CRF which sometimes misses a part of the NE. For example, the tweet '***BBC**: **US** poet wins **Dylan Thomas prize** http://is.gd/ibWvK*', the CRF extracts the mentions '*BBC*', '*US*' and '*Dylan Thomas*'. On the other hand, the SVM is able to correctly classify '*BBC*' , '*US*' and '*Dylan Thomas prize*' as true positive NE. The NEs that are correctly classified by the SVM, match the exact manual annotations. This is due to the effective segmentation approach used in the candidate generation phase as well as the usage of the disambiguation features in classifying whether or not the segment represents an NE. The disambiguation module is able to find the correct page for the mention '*Dylan Thomas prize*' which is '`http://en.wikipedia.org/wiki/Dylan_Thomas_Prize`' and thus gives higher likelihood for '*Dylan Thomas prize*' to be classified as a true positive NE.

Furthermore, the SVM is able to extract some NEs that are missed completely by the CRF. For example, in the tweet '*@ABC U destroyed #DWTS for*

*this----> **Palin** Rips **American Idol**: **AP** got an advance copy of **Sarah Palin**'s new book... http://bit.ly/aGnwmh*', the CRF extracts only '*Sarah Palin*' while the SVM extracts '*Sarah Palin*', '*American Idol*', and '*AP*'.

Another example that shows the power of using the disambiguation features for improving the extraction, is the tweet '*RT @nytonline : **Pamela Anderson** to join **Indian BB** : Former **Baywatch** star **Pamela Anderson** is join the Indian version of **Big Brother**. http://ow.ly/19YFz1*'. While CRF extracts only the two mentions of '*Pamela Anderson*', the SVM was able to extract '*Baywatch*' and '*Big Brother*' in addition to the two mentions of '*Pamela Anderson*'. The NED model links the mention '*Big Brother*' to the entity home page '`http://en.wikipedia.org/wiki/Big_Brother_(UK)`' which is the correct entity page for this mention. The similarity between the tweet context and the entity page leads to correctly classifying the segment '*Big Brother*' as a true positive NE. Similarly, '*Baywatch*' is linked to the page '`https://en.wikipedia.org/wiki/Baywatch`' and correctly extracted by the SVM. The SVM has a higher capability of correctly classifying the highly ambiguous and informally represented entities as true positives.

5.4.5 Error analysis

There are still some cases where the system fails to extract the correct mentions (false negatives). In these cases, it is often not even clear to humans whether the phrase represents an NE or not. For example, it is hard to recognize '*The Day After*' as an NE (`http://en.wikipedia.org/wiki/The_Day_After`) in the tweet '***The Day After** in **#Manhattan #NYC (Broadway & 4th Street)** http://twitpic.com/3k97qp (via @mccarthy31)*'. The CRF does detect the mention '*The Day After*' because all its tokens are common words that are marked as nonentities in the training data. The SVM classifies '*The Day After*' as a non-NE because its linked entity '`http://en.wikipedia.org/wiki/The_Day_After`' has no similarity to the tweet context. This source of errors represents 40 per cent of the erroneous cases.

Another source of errors is where the system classifies a phrase as an NE while in fact it is not. The majority of these cases are concepts such as '*Retail Prices Index*', '*state of emergency*', or '*Breast Cancer*'. Although these are not NEs, they still have expressive Wikipedia articles that match the context of the tweet. This source of errors represents 60 per cent of the problematic cases. Having more false positives than false negatives results in achieving a slightly better recall than precision in most of the used data sets.

5.4.6 Combined extraction and disambiguation evaluation

In this experiment we compare the performance of TwitterNEED against two competitors: AIDA[16] and DBpedia Spotlight.[17] AIDA is primarily a disambig-

[16] `https://d5gate.ag5.mpi-sb.mpg.de/webaida/`
[17] `https://github.com/dbpedia-spotlight/dbpedia-spotlight/wiki`

Table 9. *Combined evaluation of NEE and NED*

(a) Brian collection

	Pre.	Rec.	F1
DBpedia Spotlight	0.1004	0.2669	0.1459
Stanford + AIDA	0.5005	0.2940	0.3704
TwitterNEED	**0.5455**	**0.5640**	**0.5546**

(b) Mena collection

	Pre.	Rec.	F1
DBpedia Spotlight	0.3711	0.5333	0.4377
Stanford + AIDA	**0.7263**	0.5569	0.6304
TwitterNEED	0.6861	**0.7157**	**0.7006**

(c) #Microposts collection

	Pre.	Rec.	F1
DBpedia Spotlight	0.1873	0.3349	0.2403
Stanford + AIDA	0.5092	0.2795	0.3609
TwitterNEED	**0.5337**	**0.5343**	**0.5339**

uation system although it uses Stanford_NER for automatic NE extraction. We consider the combination of Stanford_NER and the AIDA disambiguation system as one competitor to our extraction and disambiguation system. DBpedia Spotlight (Mendes *et al.* 2011) is a tool for automatically annotating mentions of DBpedia resources in text. We used DBpedia Spotlight through its Annotate Web Service endpoint. We used the NESpotter implementation for the extraction configuration.

For an evaluation of combined extraction and disambiguation, we consider the true positives set to include each correct exact mention extraction that is correctly assigned to its entity home page. The false positives set includes: (a) mentions that are partially extracted, (b) extracted mentions that are not part of correct NE, and (c) extracted mentions that match exactly a correct NE but are not successfully assigned to its entity home page. Finally, the false negatives set includes all NEs that are completely missed by the extractor. Evaluation is done only on the collections which have a ground truth for the entity home pages.

The results in Table 9 show the superiority of TwitterNEED over DBpedia Spotlight and the combined Stanford and AIDA system. TwitterNEED recall would be higher if we considered the top-k disambiguated home pages instead of the top one as we do here.

6 Conclusions and future work

In this paper, we present TwitterNEED, an approach for NEE and NED in tweets. We propose a hybrid approach for NEE in tweets that is based on both SVM and

CRF. The system is composed of three phases. The first phase aims to generate NE candidates with an emphasis on achieving high recall. The second phase aims to disambiguate all the candidates generated in the first phase. For this task, we propose a generic nonentity-oriented disambiguation approach. Mentions are disambiguated by assigning them to either a Wikipedia article or a home page. Instead of just selecting one entity for each mention, we produce a ranked list of possible entities. Finally, the third phase is to filter the NE candidates using features derived from disambiguation and other shape and KB features. This improves precision without significant harm to recall. Our combined extraction and disambiguation approach outperformed the state-of-the-art approaches such as DBpedia Spotlight and AIDA disambiguation system.

For future work, we would like to experiment with our approach for constructing KBs for closed domains from social media networks: as Twitter data is fresh, not everything immediately has a Wikipedia page or a home page retrievable with Google. For example, one could construct a KB for crisis management or for a local festival based on user-generated content from social media. Furthermore, we would like to include entity–entity similarity features to the disambiguation process. This will require an iterative process that repeats the disambiguation and extraction processes as suggested in Habib and van Keulen (2012a), because of the bad effect of the large number of false positives extractions on the disambiguation results in a first iteration.

References

Abeel, T., Van de Peer, Y., and Saeys, Y. 2009. Java-ml: a machine learning library. *Journal of Machine Learning Research* **10**: 931–4.

Basave, A. E. C., Varga, A., Rowe, M., Stankovic, M., and Dadzie, A.-S. 2013. Making sense of microposts (#msm2013) concept extraction challenge. In *Making Sense of Microposts (#MSM2013) Concept Extraction Challenge*, Rio de Janeiro, Brazil, pp. 1–15.

Bontcheva, K., Derczynski, L., Funk, A., Greenwood, M., Maynard, D., and Aswani, N. 2013. Twitie: An open-source information extraction pipeline for microblog text. In *Proceedings of the International Conference on Recent Advances in Natural Language Processing. Association for Computational Linguistics*, Hissar, Bulgaria, pp. 83–90.

Bunescu, R. C., and Pasca, M. 2006. Using encyclopedic knowledge for named entity disambiguation. In *EACL*, Trento, Italy, pp. 9–16.

Cano Basave, A. E., Rizzo, G., Varga, A., Rowe, M., Stankovic, M., and Dadzie, A.-S. 2014. Making sense of microposts (#microposts2014) named entity extraction & linking challenge. In *Proceedings of the 4th Workshop on Making Sense of Microposts (#Microposts2014)*, Seoul, South Korea, pp. 54–60.

Castillo, C., Mendoza, M., and Poblete, B. 2011. Information credibility on twitter. In *Proceedings of the 20th International Conference on World Wide Web*, Hyderabad, India. ACM, pp. 675–84.

Chang, C.-C. and Lin, C.-J. 2011. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* **2**(3–27): 1–27. Software available at `http://www.csie.ntu.edu.tw/ cjlin/libsvm`.

Christoforaki, M., Erunse, I., and Yu, C. 2011. Searching social updates for topic-centric entities. In *Proceedings of the 1st International Workshop on Searching and Integrating New Web Data Sources – Very Large Data Search (VLDS)*, Seattle, WA, USA, pp. 34–9.

Cucerzan, S. 2007. Large-scale named entity disambiguation based on Wikipedia data. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, Prague, Czech Republic, pp. 708–16.

Cunningham, H., Maynard, D., Bontcheva, K., and Tablan, V. 2002. GATE: a framework and graphical development environment for robust NLP tools and applications. In *Proceedings of the 40th Anniversary Meeting of the Association for Computational Linguistics (ACL'02)*, Philadelphia, Pennsylvania, USA, pp. 168–75.

Dann, S. 2010. Twitter content classification. *First Monday* **15**(12), `http://firstmonday.org/ojs/index.php/fm/article/viewArticle/2745/2681`.

Davis, A., Veloso, A., da Silva, A. S., Meira, Jr., W., and Laender, A. H. F. 2012. Named entity disambiguation in streaming data. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers – Volume 1*, ACL '12, Jeju Island, Korea, pp. 815–24.

Delgado, A. D., Mart'ınez, R., Pérez Garc'ıa-Plaza, A., and Fresno, V. 2012. Unsupervised Real-Time company name disambiguation in twitter. In *Workshop on Real-Time Analysis and Mining of Social Streams (RAMSS)*, Palo Alto, California, USA, pp. 25–8.

Derczynski, L. and Bontcheva, K. 2013. Mining social media with linked open data, entity recognition, and event extraction. In *Proceedings of the 3rd Workshop on Data Extraction and Object Search (DEOS 2013)*, Oxford, UK.

Dice, L. R. 1945. Measures of the amount of ecologic association between species. *Ecology* **26**(3): 297–302.

Finkel, J. R., Grenager, T., and Manning, C. 2005. Incorporating non-local information into information extraction systems by gibbs sampling. In *ACL*, University of Michigan, USA, pp. 363–70.

Gimpel, K., Schneider, N., O'Connor, B., Das, D., Mills, D., Eisenstein, J., Heilman, M., Yogatama, D., Flanigan, J., and Smith, N. A. 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: Short papers – Volume 2*, HLT '11, Portland, Oregon, USA, pp. 42–7.

Gupta, P., Goel, A., Lin, J., Sharma, A., Wang, D., and Zadeh, R. 2013. Wtf: the who to follow service at twitter. In *Proceedings of the 22nd International Conference on World Wide Web*, WWW '13, Rio de Janeiro, Brazil, pp. 505–14.

Habib, M. B. and van Keulen, M. 2012a. Improving toponym disambiguation by iteratively enhancing certainty of extraction. In *Proceedings of the 4th International Conference on Knowledge Discovery and Information Retrieval, KDIR 2012*, Barcelona, Spain. SciTePress, pp. 399–410.

Habib, M. B. and van Keulen, M. 2012b. Unsupervised improvement of named entity extraction in short informal context using disambiguation clues. In *Proc. of the Workshop on Semantic Web and Information Extraction (SWAIE 2012)*, Galway, Ireland, pp. 1–10.

Habib, M. B. and van Keulen, M. 2013. A hybrid approach for robust multilingual toponym extraction and disambiguation. In *IIS*, Warsaw, Poland, pp. 1–15.

Hoffart, J., Yosef, M. A., Bordino, I., Frstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., and Weikum, G. 2011. Robust disambiguation of named entities in text. In *Proceedings of EMNLP 2011*, Edinburgh, Scotland, UK, pp. 782–92.

Howard, P. and Hussain, M. 2013. *Democracy's Fourth Wave?: Digital Media and the Arab Spring*, Oxford Studies in Digital Politics. USA: OUP.

Jung, J. J. 2012. Online named entity recognition method for microtexts in social networking services: a case study of twitter. *Expert Systems with Applications* **39**(9): 8066–70.

Kulkarni, S., Singh, A., Ramakrishnan, G., and Chakrabarti, S. 2009. Collective annotation of wikipedia entities in web text. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '09*, Paris, France, pp. 457–66.

Li, C., Weng, J., He, Q., Yao, Y., Datta, A., Sun, A., and Lee, B.-S. 2012. Twiner: named entity recognition in targeted twitter stream. In *SIGIR*, Portland, Oregon, USA, pp. 721–30.

Li, L., Yu, Z., Zou, J., Su, L., Xian, Y., and Mao, C. 2009. Research on the method of entity homepage recognition. *Journal of Computational Information Systems (JCIS)* **5**(4): 1617–24.

Lin, T., Mausam, and Etzioni, O. 2012. Entity linking at web scale. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*, Montreal, Canada, pp. 84–8.

Locke, B., and Martin, J. 2009. *Named entity recognition: adapting to microblogging*. Senior Thesis, University of Colorado.

MacKay, D. J., and Peto, L. C. B. 1994. A hierarchical dirichlet language model. *Natural Language Engineering* **1**: 1–19.

Marsh, E., and Perzanowski, D. 1998. Muc-7 evaluation of ie technology: overview of results. In *Proceedings of the 7th Message Understanding Conference (MUC-7)*.

McCallum, A., and Li, W. 2003. Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons. In *Proceedings of CoNLL 2003*, Edmonton, Canada, pp. 188–91.

Mendes, P. N., Jakob, M., García-Silva, A., and Bizer, C. 2011. Dbpedia spotlight: Shedding light on the web of documents. In *Proceedings of the 7th International Conference on Semantic Systems, I-Semantics '11*, New York, NY, USA. ACM, pp. 1–8.

Ritter, A., Clark, S., Mausam, and Etzioni, O. 2011. Named entity recognition in tweets: An experimental study. In *Proceedings of EMNLP 2011*, Edinburgh, Scotland, UK, pp. 1524–34.

Rizzo, G. and Troncy, R. 2011. Nerd: Evaluating named entity recognition tools in the web of data. In *ISWC'11, Workshop on Web Scale Knowledge Extraction (WEKEX'11)*, Bonn, Germany.

Spina, D., Amigó, E., and Gonzalo, J. 2011. Filter keywords and majority class strategies for company name disambiguation in twitter. In *Proceedings of the 2nd International Conference on Multilingual and Multimodal Information Access Evaluation, CLEF'11*, Amsterdam, The Netherlands, pp. 50–61.

Srinivasan, H., Chen, J., and Srihari, R. 2009. Cross document person name disambiguation using entity profiles. In *Proceedings of the Text Analysis Conference (TAC) Workshop*, Gaithersburg, Maryland, USA.

Steiner, T., Verborgh, R., Gabarró Vallés, J., and Van de Walle, R. 2013. Adding meaning to social network microposts via multiple named entity disambiguation apis and tracking their data provenance. *International Journal of Computer Information Systems and Industrial Management* **5**: 69–78.

Suchanek, F. M., Kasneci, G., and Weikum, G. 2007. Yago: a core of semantic knowledge. In *Proc. of the 16th International Conference on World Wide Web, WWW '07*, Banff, Alberta, Canada, pp. 697–706.

Sullivan, S. J., Schneiders, A. G., Cheang, C.-W., Kitto, E., Lee, H., Redhead, J., Ward, S., Ahmed, O. H., and McCrory, P. R. 2012. what's happening? A content analysis of concussion-related traffic on twitter. *British Journal of Sports Medicine* **46**(4): 258–63.

Sutton, C. and McCallum, A. 2005. Piecewise training of undirected models. In *Proceedings of UAI*, Edinburgh, Scotland, UK, pp. 568–75.

Verma, M., Divya, and Sofat, S. 2014. Article: Techniques to detect spammers in twitter- a survey. *International Journal of Computer Applications* **85**(10): 27–32.

Wang, C., Chakrabarti, K., Cheng, T., and Chaudhuri, S. 2012. Targeted disambiguation of ad-hoc, homogeneous sets of named entities. In *Proceedings of the 21st International Conference on World Wide Web, WWW '12*, Lyon, France, pp. 719–28.

Wang, K., Thrasher, C., Viegas, E., Li, X., and Hsu, B.-J. P. 2010. An overview of microsoft web n-gram corpus and applications. In *Proceedings of the NAACL HLT 2010*, Los Angeles, California, USA, pp. 45–8.

Westerveld, T., Kraaij, W., and Hiemstra, D. 2002. Retrieving web pp. using content, links, urls and anchors. In *Proceedings of the 10th Text REtrieval Conference, TREC 2001*, vol. SP 500, Gaithersburg, Maryland, USA, pp. 663–72.

Winkels, M. 2013. The global social network landscape a country-by-country guide to social network usage. http://www.optimediaintelligence.es/noticias_archivos/719_20130715123913.pdf.

Wu, T.-F., Lin, C.-J., and Weng, R. C. 2004. Probability estimates for multi-class classification by pairwise coupling. *Journal of Machine Learning Research* **5**: 975–1005.

Yerva, S. R., Miklós, Z., and Aberer, K. 2012. Entity-based classification of twitter messages. *IJCSA*, **9**(1): 88–115.

Yosef, M., Hoffart, J., Bordino, I., Spaniol, M., and Weikum, G. 2011. Aida: An online tool for accurate disambiguation of named entities in text and tables. *Proc. of the VLDB Endowment* **4**(12): 1450–53.

Zhai, C. and Lafferty, J. 2001. A study of smoothing methods for language models applied to ad hoc information retrieval. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '01*, New Orleans, Louisiana, USA, pp. 334–42.

Zhu, Z., Hiemstra, D., Apers, P. M. G., and Wombacher, A. 2012. Separate training for conditional random fields using co-occurrence rate factorization. Technical Report TR-CTIT-12-29, Centre for Telematics and Information Technology, University of Twente, Enschede.

Zhu, Z., Hiemstra, D., Apers, P. M. G., and Wombacher, A. 2013. Closed form maximum likelihood estimator of conditional random fields. Technical Report TR-CTIT-13-03, Centre for Telematics and Information Technology, University of Twente, Enschede.