Extracting Information Networks from the Blogosphere: State-of-the-Art and Challenges

Filipe Mesquita
Department of Computing Science,
University of Alberta, Canada
mesquita@cs.ualberta.ca

Yuval Merhav
Computer Science Department,
Illinois Institute of Technology
yuval@ir.iit.edu

Denilson Barbosa
Department of Computing Science,
University of Alberta, Canada
denilson@cs.ualberta.ca

Abstract

We study the problem of automatically extracting relational networks of recognizable entities from the blogosphere. We describe a highly parallel and efficient system capable of processing millions of blog posts in a few hours, and experiments based on state-of-the-art techniques for the extraction and identification of entities and relations. From these, we devise a tuned approach that achieves substantial precision and recall at the task at hand. These results are indicate that effective large-scale extraction of such networks from the blogosphere is possible.

1. Introduction

Social Networks are the underlying structures that simultaneously reflect and influence most of our activities, opinions, and beliefs. For several decades, Sociologists have developed and refined techniques for observing and understanding such networks, at increasingly larger scales. One environment which stands out as a rich source of social information is the so-called *blogsosphere*–the network of social media sites, in which individuals express and discuss opinions, facts, events, and ideas pertaining to their lives or society at large. With the dramatic increase in size and diversity of the blogosphere in recent years, the automatic extraction of information from the blogosphere promises a viable approach for gathering rich social networks.

Previous works have focused on analyzing the readily available network of participants (i.e., authors) in the blogosphere. Unlike them, this paper focuses on the extraction of the networks of facts, ideas, and opinions expressed and discussed collectively by blog authors. In other words, we are interested in the networks that are described *in* social media repositories, by their participants. Given its size and diversity, extracting reliable information from the blogosphere is a formidable challenge that has received increasingly more interest from academia and industry. Furthermore, as the blogosphere attracts more and more participants from all segments of society, the extraction of the information networks latent in it may provide a better understanding of our collective view of the society we live in and talk about.

We present a system for SOcial Network EXtraction (SONEX) that works by identifying entities (e.g., people, organizations, geo-political entities, etc.) and extracting relations among them from the blogosphere. SONEX assigns meaningful labels for each relation, enabling the construction of detailed social networks. SONEX is developed as part of RankMe, an entity search and summarization engine for the blogosphere, which is our current research focus.

SONEX works by extracting and clustering *entity pairs* from the posts in the blogosphere. An entity pair is defined as the union of all sentences containing a given pair of named entities (e.g., Barack Obama and Michelle Obama). The *context* of an entity pair consists of selected terms that appear *between* the entities in the entity pair. The clustering step groups together entities with similar context; once this is done, each cluster is analyzed and a common label is assigned by inspecting the contexts of the pairs within the cluster.

Contributions. Many challenges exist in building a system such as SONEX, ranging from handling informal text (e.g., slang) and duplicate content (blogs often copy content from mainstream news or other blogs), to performance issues related to text processing (e.g., named entity recognition). This paper describes: (1) a scalable and efficient implementation of SONEX, capable of handling 10 million blogs in one day using 10 commodity-level PCs; (2) an extensive evaluation of state-of-the-art techniques for solving individual sub-problems in SONEX based on the Spinn3r data set (Burton, Java, and Soboroff 2009), containing 25 million blog posts in English (44 million in total) collected from August 1st, 2008 to October 1st, 2008; (3) an indication of how to tune and combine such techniques to successfully achieve our goal; and (4) a novel and rigorous automatic evaluation approach that relies on public, curated online databases for the construction of a ground truth for the task at hand (as opposed to a manual evaluation).

2. Related Work

Social network extraction. Techniques for the extraction of social networks from text, in smaller scales, have been proposed before. Referral Web (Kautz, Selman, and Shah 1997) is a system that takes a person name as input and finds people related to this person on the Web by using an external search engine. Referral Web uses the number

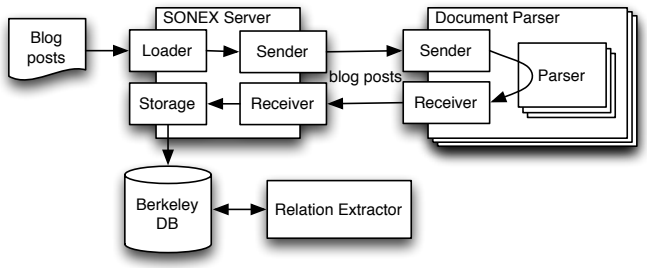
of pages where two person names co-occur to measure how much they are related. Unlike Referral Web, SONEX assigns a type for each edge by finding relations among entity pairs. Furthermore, our system uses sentences instead of web pages as the unit for co-occurrence resolution.

Targeted Relation Extraction. The seminal work on this area focused on a few predefined and domain-specific relations, and can be divided into two main approaches: bootstrapping and supervised learning. Bootstrapping methods use sample instances of the relations as input to successively extract more instances (Brin 1998; Agichtein and Gravano 2000). On the other hand, methods using supervised learning exploit either linguistic and statistical features defined *a priori* (Kambhatla 2004; GuoDong et al. 2005; Fisher et al. 1995; Craven et al. 2000; Rosario and Hearst 2004), or kernels methods (Zelenko, Aone, and Richardella 2003; Bunescu and Mooney 2005; Culotta and Sorensen 2004). Although effective, these methods do not scale to large and heterogeneous corpora that contain large number of unknown relations.

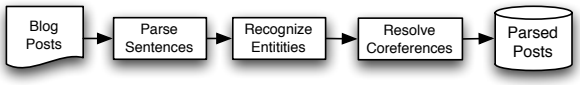
Open Relation Extraction. The large-scale extraction of relations without any relation-specific user input has been termed as Open Relation Extraction (ORE) (Banko et al. 2007). Since the relations are not known in advance, ORE also requires automatically assigning labels to each discovered relation. Recent systems addressing this problem are StatSnowBall (Zhu et al. 2009), TextRunner (Banko et al. 2007) and O-CRF (Banko and Etzioni 2008).

More related to our work are some systems that are based on clustering of entity pairs, based on their context, to produce relations (Hasegawa, Sekine, and Grishman 2004). This idea assumes that there is one dominant relation for each entity pair, and this relation can be identified by their context. Zhang et al. use parse trees of the context to allow pairs to appear in more than one cluster (Zhang et al. 2005). They assigned labels to clusters based on the most frequent "Head Word" in a cluster. To reduce noise, a common problem with text mining, known feature selection and ranking methods for clustering have been applied (Chen et al. 2005; Rosenfeld and Feldman 2007). Both works used the K-Means clustering algorithm with the stability-based criterion to automatically estimate the number of clusters. Rosenfeld and Feldman report that the Hierarchical Agglomerative Clustering (HAC) with single linkage outperforms both K-Means and the other variants of HAC (Rosenfeld and Feldman 2007). Finally, Shinyama and Sekine first cluster news articles into similar topics, identify patterns between named entities within a cluster, and then cluster the patterns to group similar entity pairs together (Shinyama and Sekine 2006). SONEX extends these works by clustering entity pairs extracted from the blogosphere. As far as we know, this is the first work to address the problem of relation extraction in this environment.

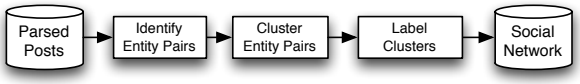
Accuracy Evaluation. The current practice for evaluating accuracy of ORE resorts to using gold standard relations from the Automatic Content Extraction (ACE)[1] or by clustering a small number of entity pairs manually. The ACE benchmark is a private corpus composed by news articles; therefore, it is not suitable for evaluating relations extracted from the blogosphere. On the other hand, it is hard to conduct an unbiased evaluation by choosing and clustering pairs manually. One of our contributions is an automatic method for evaluating both the clustering and labeling process.



(a) SONEX architecture.



(b) Document workflow.



(c) Entity pair workflow.

Figure 1: SONEX architecture and main workflows.

3. The SONEX System

We now detail the SONEX system, its document processing workflows, and some of our design decisions that allowed for the processing of the Spinnr3 dataset in reasonable time.

Pre-processing. The first step in our approach is to convert the blog posts written in English from the Spinn3r data set into plain text for easier manipulation. We start by replacing HTML tags and entities referring to special characters and punctuation marks (e.g. `&quot` or `<BR>`) with their corresponding plain text alternatives; this is accomplished with the Apache Commons library[2]. Next, we convert Unicode characters in blog posts to ASCII, using the LVG component of the SPECIALIST library[3].

Document workflow. Figure 1(b) illustrates the steps to process a document in SONEX. First, we extract sentences from a blog post using LingPipe[4], which employs sophisticated heuristics to accurately identify sentence boundaries.

Next, we identify named entities in the blog posts and assign a type for each named entity. For this, we use the LBJ

[1] http://projects.ldc.upenn.edu/ace/

[2] http://commons.apache.org/lang/

[3] http://lexsrv3.nlm.nih.gov/SPECIALIST/

[4] http://alias-i.com/lingpipe

Tagger[5], a state-of-the-art named entity recognition (NER) system. LBJ relies on the so-called BILOU scheme: the classifiers are trained to recognize the Beginning, the Inside, the Outside and the Last tokens of multi-token entities as well as single token (Unit-length) entities. It has been shown that this approach outperforms the more widely used BIO scheme (Ratinov and Roth 2009), which recognizes the Beginning, the Inside and the Outside of an entity name only. In addition, LBJ is able to identify four types of entity: person, organization, location and miscelaneous.

The final step is to identify in the text different references to the same real-world entity. This is accomplished using a *coreference* resolution tool to group these names together. In this work, we used Orthomatcher from the GATE framework[6], which has been shown experimentally to yield very high precision (96%) and recall (93%) on news stories (Bontcheva et al. 2002).

Entity pair workflow. Once the processed documents are stored in our database, we perform a cleaning on the corpus. First, we remove duplicate sentences, since preliminary experiments showed that they decrease the performance of the system. We found that 20% (around 10 million) of the sentences that contain entity pairs were duplicates. This is probably due to blogs that copy content from news articles or other blogs. In order to clean the NER mistakes on assigning types to entities, we replace the types of all entities with the same name but different type by their most frequent type. For instance, all entities with name "Barack Obama" was assigned to the type person.

Figure 1(c) shows the steps to produce the relations among entity pairs. In the first step, we identify entity pairs by searching for pair of entities within five words between each other in a sentence. We eliminate entity pairs whose context has more than 50 characters and less than 3, since they often are not useful. As usual, we set a threshold to discard entity pairs that appear in too few sentences since their context is often not useful for clustering. Although previous work set this threshold to 30 (Hasegawa, Sekine, and Grishman 2004), we decided to set it to 10 in order to evaluate the impact of unpopular entities in the results, as discussed later.

The next step is to find which entity pairs should be in the same relation. For this, SONEX clusters entity pairs whose context are similar. We refer to the contexts of all entity pairs in a cluster as the *cluster context*. Our system also assigns a descriptive name for each cluster by choosing one of the words appearing in the cluster context. The clustering and labeling steps are described in Section 4..

Our final outcome is the social network of the entities extracted from the blogosphere. Producing this network from entity pairs and clusters is straightforward. An entity is a node in the network and an entity pair is an edge, whose type is defined by the label of the entity pair's cluster.

[5]http://l2r.cs.uiuc.edu/ cogcomp/software.php

[6]http://gate.ac.uk/

Architecture

SONEX comprises three modules (Figure 1(a)): Server, Document Parser and Relation Extractor. Each module is implemented as a separate multi-threaded process, for increased parallelism. The Server consists of four main components: Loader, Sender, Receiver and Storage, which are responsible for reading documents, sending them to the Document Parser module, receiving the parsed posts back and storing all processed data in a local database. For the latter, we use Berkeley DB. The Server dedicates a Sender and a Receiver thread to each independent Document Parser connection.

The Server and the Document Parser modules work in a pipeline fashion. Also, all communication between the modules is buffered in local queues, to avoid synchronization as much as possible. As a result, we are able to adjust to number of instances of each module that can run in parallel, so as to maximize throughput. It is worth mentioning that our de-centralized architecture allows the processing of blog posts in 9ms on average, using a total of 10 commodity-level individual machines. This is a drastic improvement over the purely sequential approach in which blog posts are processed in 200ms on average.

The last component of SONEX is the Relation Extractor module, which runs after all blog posts are processed and all entities are identified. We discuss this process next.

4. Relation Extraction

The goal of the Relation Extraction module is to identify the relationship between pairs of entities, based on the sentences that relate these entities. In general, this comprises three distinct steps (Hasegawa, Sekine, and Grishman 2004): identifying entity pairs, clustering such pairs, and labelling the resulting clusters (see Figure 1(c)). Below we describe the challenges in each of these steps, and the various strategies we tested while developing SONEX (the next section contains the details of the evaluation).

Identifying Entity Pairs. We defined that two named entities form a pair if they appear within the same sentence and are separated by at most 5 intervening words. Often, there are several sentences in which the same pair of entities occur. We use the context of a entity pair to produce a vector for this pair where the features corresponds to the words in the context. Therefore, every pair is represented by a vector in the Vector Space Model (Manning, Raghavan, and Schütze 2008).

Each word w in the context vector of every pair is weighted by the widely adopted $tf \cdot idf$ weighting scheme (Manning, Raghavan, and Schütze 2008). Here, tf is the normalized frequency of the word in the context, while $idf = log(\frac{|D|}{d:w \in d})$, where $|D|$ is the total number of entity pairs, and $d : w \in d$ is the number of entity pairs that contain the word w at least once. Observe that this is identical to the usual $tf \cdot idf$ weight; thus, we compute the similarity between context vectors by using the cosine similarity measure (Manning, Raghavan, and Schütze 2008).

Rank	PoS Pattern	Example
1	to+Verb	to acquire
2	Verb+Prep	acquired by
3	Noun+Prep	acquisition of
4	Verb	offered
5	Noun	deal

Figure 2: Ranked part of speech patterns used by SONEX.

Our ultimate goal is to cluster entity pairs that belong to the same relation. Regardless of the clustering algorithm in use, the feature space plays an essential role. SONEX currently extracts the following features:

- **Unigrams**: The basic feature space containing all stemmed (Sparck Jones and Willett 1997) single words in the context of a entity pair, excluding stop words.
- **Bigrams**: Many relations may be better described by more than one word. For this reason, we include word bigrams, that is, two words that appear in sequence (ignoring those formed by stop words only.)
- **Part of Speech Patterns (POS)**: Banko and Etzioni claim that many binary relations in English are expressed using a compact set of relation-independent linguistics patterns (Banko and Etzioni 2008). We assume that a context sentence contains one relation at most. Hence, using the Stanford POS Tagger (Toutanova et al. 2003), we extract one of the predefined part of speech patterns listed in Figure 2 from sentences. If a context sentence contains more than one pattern, only the highest ranked one is extracted. We ranked the patterns according their frequency on sentences as estimated by previous work (Banko and Etzioni 2008).

Clustering Entity Pairs. As customary, we use Hierarchical Agglomerative Clustering (HAC) to cluster the entity pairs (Hasegawa, Sekine, and Grishman 2004; Zhang et al. 2005; Rosenfeld and Feldman 2007). HAC is a good option for our task since it does not require the number of clusters in advance. To measure the similarity between two clusters, we experiment with single, complete, and average link approaches (Manning, Raghavan, and Schütze 2008). Single link considers only the similarity between the two closest entity pairs from distinct clusters, while complete link considers the furthest ones. The average link considers the average similarity between all entity pairs from distinct clusters.

Labelling Clusters. The last phase is to label every cluster with a descriptive name. Currently, SONEX uses the following methods:

- **Centroid**: The centroid of each cluster (arithmetic mean for each dimension over all the points in the cluster) is computed, and then the feature with the largest mean value is selected as the cluster label.
- **Standard Deviation (Sdev):** A disadvantage of the centroid method is that the mean can be too biased towards one pair. To mitigate this problem, we propose to penalize terms with large standard deviation among the cluster's pairs. In this method, the feature to be selected as the label is the one that maximizes the value of the mean divided by the its standard deviation among all the pairs within a cluster.

5. Experiments

In this section we present an extensive evaluation of the SONEX system, thus characterizing the state-of-the-art in entity and relation extraction applied to the blogosphere. We propose automatic methods for evaluating the clustering process and the labels assigned to clusters.

Evaluating relation extraction methods

One problem to evaluate our system is that there is no benchmark data sets for the task of extracting relations from the blogosphere. The existing relation extraction benchmarks, such as ACE (recall the Related Work), are extracted from news stories, in which the text is produced and revised by journalists. Thus, ACE is not representative of the challenges faced when dealing with social media. To avoid this limitation, many researchers *manually evaluate* their results by assigning a few entity pairs into their proper relations, and using them as the ground truth for their evaluation. Two problems exist with this approach: manual evaluation can be easily biased by choosing either entity pairs or relations that the system is most likely to identify, and it does not scale.

One of our contributions is an automatic method that relies on publicly-available and curated databases for the evaluation of the accuracy. For the results reported in this paper, we used Freebase[7], a collaborative online database maintained by an active community of users. At the time of writing, Freebase contained over 11 million interconnected topics, including entities. Entities are connected by properties, such as "spouse", "place of birth", etc. For instance, Microsoft is connected to Bill Gates through the property "founders". Our evaluation considers a Freebase property as a relation and entities connected through a property are entity pairs of a relation.

The objective of our evaluation method is to asses the similarity between the relations extracted by the system and the Freebase relations. A high similarity means that the system is as effective as a human-maintained database at identifying relations between entities.

Choosing Freebase relations. To identify which relations were described by the Spinn3r data set, we selected a sample of 3,000 pairs randomly chosen from three groups. The first group contains 1,000 of these pairs appear in more than 300 sentences, the second contains 1,000 pairs appear in more than 100 sentences but less than 300 and the remaining 1,000 appear in less than 100 sentences compose the third group. We analyzed these pairs to determine a number of relations that could be extracted from the Spinn3r data set. In this

[7] http://www.freebase.com

Relation name	Domain	NER Type	# Pairs
Capital	Country–City/Town	LOC–LOC	77
Olym. Ath. Affiliation	Olym. athlete–Country	PER–LOC	40
Written work	Author–Work written	PER–MISC	28
Headquarters	Company–City/Town	ORG–LOC	21
Acquired by	Company–Company	ORG–ORG	11
Films Produced	Film Producer–Film	PER–MISC	11
Products	Company– Product	ORG–MISC	4
Total			192

Figure 3: Freebase relations selected for the evaluation after analyzing 3,000 pairs from Spinn3r. Domain shows the kind of Freebase entity belongs to each relation, while NER types shows the corresponding NER types.

analysis, we found 7 relations that had a corresponding property in Freebase as shown in Figure 3. We did not look at the sentences at this point to avoid biased decisions.

Determining the evaluation pairs. One problem to compare the relations extracted by our system and the relations from Freebase is that they do not contain the same pairs. Many pairs in the Spinn3r data set do not exist in Freebase and *vice versa*. To overcome this problem, we evaluate the pairs that appear in both Spinn3r and Freebase only. We call them *intersection pairs*.

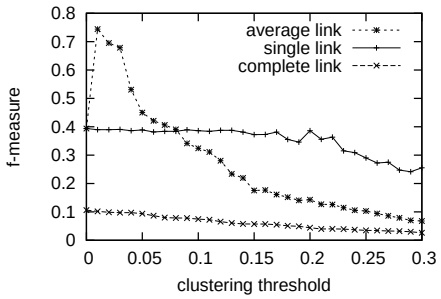
We define the intersection pairs as follows. First, we define the NER types of the entities that belong to each relation. Intersection pairs are those from Spinn3r and Freebase that match both entity names and types exactly. The number of pairs and the entity types for each relation are shown in Figure 3. "PER", "ORG" , "LOC" and "MISC" means person, organization, location and miscellaneous, respectively.

Data sets. We test our system on two data sets derived from Spinn3r: INTER and 10PERC. INTER contains the intersection pairs only, while 10PERC contains 10% of the pairs (approximately 13,000 pairs) in Spinn3r including the intersection pairs. Clustering only the intersection pairs reproduces the same experimental conditions as in a manual evaluation. In particular, INTER represents a sample of few hundreds of pairs whose context are likely to describe a single relation. Conversely, the 10PERC data set presents more realistic conditions, where the context of many pairs described several relations. Our goal is to compare the results from both data sets and provide hints on how to extract good relations even when many pairs describes more than one relation.

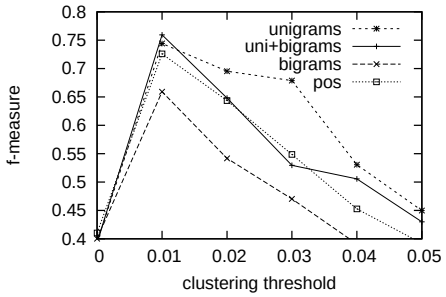
Metrics. Our goal for the following experiments is to measure the similarity between the relations extracted from our system and the relations from Freebase. For this we use precision (P), recall (R) and f-measure (M) as follows:

$$P = \frac{|S \cap F|}{|S|}, \;\; R = \frac{|S \cap F|}{|F|} \;\; and \;\; M = \frac{2PR}{P+R}$$

where S (resp. F) is the set of all *pairs of entity pairs* that belong to the same relation in our system (resp. in Freebase).



(a) Best clustering similarity measure.



(b) Best feature set.

Figure 4: Choosing the best clustering method on INTER.

More precisely, let S_p (resp. F_p) be the relation that contains the entity pair p in our system (resp. in Freebase); the sets S and F are defined as follows (Manning, Raghavan, and Schütze 2008):

$$S = \{\langle p, q\rangle | S_p = S_q\} \text{ and } F = \{\langle p, q\rangle | F_p = F_q\},$$

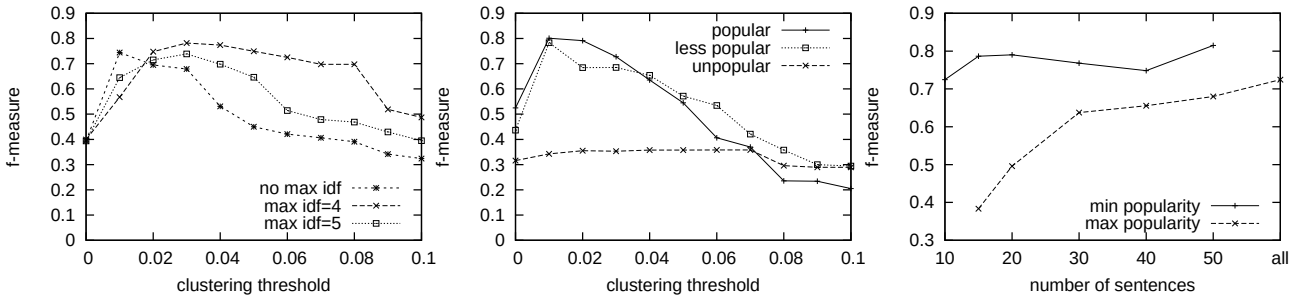
where $p \neq q$ (in both definitions above).

A high precision means that the entity pairs clustered together in our system are often in the same Freebase relation. Conversely, a high recall means that entity pairs in the same Freebase relation are often clustered together in our system. Finally, a high f-measure means high precision and recall.

Experiments with the INTER data set

Choosing the best clustering method. Our goal is to find the best clustering method for the INTER data set. For this, we test three cluster similarity measures (single, complete and average link) and four feature sets (unigrams, unigrams and bigrams, bigrams, and POS patterns).

Figure 4(a) presents the results for each cluster similarity measure using unigrams only. This experiment shows that the average link measure outperformed both single link and complete link. This is because single link produces few clusters containing many pairs. This behaviour results in high recall (0.89) but low precision (0.25) for the best clustering threshold (0.0). On the other hand, complete link produces many clusters containing few pairs, which results in low recall (0.05) but high precision (0.84) for the best clustering threshold (0.0). Average link demonstrates a better balance



(a) Using a maximum idf threshold. (b) Impact of popularity on cluster quality. (c) Pruning by popularity.

Figure 5: Applying pruning strategies on the INTER data set.

between precision (0.84) and recall (0.67) for the best clustering threshold (0.01). Therefore, we will use average link for the following experiments.

Figure 4(b) shows the average link performance for different feature sets. Observe that using unigrams only is comparable to using POS patterns or both unigrams and bigrams. Since unigrams is simpler, we will use it from now on.

Pruning. Now, we present some pruning strategies that our experiments demonstrate to be successful for the INTER data set. One strategy is to reduce the number of features. For this, we choose to filter words that appear in the context of too few entity pairs by setting a maximum idf value. Figure 5(a) shows that using a maximum idf of 4 produces higher precision (0.9) and recall (0.69) on the best clustering threshold (0.03). Furthermore, using a maximum idf threshold produces good results in a wider range of clustering thresholds than not using it. This behaviour is desirable for users trying to adjust the clustering threshold for a particular data set.

We also performed prunings on entity pairs appearing in a low number of sentences. First, we split the intersection pairs into three disjoint groups according their *popularity*, that is, the number of sentences in which they appear. Each group contains 64 entity pairs. The popular group contains pairs with 30 or more sentences, the less popular contains pairs with 16–29 sentences and the unpopular contains pairs with 16 or less (two pairs with 16 sentences were randomly allocated to the unpopular group to complete 64 pairs).

Figure 5(b) shows that clustering unpopular pairs produces bad results, while using popular and less popular pairs yields better results than clustering all pairs. One would then expect that removing all unpopular pairs would improve the results. However, this is not clearly true, as shown in Figure 5(c):the min and max popularity curves consider pairs with a minimum and maximum number of sentences (x-axis). The graphs show that pruning unpopular pairs *slightly* improves accuracy (min curve); however, removing popular pairs *substantially* degrades accuracy (max curve). One explanation for this is that increasing the number of popular pairs also increases the number of terms in the resulting clusters. In turn, this also increases the chances of unpopular pairs being clustered together: even if they do not share common terms, they might share terms with the popular pairs.

Experiments with the 10PERC data set

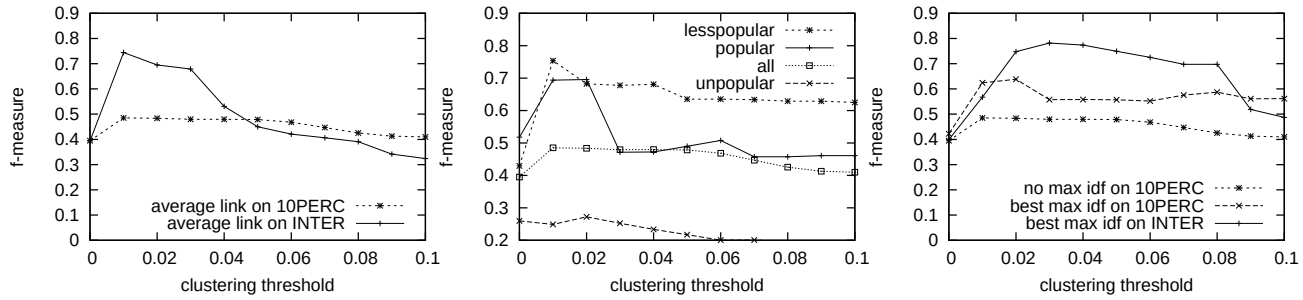
In this section we compare the results between the INTER and 10PERC data sets in order to highlight the challenges of extracting relations in more realistic conditions. In the following experiments we run the system over the 10PERC and then keep only the intersection pairs in the clusters.

We first compare the performance of average link on 10PERC and INTER. Figure 6(a) shows a substantial decrease of the results on 10PERC compared to the results on INTER. Our finds suggest that this decrease is caused by our system's inability to cluster unpopular pairs. Figure 6(b) shows that our system performs as good as in the INTER data set when clustering only popular and less popular pairs. However, we observe that the unpopular pairs greatly jeopardize the results. This observation is corroborated by the difference between the all pairs curve and the popular and less popular curves.

We also test how the maximum idf threshold affects the results for the 10PERC data set. Figure 6(c) shows that using max idf threshold of 5 improves the results reasonably, twice more than it improves in the INTER data set. However, the performance using maximum idf threshold on 10PERC is not yet comparable to the performance on INTER.

Evaluating cluster labels

Now, we evaluate the labels assigned by our system. Given a pair (e.g. J. K. Rowling–Harry Potter), we want to assess how related are the label produced by the system (e.g. book) to the Freebase relation (e.g. work written). For this, we assign to each Freebase relation a Wikipedia page that describes this relation as shown in Figure 7. For example, we assign the Wikipedia page "Authorship" to the relation "Works written". We measure the relatedness of a label to a relation by using the $tf \cdot idf$ weight as a metric of *importance* of this label in the relation's Wikipedia page. The best label that the system may produce is the one with the highest weight among the *candidate labels*. The candidate labels for a pair are the words in the context of its cluster. Hence, we generate a ranking of the candidate labels and report the ranking position of the label assigned for each pair.



(a) Average link performance. (b) Unpopular pairs jeopardize the results. (c) Pruning by maximum idf.

Figure 6: Experiments with the 10PERC data set.

Relation name	Wikipedia Page
Capital	Capital City
Olympic Athlete Affiliation	Olympic Games
Written work	Authorship
Headquarters	Headquarters
Acquired by	Merge&Acquisition
Films produced	Film producer
Products	Product (business)

Figure 7: Wikipedia pages for each Freebase relation.

	INTER		10PERC	
	Centroid	Sdev	Centroid	Sdev
Relation	Pos(Pairs)	Pos(Pairs)	Pos(Pairs)	Pos(Pairs)
Capital	1 (71)	2 (72)	1.6 (62)	1.7 (62)
Athlete Affiliation	3.9 (14)	5.5 (23)	25.1 (7)	19 (10)
Written work	4 (21)	4 (21)	1.1 (15)	2.9 (15)
Headquarters	1 (12)	2.4 (14)	1 (2)	N/A (0)
Acquired By	9 (7)	5.1 (9)	6.7 (3)	10.7 (3)
Films produced	N/A (0)	2 (10)	N/A (0)	N/A (0)
Product	1 (3)	N/A (0)	2 (1)	N/A (0)
Total	2.2 (128)	3.05 (149)	3.5 (90)	4.01 (90)

Figure 8: Experiment with labels.

We use the Yahoo! Search API[8] to compute the idf value for the words that appear in Wikipedia pages we selected. Since we are limited to a number of searches, we use only words that appear more than three times in a relation's Wikipedia page.

Figure 8 presents the average ranking position for the Centroid and Sdev label assigning methods on both INTER and 10PERC data sets. We also report the average ranking position (Pos) for each relation and the number of pairs (Pairs) whose label was found in the corresponding relation's Wikipedia page (the label is not evaluated if it does not exist in the page). Centroid performed better on both data sets, but Sdev was able to label more pairs with words in the Wikipedia pages. The label assigned by Centroid is in average among the 2.2 most important words in the ranking of candidate labels for the INTER data set and among the 3.5 most important words for the 10PERC.

[8]http://developer.yahoo.com/search/

	INTER		10PERC	
Relation name	Centroid	Sdev	Centroid	Sdev
Capital	capital	city	capital	capital
Athlete Affiliation	representing	representing	hit	right
Written work	book	book	author	book
Headquarters	headquarters	based	near	maybe
Acquired by	sold	acquired	company	reached
Films produced	documentary	movie	star	star
Products	sold	acquired	offers	words

Figure 9: Label examples.

Figure 9 shows the most frequent labels for each relation in our tests.

6. Conclusion

We presented a system that extracts information networks from the blogosphere by identifying named entities and relations among them. We discussed the architecture and workflows of our system, which allowed us to parse millions of blogs in a reasonable time. We proposed automatic methods for evaluating the clustering and labelling processes. Our experiments shed some light on the performance (and tuning) of state-the-art extraction tools in the context of information networks in the blogosphere. More specifically, the experiments showed that the relations produced by SONEX were similar to corresponding relations obtained from a curated database. In addition, the labels assigned by our system to each relation were among the most important words in a Wikipedia page describing those relations. We were able to produce this results by pruning rare words and unpopular pairs. In summary, our results are encouraging, given that blogs often use informal language (unlike the cleaner corpora for which the tools were originally optimized).

Our work on SONEX and RankMe are ongoing. We are producing an evaluation benchmark for the Spinn3r data set using every relation available in Freebase. This would help overcome the severe problem of lack of benchmarks for extracting relations from the blogosphere. We are also studying different ways to improve the results, including filtering blogs by measures of quality or structural rank in the blogosphere (e.g. in-degree) and filtering blog posts by time

window, since many relations are time sensitive (e.g. personal relationships). Another direction for future work is to investigate relation extraction methods driven by a given schema describing the relations of interest. These methods are especially useful for extracting relations from a specific domain, such as stock market or politics.

Acknowledgements

This work was supported in part by grants from the Natural Science and Engineering Research Council of Canada, and the Alberta Ingenuity Fund.

References

Agichtein, E., and Gravano, L. 2000. Snowball: extracting relations from large plain-text collections. In *Proceedings of the ACM Conference on Digital libraries*, 85–94. ACM.

Banko, M., and Etzioni, O. 2008. The tradeoffs between open and traditional relation extraction. In *Proceedings of the Annual Meeting of the ACL*, 28–36. Association for Computational Linguistics.

Banko, M.; Cafarella, M. J.; Soderland, S.; Broadhead, M.; and Etzioni, O. 2007. Open information extraction from the Web. In *Proceedings of the International Joint Conference on Artificial Intelligence*, 2670–2676.

Bontcheva, K.; Dimitrov, M.; Maynard, D.; Tablan, V.; and Cunningham, H. 2002. Shallow methods for named entity coreference resolution. In *Chaines de references et resolveurs d'anaphores, workshop TALN*.

Brin, S. 1998. Extracting patterns and relations from the world wide web. In *The World Wide Web and Databases, International Workshop*, 172–183.

Bunescu, R. C., and Mooney, R. J. 2005. A shortest path dependency kernel for relation extraction. In Mooney, R. J., ed., *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, 724–731. Association for Computational Linguistics.

Burton, K.; Java, A.; and Soboroff, I. 2009. The icwsm 2009 spinn3r dataset. In *Proceedings of the Annual Conference on Weblogs and Social Media*.

Chen, J.; Ji, D.; Tan, C. L.; and Niu, Z. 2005. Unsupervised feature selection for relation extraction. In *Proceedings of the International Joint Conference on Natural Language Processing*. Springer.

Craven, M.; DiPasquo, D.; Freitag, D.; McCallum, A.; Mitchell, T. M.; Nigam, K.; and Slattery, S. 2000. Learning to construct knowledge bases from the world wide web. *Artif. Intell.* 118(1-2):69–113.

Culotta, A., and Sorensen, J. S. 2004. Dependency tree kernels for relation extraction. In *Proceedings of the Annual Meeting of the ACL*, 423–429. Association for Computational Linguistics.

Fisher, D.; Soderland, S.; Feng, F.; and Lehnert, W. 1995. Description of the UMass system as used for MUC-6. In *Proceedings of the Conference on Message Understanding*, 127–140. Association for Computational Linguistics.

GuoDong, Z.; Jian, S.; Jie, Z.; and Min, Z. 2005. Exploring various knowledge in relation extraction. In *Proceedings of the Annual Meeting of the ACL*, 427–434. Association for Computational Linguistics.

Hasegawa, T.; Sekine, S.; and Grishman, R. 2004. Discovering relations among named entities from large corpora. In *Proceedings of the Annual Meeting of the ACL*, 415. Association for Computational Linguistics.

Kambhatla, N. 2004. Combining lexical, syntactic and semantic features with maximum entropy models. In *Proceedings of the Annual Meeting of the ACL*, 22. Association for Computational Linguistics.

Kautz, H.; Selman, B.; and Shah, M. 1997. Referral web: combining social networks and collaborative filtering. *Commun. ACM* 40(3):63–65.

Manning, C. D.; Raghavan, P.; and Schütze, H. 2008. *Introduction to Information Retrieval*. Cambridge University Press.

Ratinov, L., and Roth, D. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Conference on Computational Natural Language Learning*, 147–155. Association for Computational Linguistics.

Rosario, B., and Hearst, M. A. 2004. Classifying semantic relations in bioscience texts. In *Proceedings of the Annual Meeting of the ACL*, 430–437. Association for Computational Linguistics.

Rosenfeld, B., and Feldman, R. 2007. Clustering for unsupervised relation identification. In *Proceedings of the ACM Conference on Information and Knowledge Management*, 411–418. ACM.

Shinyama, Y., and Sekine, S. 2006. Preemptive information extraction using unrestricted relation discovery. In *Proceedings of Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, 304–311. Association for Computational Linguistics.

Sparck Jones, K., and Willett, P., eds. 1997. *Readings in information retrieval*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.

Toutanova, K.; Klein, D.; Manning, C. D.; and Singer, Y. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the Annual Meeting of the ACL*. Association for Computational Linguistics.

Zelenko, D.; Aone, C.; and Richardella, A. 2003. Kernel methods for relation extraction. *J. Mach. Learn. Res.* 3:1083–1106.

Zhang, M.; Su, J.; Wang, D.; Zhou, G.; and Tan, C. L. 2005. Discovering relations between named entities from a large raw corpus using tree similarity-based clustering. In *Proceedings of the international Joint Conference on Natural Language Processing*, 378–389. Springer.

Zhu, J.; Nie, Z.; Liu, X.; Zhang, B.; and Wen, J.-R. 2009. Statsnowball: a statistical approach to extracting entity relationships. In *Proceedings of the International Conference on World Wide Web*, 101–110. ACM.