Searching Social Updates for Topic-centric Entities

Maria Christoforaki
NYU Poly
Brooklyn, New York
christom@cis.poly.edu

Ivie Erunse
NYU Poly
Brooklyn, New York
ieruns01@students.poly.edu

Cong Yu
Google Research
New York, NY
congyu@google.com

ABSTRACT

With the growing popularity of social networking services, real time short messages, such as Facebook news feeds and Twitter tweets, are becoming increasingly important information sources. People use these services to search for and consume content about interesting topics and events. Given a keyword search for a certain topic, simply returning those messages often does not give a comprehensive summary of the topic, primarily due to the brevity and redundancy of the messages. To address this challenge, we propose a topic centric entity extraction system where interesting entities pertaining to a topic are mined and extracted from short messages returned as search results on the topic. Specifically, we leverage signals from three main aspects: *message content*, *social connections* (i.e., message sender's follower network), and *referenced Web pages* (i.e., URLs embedded within the messages), and propose: 1) page ranking algorithms for identifying relevant pages embedded within the messages; and 2) entity ranking algorithms for identifying relevant entities extracted from those URLs. Comprehensive experiments using real Twitter data show that our ranking algorithms are efficient and outperform baseline algorithms significantly in terms of extraction quality.

Keywords

Social Stream, Entity Extraction, Ranking

1. INTRODUCTION

Social networking sites such as Twitter [2] enable users to share their opinions and feelings on a variety of topics with their friends and followers in the form of public text streams. These so called *social updates* have become one of the most important information sources, from which many users learn what's going on among their friends and around the world.

A typical information exploration paradigm on those social networking sites is to search (using keywords) for information related to a certain topic of interests to the user (e.g., "Libya"), and browse through the messages returned. While such results are useful, going through them for many popular topics suffers from the well known information overload problem, i.e., too many messages are returned and multiple messages are often conveying the same information. A conventional approach is to rank those messages and return the most relevant ones first. However, unlike Web pages, those social text messages are very short (e.g., Twitter has a length limitation of 140 characters on individual posts), and current ranking mechanisms that are designed for Web pages, do not perform well on those short messages. Furthermore, simply reading through a few messages ranked at the top typically cannot provide a user with enough information about the topic. In this paper, we propose an alternative approach, where we present the users with a summary of the topic of interest in the form of a set of participating entities and relevant URLs.

Specifically, we address the following problem: *given a social text stream associated with a topic of interest (defined as a keyword query), efficiently identify a ranked list of URLs and entities associated with the given topic.* The following example provides an overview of the system that we have built to solve this problem.

Example *Consider a user looking for recent news about "Tiger Woods." She would submit the hashtag #tigerwoods to the system, which in turn contacts Twitter through its API and obtains a large list of recent tweets associated with the requested topic. The system then extracts URLs (e.g., http://bit.ly/lsHHWP) from those tweets, ranks them, and prunes away those that are considered to be not useful. The remaining URLs are sent to the entity extraction system*[1]*, which extracts entities from those URLs. Finally, the extracted entities are ranked and returned to the user as the summary result. For #tigerwoods, this will contain "Elin Nordegren" and "Rachel Uchitel" as people, "Sweden" and "Orlando" as locations, etc.*

As this example shows, we adopt Twitter [2], one of the most popular social text stream services, as our primary data source of social updates. There are three main reasons why Twitter was chosen. First, as of September 2010, the service claims 200 million registered users, and 65 million messages (known as tweets) written per day, making it a huge and "fresh" knowledge base. Second, hashtags [10], which provide "ready" topic labels for the messages, are commonly adopted in tweets, and make it much easier for us to

[1]We use existing extraction systems for the actual entity extraction.

locate tweets for a particular subject. Finally, the social network (i.e., follower, followee, etc.) on Twitter can be exploited to improve the information extraction process with respect to the result quality.

Since tweets are restricted to 140 characters, users have come up with creative ways to maximize information sharing and one common practice is to embed shortened URLs in tweets (as the example illustrates). This presents us with two competing alternatives as the source for our entity extraction: the content of the tweets themselves or the content of the URLs mentioned within the tweets. We analyze both approaches and show that URL-based entity extraction provides significantly higher coverage for a variety of topics than tweet-based entity extraction. However, performing entity extraction from URLs within tweets presents two significant challenges. First, not all URLs in tweets are applicable to a topic and therefore we must detect the relevant ones. Second, fetching the page content of a URL and performing extraction over the content is an expensive process and therefore we must limit the number of URLs to be processed. Similar challenges are also evident when entities are extracted: they must also be filtered and ranked before they are displayed to the user.

Main contributions of the paper include: First, we introduce the novel social updates exploration paradigm of presenting users with a set of entities and URLs corresponding to a given topic. Second, we design and implement a machine learning based URL ranking component to identify embedded URLs that are relevant to the topic from the social updates. Third, we design and implement an entity ranking component that leverages information extraction and machine learning techniques to extract and rank entities that are relevant to the topic from those URLs. Finally, we provide a comprehensive evaluation of our approach using a range of topics over real Twitter data sets.

The remainder of this paper is organized as follows. Section 2 describes recent studies that are related to our work. Section 3 describes the machine learning based URL Ranker that aims to identify relevant URLs from the tweets of a given topic. Section 4 describes our Entity Ranker, which aims to identify relevant entities among those extracted from the relevant URLs. Both components use machine learning techniques and we will focus on describing the set of features we employed. In Section 5, we describe the set of comprehensive experiments conducted to evaluate the quality of the results produced by our system. Finally, we conclude in Section 6.

2. RELATED WORK

While our work is, to the best of our knowledge, the first to study the problem of searching for a comprehensive list of entities from social updates for a given topic, there are a few recent studies focusing on ranking Twitter content. For example, in [9, 12], the authors study useful features to rank the links embedded in tweets with the goal of helping Web search engines crawl more real time pages. Others, such as [6], propose URL recommender algorithms that use topic relevance and social voting to help users identify URLs that are of interests to them (in a topic agnostic way). Finally, [15] introduces a ranking mechanism to identify influential micro-bloggers. Our work builds upon similar features used in those ranking studies.

Entity extraction and ranking is another area that is closely related to our work. For example, [14, 11] describes an approach for selecting top-k named entities by creating a probability score which incorporates document importance and extraction confidence. While extraction confidence is an important signal, entity ranking in our work also needs to consider relevant and timeliness to the topic and there poses more challenges. In our experimental study, we compare our approach with a base approach that employs the extraction confidence directly. Entity ranking has also been studied in the domain of Wikipedia [8]. However, unlike Wikipedia, Twitter streams pose unique challenges for named entity recognition and classification [7, 5]. In our work, we extend many existing ranking features from those studies, as well as define Twitter specific entity ranking features.

3. URL RANKER

Given our goal of extracting topic-centric entities, one of the first questions is where the entities should be extracted from. There are two main approaches. The first approach is to directly apply the entity extraction on the textual content of the social updates (i.e., tweets), a strategy we call $Source_{TS}$. The second approach is to use the URLs embedded in those tweets and perform entity extraction on the page content of those URLs. The URL-based approach is more costly since it involves the additional step of fetching the page content. However, as our experimental analyses show in Section 5.1, the URL-based approach achieves substantially higher recalls with reasonable performance overhead. Therefore, we make the decision to leverage the URLs for entity extraction instead of simply relying on the tweets content alone.

Given the facts that there are often a large number of URLs embedded in the tweets (especially for popular topics), performing entity extraction on all those URLs is impractical. Our focus in this section is therefore on identifying good URLs from which entities can be extracted. We note that while URL recommendation based on user's interests has been studied previously [6], our work is the first attempt to rank URLs within tweets based on their suitability for topic-centric entity extraction.

We propose four main strategies for ranking informative URLs as outlined below. For each of those strategie, we assume a topic in the form of a hashtag (e.g., #tigerwoods) is given by the user and used to fetch the recent tweets containing the hashtag, until up to N URLs have been found in the tweets. We then aim to identify k most informative URLs, from which entities will be extracted[2].

Fresh URLs ($Source_{FU}$): We fetch the latest k URLs available in the Twitter stream and use them as the sources for entity extraction. This corresponds to the intuition that more recent URLs are more important.

Popular URLs ($Source_{PU}$): We fetch the k URLs with most frequent appearances in the tweets. This corresponds to the wisdom of crowd intuition that URLs endorsed by more users are more important.

Authoritative URLs ($Source_{AU}$): We fetch the k URLs that appeared in the tweets of users with the largest number of followers. This corresponds to the intuition that more authoritative users provide more useful information.

Mixed Feature URLs ($Source_{MFU}$): Finally, we adopt

[2] N and k are empirically set to 100 and 15, respectively, in our experiments.

a machine learning based ranking strategy and train an SVM (Support Vector Machine) classifier to rank URLs based on various features (described below). The k URLs with the highest probability estimates [4] of being good are identified and their content fetched for entity extraction.

The first three strategies can be implemented in a straightforward way. In the rest of this section, we describe in details the features being used in the $Source_{MFU}$ strategy, some of which are novel features proposed for first time in this study.

Let $S = \{t_1, t_2, ..., t_k\}$ be the sequence of tweets that are retrieved using the topic hashtag, and let $L_{all} = \{l_1, l_2, ..., l_m\}$ be the set of URLs embedded in those tweets. For each URL $l \in L$, the set $T(l) = \{t \in S \mid t \text{ contains } l\}$ denotes all the tweets in S that contains l. Let u_t be the publisher of tweet t, the set $U(l) = \{u_t \mid t \in T(l)\}$ denotes the set of authors who posted at least one tweet containing l.

3.1 Tweet Features

We consider the following two tweet content based features for each URL l under consideration.

Subject Specificity, $SS(l)$, is a novel feature that is computed as the average number of hashtags (i.e., topics) within each $t \in T(l)$. Intuitively, the more specific an URL is to the given topic, the more likely it contains useful information about the topic.

Spam Likelihood, $SL(l)$, measures how likely the URL is a spam URL. Here we adopt a simple strategy based on the set of pre-defined spam phrases (e.g., "tweet to win"), and l is marked as spam if at least t tweets in $T(l)$ contain at least one of the spam phrases. (t is usually set to a small number such as 1 or 2.) Note that while more advanced spam detection approaches such as [13] are available, we choose not to use them because spam is not a serious problem for the topics we have chosen in our study. Spam detection for social updates is by itself an interesting topic and beyond the scope of this paper.

3.2 Link Features

Next, we look at some of the key properties of the candidate URL l itself.

Link Popularity, $LP(l)$, measures how popular l is within the full social stream S, and is computed as $LP(l) = \frac{|T(l)|}{|S|}$. Note that URL shorteners (e.g., bit.ly) are frequently employed to shorten the URLs so that they can fit within the tweet length limit. To confirm two URLs being the same, we sometimes need to ping the shortener service when they are shortened by different shortener services.

Multimedia Likelihood, $ML(l)$, measures how likely the URL points to a multimedia file such as a picture, video, etc. This feature is particularly interesting since the goal of URL ranker is to identify URLs that can help achieve the overall goal of extracting entities. As a result, URLs pointing to popular multimedia portals such as flickr and youtube or multimedia specific sites such as twitpic and yfrog, while interesting, are not informative in this setting. We identify a dictionary of such domains and use this dictionary to predict whether a URL is likely to point to a multimedia page or a content page.

3.3 Social Features

Tweet publishers' social characteristics play an important role in the quality of tweets. For this reason we choose to consider a comprehensive set of publisher social features for the candidate URL l.

Publisher Experience, $PE(l)$, measures the average tenure of the publishers. Specifically, let $PE(u)$ be the number of days the publisher u has been a Twitter user, $PE(l) = \frac{1}{|U(l)|} \sum_{u \in U(l)} (PE(u))$.

Publisher Statuses, $PS(l)$, which measures the average number of message each publisher in $U(l)$ has contributed to Twitter. Let $PS(u)$ be the number of messages the publisher u has published since joining Twitter, $PS(l) = \frac{1}{|U(l)|} \sum_{u \in U(l)} (PS(u))$.

Publisher Activity, $PA(l)$, measures how active the publishers are on Twitter. Let $PA(u) = \frac{PE(u)}{PS(u)}$ represent how active of a publisher is on Twitter, $PA(l) = \frac{1}{|U(l)|} \sum_{u \in U(l)} (PA(u))$.

Publisher Importance, $PI(l)$, measures how important the publishers are within the Twitter communities. The importance is approximated with the number of followers a publisher has. Let $PI(u)$ represent the number of Twitter followers the publisher u has, $PI(l) = \frac{1}{|U(l)|} \sum_{u \in U(l)} (PI(u))$.

Publisher Diversity, $PD(l)$, is the fraction of the number of distinct users over the total number of tweets in $T(l)$. Intuitively, it is a bad sign if all the tweets associated with the URL are published by a few publishers.

Publisher Entropy, $PE(l)$, is a novel and more advanced measure of publisher diversity. Let $|T^u(l)|$ be the number of tweets in $T(l)$ that are published by u, we can compute $PE(l) = -\sum_{u \in U(l)} \frac{|T^u(l)|}{|T(l)|} \log \frac{|T^u(l)|}{|T(l)|}$.

Publisher Sociability, $PS(l)$, measures the social connectivity of the publishers. We define the friends of publisher u, $F(u)$, as the number of users who follows u and who u also follows. Hence, $PS(l) = \frac{1}{|U(l)|} \sum_{u \in U(l)} (F(u))$.

Follower to Followee Ratio, $FFR(l)$, approximates the "celebritiness" of the publisher and is computed as $FFR(l) = \frac{1}{|U(l)|} \sum_{u \in U(l)} \frac{\#users\ following\ u}{\#users\ u\ follows}$.

4. ENTITY RANKER

The set of top URLs as identified by the URL Ranker are submitted to AlchemyAPI [1], which performs the named entity extraction from the page contents. Specifically, we obtain a collection $E = \{e_1, e_2, ..., e_n\}$ of n entities as the result, and focus on how to identify those that are most central to the topic given by the user. We again emphasize that, in this work, our focus is not on improving the extraction techniques themselves, but rather on using the features that are unique to social updates to rank candidate entities obtained through existing state-of-art extraction systems. Similar to the URL Ranker, we adopt a machine learning approach and train a Naive Bayes classifier to make predictions on whether an entity is *good* based on various features as described below.

Let S be the sequence of tweets as defined in Section 3 and $L_{top} = \{l_1, l_2, ..., l_m\}$ be the top URLs as ranked by the URL Ranker. Let $E = \{e_1, e_2, ..., e_n\}$ be the set of entities extracted from L_{top} and $L(e) \subseteq L_{top}$ ($e \in E$) be the set of URLs from which e is extracted and $T(e) = \{t \mid \exists l \in L(e) \text{ s.t. } l \text{ is embedded in } t\}$. Finally, let $f_{(e,l)}$ be the term frequency of entity e in the content of URL l, and $h_{(e,l)} = 1$ if e can be extracted from the title of l.

We focus on three large categories of features: *text* features, which measure how important the entity is to the pages where it is extracted from; *tweet* features, which re-

late to the tweets where the entity is indirectly (i.e., through the embedded URL) extracted from; *social* features, which relate to the publishers who posted those tweets. A number of those features are proposed for the first time in this study.

4.1 Text Features

We consider the following features for candidate entity e:

Term Frequency, $TF(e)$, is the average term frequency of an entity e among all the URLs and is computed as:

$$TF(e) = \log \frac{1}{|L(e)|} \sum_{l \in L(e)} f_{e,l},$$

where we take the logarithmic so that the feature is not dominated with extremely high frequencies.

Entity Distribution, $ED(e)$, is a novel feature that measures the distribution of entities on the pages they were extracted from, and is computed as:

$$ED(e) = \frac{\sum_{l \in L(e)} \sum_{c \in l} f_{e,c} \log f_{e,c}}{|L(e)|}$$

where the content of each URL is divided into 10 equal sized chunks, $c_1, c_2, ..., c_{10}$, and $f_{e,c}$ measures the frequency of the entities within the chunk c. Intuitively, only entities that are central to the page will have a high distribution value over the page and only entities that are central to the topic will have high distribution value across all pages.

Inverted Document Frequency, $IDF(e)$, which is the equivalent of traditional term IDF and is computed as:

$$IDF(e) = \log \frac{|\{l \mid l \in L \wedge f_{e,l} > 0\}|}{|L|}$$

Is In Title Indicator, $IITi(e)$, which indicates the appearance of e inside a page title and is computed as:

$$IITi(e) = \begin{cases} 1 & : \exists l \in L(e), h_{(e,l)} > 0 \\ 0 & : otherwise \end{cases}$$

4.2 Tweet Features

This set of features relate to the tweets that the candidate entity e is extracted from:

Subject Specificity, $SS(e)$, is similar to $SS(l)$ and measures the average number of hashtags with each $t \in T(e)$.

URL Frequency, $UF(e)$, measures the average frequency the URLs containing e and is computed as $UF(e) = \frac{LP(l)}{|L(e)|}, l \in L(e)$, where $LP(l)$ is the Link Popularity as defined earlier.

Hashtag in Entity, $HIE(e)$, indicates whether the entity name overlaps with the topic hashtag itself. For example, the entity "National AIDS Commission" overlaps with the topic hashtag #aids, and therefore HIE("National AIDS Commission") = 1 for the topic #aids.

Is In Tweet, $IITw(e)$, indicates whether e is found in any tweet $t \in T(e)$ directly.

4.3 Social Features

Finally, we apply to the candidate entity e the similar set of social features that we apply to the URL as in Section 3.3. For completeness, we list the explicit definitions here. Note that $U(e)$ is the set of publishers who posted at least one tweet $t \in T(e)$, and $PE(u), PS(u), PA(u), PI(u)$, and $F(u)$ are defined in Section 3.3.

Publisher Experience, $PE(e) = \frac{1}{|U(e)|} \sum_{u \in U(e)} (PE(u))$.
Publisher Statuses, $PS(e) = \frac{1}{|U(e)|} \sum_{u \in U(e)} (PS(u))$.
Publisher Activity, $PA(e) = \frac{1}{|U(e)|} \sum_{u \in U(e)} (PA(u))$.
Publisher Importance, $PI(l) = \frac{1}{|U(e)|} \sum_{u \in U(e)} (PI(u))$.
Publisher Sociability, $PS(e = \frac{1}{|U(e)|} \sum_{u \in U(e)} (F(u))$.
Follower to Followee Ratio, $FFR(e) = \frac{1}{|U(e)|} \sum_{u \in U(e)} \frac{\#users\ following\ u}{\#users\ u\ follows}$.

5. EXPERIMENTAL EVALUATION

We implemented our system as an application over Twitter using Python/Java and conducted an extensive set of experiments to evaluate the quality of the topic centric entity search results. All experiments are run on a Windows XP machine with 1GB RAM, 1.6GHz Intel Core Duo CPU, and 20Mbps Internet connection.

Tweet Data Set: We chose 20 topics that were newsworthy over a week time period in 2010 (November 19-26, 2010) using Google Trends and identified corresponding most popular hashtags for each topic. Based on those hashtags, we fetched 380,000 tweets using the Twitter API. We discarded any tweet whose publishers are no longer with Twitter, which left us with 350,000 tweets. Among those, nearly 50% of the tweets (177,000) contain embedded URLs.

Training/Testing Data: For each of those 20 topics, we asked 10 volunteers to gather as many relevant entities as possible through any medium they considered as appropriate (newspapers, blogs, wikipedia, etc.). The result is a set of *golden entities* for each topic. Among the 20 original topics, 10 topics involve multiple entities, and they are selected for further study. For each of those 10 multi-entity topics, 100 most recently embedded URLs were extracted and rated by the volunteers based on the URLs' relevances to the topic on a scale of 0 to 2, representing "not relevant", "somewhat relevant", "very relevant", respectively. This URL rating data is used to evaluate our URL Ranker. Similarly, we performed entity extraction using the Alchemy API [1] for all 100 URLs of each of the 10 topics. On average, this produced 200 entities extracted per topic. Those extracted entities were rated by the volunteers using a similar relevance scale as in the URL rating data. This Entity rating data is used to evaluate our Entity Ranker.

Examples: Table 1 illustrates two example topics with top URLs and entities rated by the users. The first one is the insider trading investigation involving various former employees of SAC Capital, which is founded by Steve Cohen. The second one is on the decision by Pope Benedict XVI to allow the use of condom for AIDS prevention. Our system is able to successfully identify most of the URLs and entities that are considered relevant.

Methodology: We perform 10-fold cross validation for the quality experiments. In each run, both rankers (URL and Entity) are trained on the rating data of 9 topics and tested on the remaining topic. The average number across all 10 runs are reported. We use nDCG@K (*normalized discounted cumulative gain*) as the measure to evaluate the quality of the various algorithms.

5.1 Benefits & Overhead of URL-based Extraction

Topic	URL	location	organization	person
#insidertrading	http://goo.gl/wFxuz	Mountain View	Primary Global	Raj Rajaratnam
	http://on.wsj.com/cPdOg9	Boston	Loch Capital	Steven Fortuna
	http://viigo.im/5rcV	Wall Street	SAC Capital Advisors	Steve Cohen
#aids	http://huff.to/gWAPIC	Geneve	CDC	Howard Jaffe
	http://bbc.in/9mTIeA	Vatican	UN AIDS Agency	Pope Benedict
	http://bit.ly/bd8ieh	Rome	Catholic Church	Peter Seewald

Table 1: Two example topics (*insider trading* and *AIDS*) with top ranked URLs and entities of each type: location, organization, and person, over the time period November 19-26, 2010.

Extraction Source	topic coverage	% increase
$Source_{TS}$	0.39	n/a
$Source_{FU}$	0.46	17.95
$Source_{AU}$	0.47	20.51
$Source_{PU}$	0.53	35.90
$Source_{MFU}$	0.59	51.28

Table 2: Extracting from URLs improves coverage significantly.

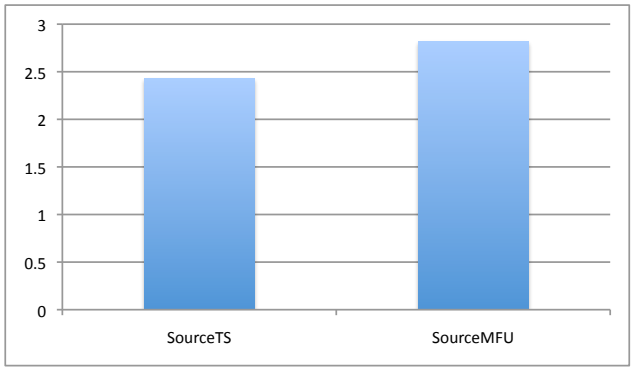

Figure 1: Comparing average overall latencies (seconds) between extracting from tweet content ($Source_{TS}$) and from URL ($Source_{MFU}$).

The first question we ask is whether extracting from the page content of the URLs embedded in the tweets is worth the effort, i.e., can we actually get a better coverage of the entities extracting from the URLs than simply extracting from the tweets? The answer is a resounding yes. As Table 2 shows, the four strategies (Section 3) that extract entities from URLs all achieve a significantly higher coverage (i.e., recall) of the golden entities for the topics than the strategy ($Source_{TS}$) that purely extracts entities from the tweets, even though the latter strategy goes over much more tweets[3]. The strategy that employs the machine learning approach achieves the highest increase in coverage at over 50%. It is interesting to see that authoritative URLs ($Source_{AU}$) performs worse than popular URLs ($Source_{PU}$) as an entity source, probably because the most authoritative URLs may be important but have highly overlapping content.

Furthermore, the much improved coverage comes at a reasonable performance overhead. As shown in Figure 1, extraction over the URL content incurs only about 16% overhead into the overall entity extraction latency (i.e., from a query topic being issued to the set of entities being returned). There are two main reasons for this surprisingly low overhead. First, content fetching can be done in parallel, which means the additional latency cost is only one page fetch even though 100 pages are retrieved. Second, the cost of actual entity extraction dominates the overall latency and therefore an additional page fetch is less significant in comparison[4].

[3]In this experiment, we do not rank the entities and use all entities extracted from the top-15 URLs.

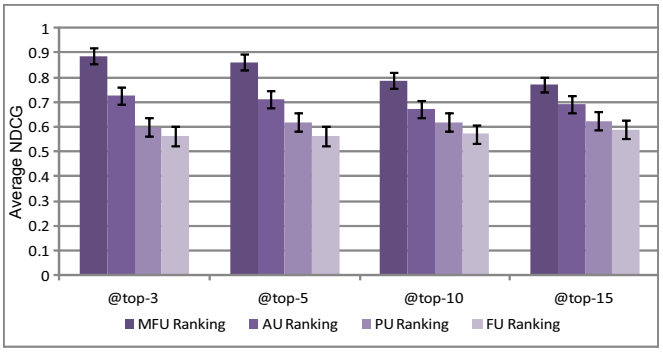

Figure 2: Average nDCG with error bars for URL rankings across different k values.

5.2 URL Ranker Evaluation

In this experiment, we compare the machine learned URL ranking algorithm (MFU) against all three baseline algorithms, Fresh URL (FU), Popular URL (PU), and Authoritative URL (AU) in terms of the quality of the URL rankings using nDCG. For MFU, the SVM classifier is trained using libsvm [4]. After training, it makes predictions on which class ("not relevant", "somewhat relevant", "very relevant") each of the test URLs belongs to. The URLs are then ranked according to their combined probability estimates of the latter two classes, and top k URLs are chosen, where k ranges from 3 to 15 in our experiments. For the three baseline algorithms, the URLs are ranked in the intuitive order, i.e., latest (FU), most popular (PU), most authoritative (AU), respectively. As shown in Figure 2, MFU handily beats all three baseline algorithms across all k values. We further note that URL recency performs badly compared with the other three algorithms, and yet it is currently the default way of showing the tweets to the users!

5.3 Entity Ranker Evaluation

In this experiment, we compared the machine learned entity ranking algorithm against the two baseline algorithms:

[4]We believe this is the case for the majority of advanced extraction systems currently available.

random, which randomly chose k entities from the set of extracted entities, and *alchemy*, which uses the extraction confidence (provided by the Alchemy API) to rank the entities. We further provide detailed analysis of the importance of each category of features ($TEXT$, $TWEET$, $SOCIAL$, see Section 4) by using all possible combinations of the feature categories. All classifiers are naive Bayes and trained using Weka [3]. As mentioned earlier, for each run, entity rating data of 9 topics are used to train the classifiers, which then make predictions on entities for the remaining topic. The probability estimates for the "very relevant" class are used to rank the entities. The results are averaged over the 10 cross-validation runs.

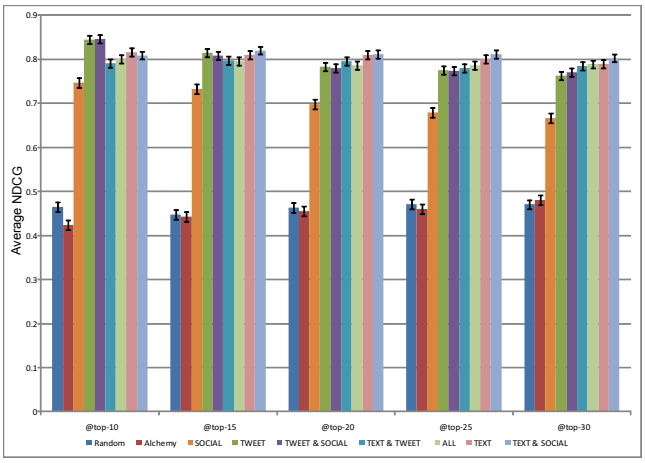

Figure 3: Average nDCG with error bars for entity rankings across different k values.

As shown in Figure 3, the machine learning based rankings performs significantly better than the two baseline algorithms in terms of nDCG. The extraction confidence based ranking performs identically with the random ranking, indicating that extraction confidence is a poor measure of how relevant the entity is to the given topic. Among the three categories of features, both $TEXT$ and $TWEET$ are sufficient to achieve good results. In fact, when $k = 10$, $TEXT$ features alone can achieve very good results. Contrary to what we believed prior to this study, ranking with $SOCIAL$ features alone performs noticeably worse than rankings with at least some $TEXT$ or $TWEET$ features, indicating that popular Twitter users are not necessarily the best sources for a comprehensive summary of a topic.

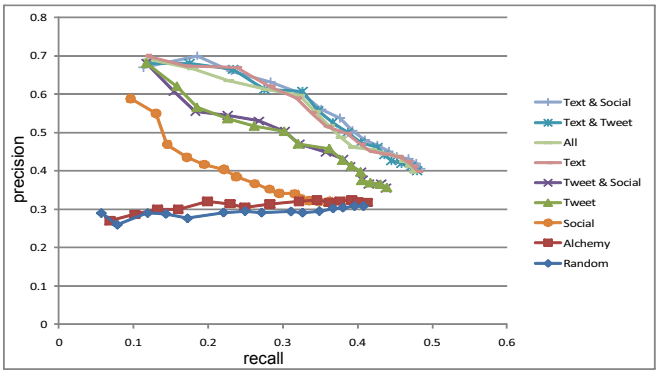

Figure 4: Precision-recall curve for entity rankings.

Finally, figure 4 illustrates the precision-recall curve over the same set of rankings, where the recall is computed as the percentage of entities each ranking recovered over the golden set of entities for each topic. Similar to the nDCG study, all machine learned rankings perform significantly better than the baseline rankings and $TEXT$ and $TWEET$ are the more important feature categories than the $SOCIAL$ features.

6. CONCLUSIONS AND REMARKS

In this study we design and implement a system to search for the most relevant named entities of a given topic from social updates. Our approach consists of two main components, URL Ranker and Entity Ranker. The former extracts and ranks top URLs from the sequence of social updates based on various features, while the latter leverages existing entity extraction to extract entities and machine learning techniques to rank the entities. An extensive set of experiments show that both our ranking algorithms significantly outperforms their respective baseline algorithms. We also study the various categories of features for entity ranking and show that text and tweet features are more important than social features for entity search.

7. REFERENCES

[1] Alchemy API: http://www.alchemyapi.com.

[2] Twitter: http://www.twitter.com/.

[3] Weka 3.0: http://www.cs.waikato.ac.nz/ml/weka/.

[4] C.-C. Chang and C.-J. Lin. *LIBSVM: a library for support vector machines*, 2001.

[5] B. Chen, P. Mitra, and Q. Zhao. Temporal and information flow based event detection from social text streams. In *AAAI*, 2007.

[6] J. Chen, R. Nairn, L. Nelson, M. S. Bernstein, and E. H. Chi. Short and tweet: experiments on recommending content from information streams. In *CHI*, 2010.

[7] W. J. Corvey, S. Vieweg, T. Rood, and M. Palmer. Twitter in mass emergency: what NLP techniques can contribute. In *NAACL HLT*, 2010.

[8] A. P. de Vries, A.-M. Vercoustre, J. A. Thom, N. Craswell, and M. Lalmas. Overview of the INEX 2007 entity ranking track. In *INEX*, 2007.

[9] A. Dong, R. Zhang, P. Kolari, J. Bai, F. Diaz, Y. Chang, Z. Zheng, and H. Zha. Time is of the essence: improving recency ranking using twitter data. In *WWW*, 2010.

[10] J. Huang, K. M. Thornton, and E. N. Efthimiadis. Conversational tagging in twitter. In *HT*, 2010.

[11] J. Huang and C. Yu. Prioritization of domain-specific web information extraction. In *AAAI*, 2010.

[12] V. Kandylas and A. Dasdan. The utility of tweeted urls for web search. In *WWW*, 2010.

[13] K. Lee, J. Caverlee, and S. Webb. Uncovering social spammers: social honeypots + machine learning. In *SIGIR*, 2010.

[14] M. Solomon, C. Yu, and L. Gravano. Popularity-guided top-k extraction of entity attributes. In *WebDB*, 2010.

[15] J. Weng, E.-P. Lim, J. Jiang, and Q. He. Twitterrank: finding topic-sensitive influential twitterers. In *WSDM*, 2010.