Relational Duality: Unsupervised Extraction of Semantic Relations between Entities on the Web

Danushka Bollegala
The University of Tokyo
Hongo 7-3-1, Tokyo
113-8656, Japan
danushka@mi.ci.i.u-tokyo.ac.jp

Yutaka Matsuo
The University of Tokyo
Hongo 7-3-1, Tokyo
113-8656, Japan
matsuo@biz-model.t.u-tokyo.ac.jp

Mitsuru Ishizuka
The University of Tokyo
Hongo 7-3-1, Tokyo
113-8656, Japan
ishizuka@i.u-tokyo.ac.jp

ABSTRACT

Extracting semantic relations among entities is an important first step in various tasks in Web mining and natural language processing such as information extraction, relation detection, and social network mining. A relation can be expressed *extensionally* by stating all the instances of that relation or *intensionally* by defining all the paraphrases of that relation. For example, consider the ACQUISITION relation between two companies. An extensional definition of ACQUISITION contains all pairs of companies in which one company is acquired by another (e.g. (*YouTube*, *Google*) or (*Powerset*, *Microsoft*)). On the other hand we can intensionally define ACQUISITION as the relation described by lexical patterns such as ***X** is acquired by **Y***, or ***Y** purchased **X***, where **X** and **Y** denote two companies. We use this dual representation of semantic relations to propose a novel sequential co-clustering algorithm that can extract numerous relations efficiently from unlabeled data. We provide an efficient heuristic to find the parameters of the proposed co-clustering algorithm. Using the clusters produced by the algorithm, we train an L1 regularized logistic regression model to identify the representative patterns that describe the relation expressed by each cluster. We evaluate the proposed method in three different tasks: measuring relational similarity between entity pairs, open information extraction (Open IE), and classifying relations in a social network system. Experiments conducted using a benchmark dataset show that the proposed method improves existing relational similarity measures. Moreover, the proposed method significantly outperforms the current state-of-the-art Open IE systems in terms of both precision and recall. The proposed method correctly classifies 53 relation types in an online social network containing $470,671$ nodes and $35,652,475$ edges, thereby demonstrating its efficacy in real-world relation detection tasks.

Categories and Subject Descriptors

H.3.3 [**Information Systems**]: Information Search and Retrieval

General Terms

Algorithms

Keywords

Relation Extraction, Relational Similarity, Web Mining, Relational Duality

WWW 2010, April 26–30, 2010, Raleigh, North Carolina, USA.
ACM 978-1-60558-799-8/10/04.

1. INTRODUCTION

The World Wide Web contains numerous real-world entities connected by numerous semantic relations. Identifying the relations between entities is of paramount importance for numerous tasks on the Web such as information retrieval [29], information extraction [4], and social network extraction [23]. A semantic relation that exists between two given objects (e.g., concepts, words, or named-entities) can be defined in two ways [12]: *extensionally* or *intensionally*. An extensional definition of a concept formulates its meaning by specifying *every object* that falls under the definition of the concept. On the other hand, an intensional definition of a concept formulates its meaning by specifying *all the properties* that are necessary to reach that definition. For example, consider the ACQUISITION relation between two companies. An extensional definition of the ACQUISITION relation enumerates all pairs of entities between which an ACQUISITION relation holds (e.g. (*You Tube*,*Google*), (*Powerset*, *Microsoft*), etc.) Alternatively, we can express the ACQUISITION relation intensionally by stating the different ways that we can express an acquisition between two companies **X** and **Y** such as ***X** is acquired by **Y***, ***X** is purchased by **Y***, or ***X** is bought by **Y***. As described in this paper, we refer to this dyadic representations of semantic relations as *relational duality*, and use it in an unsupervised co-clustering algorithm to extract numerous semantic relations from a given corpus. In contrast to previously proposed supervised or semi-supervised approaches, which require some form of human intervention such as annotated training data, seeds of entity-pairs, or extraction patterns, the proposed method is fully unsupervised.

Extracting relations between entities has received much attention lately. In object-level search engines such as Renlifang[1] [38], it is particularly important to mine entity relations from the Web to build automatically, an entity relation graph to link all the extracted information together. In contrast to document-level search engines, for which a user enters a keyword and retrieves a set of documents, in an object-level search engine, users search for a particular entity or a relation between entities. In open information extraction (Open IE) systems such as TextRunner [4], the goal is to extract a large set of relational tuples without the need for any human input. The extracted relations can then be used to answer natural language questions. In the bio-medical domain, identifying the relations between proteins and diseases is helpful to discover potential side effects of various medicines.

Despite its numerous applications, extracting semantic relations among entities at Web scale is challenging for several reasons. First, a single semantic relation can be expressed using multiple lexical patterns. For example, aside from the pattern ***X** acquired **Y***, an ac-

[1] http://renlifang.msra.cn

quisition between two companies **X** and **Y** can be expressed using patterns such as ***X purchased Y***, ***X completed its acquisition of Y***, etc. Second, there might exist more than one semantic relation between a pair of entities. For example, before an ACQUISITION relation is established between two companies, those companies can have a COMPETITOR relation. A relation extraction system must discover the different relations that hold between a pair of entities. Third, the entities themselves might have variants. For example, Microsoft Corp. is often designated as *the Redmond software giant*. Manually specifying all different name variants of an entity is not feasible. Moreover, the scale and the heterogeneity of Web text prohibit the use of time-consuming, domain-specific approaches that require deep language processing techniques. Supervised approaches to relation extraction that require manual annotation of all relations to be extracted are costly and impossible to execute on a Web scale because we do not know in advance the number or the types of relations that we must extract from the Web. Semi-supervised approaches to relation extraction require some seed instances (i.e., a few pairs of entities between which the desired relation exist) or extraction patterns (either domain specific or independent) to be provided by a human. Unfortunately, the quality of the extracted relations depends heavily on the initial seeds given to the system. Moreover, it is not clear how many seeds are necessary to extract a particular relation correctly beforehand.

We propose an unsupervised approach that solves all the above-described problems in a principled manner. Given a text corpus, the proposed method first extracts all mentions of entities and lexical-syntactic patterns that connect those entities. We then represent pairs of entities and lexical-syntactic patterns in a matrix whose rows represent pairs of entities and columns represent lexical-syntactic patterns. An efficient sequential co-clustering algorithm is proposed to simultaneously identify different patterns that describe the same semantic relation, and entity-pairs between which the same semantic relation holds. Moreover, we introduce a computationally efficient heuristic to determine the row (entity-pair) and column (lexical-syntactic pattern) clustering thresholds in the algorithm. To identify the relations represented by each pattern cluster, we use the resultant clusters to train a self-supervised multi-class logistic regression model with L1 regularization [24]. This approach produces a sparse representation of relations over lexical-syntactic patterns, thereby enabling us to identify the representative patterns in a cluster.

The contributions of this paper can be summarized as follows.

- We propose a dyadic representation of semantic relations that exist between a pair of entities using extensional and intensional representations. Specifically, a semantic relation R is expressed extensionally by extracting the set of entity pairs $E(R)$, between which relation R holds. Alternatively, the same relation R can be expressed intensionally by specifying the set of lexical-syntactic patterns $P(R)$ used to express R. To identify the potential entities in a corpus, we use a part-of-speech tagger and a noun-phrase chunker. We employ a subsequence pattern-mining algorithm to extract lexical-syntactic patterns that express numerous semantic relations between a pair of entities.

- We propose a sequential co-clustering algorithm to simultaneously cluster different lexical-syntactic patterns that describe a particular semantic relation, and different entity pairs between which the same semantic relation holds. The proposed sequential co-clustering algorithm avoids combinatorial pairwise comparisons of data points and scales linearly with the number of data points to be clustered. This desirable property of the proposed co-clustering algorithm enables us to use it with large real-world datasets containing numerous semantic relations. Moreover, we propose an efficient heuristic to determine the optimum values of clustering thresholds.

- To identify representative patterns that describe the semantic relation expressed by a cluster, we train a multi-class logistic regression model with L1 regularization. The training is conducted in a self-supervised manner that requires no manually annotated training data, which is particularly important because we do not know the number or the type of relations that exist in a given corpus beforehand. Moreover, L1 regularization yields sparse representations, consequently enabling us to find the most representative patterns that describe a particular semantic relation.

- We evaluate the proposed relation extraction algorithm in three tasks: measuring relational similarity between entity-pairs, open information extraction, and classification of relations in a social network system. The proposed method outperforms previously proposed Open IE systems on a benchmark dataset. Furthermore, the produced pattern clusters improve existing relational similarity measures. Moreover, the proposed method is used to classify 53 different relations in a social network of $470,671$ nodes and $35,652,475$ edges, thereby demonstrating its efficacy in real-world large-scale applications.

The remainder of the paper is organized as follows. In Section 2.1, we introduce the concept of relational duality. We present a single-pass extraction algorithm in Section 2.2 to identify both entity pairs and lexical-syntactic patterns simultaneously in a given corpus. A sequential co-clustering algorithm is proposed in Section 2.3 to identify the numerous semantic relations described by extensional and intensional representations of relations extracted in Section 2.2. A self-supervised classifier is presented in Section 2.4 to identify the representative patterns that describe the relation captured by each cluster. We evaluate the proposed relation extraction method using three tasks in Section 3. We discuss related research efforts in Section 4 and conclude this paper.

2. RELATIONAL DUALITY

2.1 Outline

We propose *relational duality*: a dyadic representation of the semantic relations that exist between two entities. A semantic relation R is definable by stating all entity pairs between which relation R holds. We use the notation $E(R)$ to denote the set of entity pairs in which the relation R holds between the two entities in each element (i.e. entity pair). Defining a concept by enumerating all its instances is called an *extensional* definition of the concept. Alternatively, we can define R by stating the different properties that must be satisfied by two entities to realize a relation R between them. We denote the set of properties of R as $P(R)$. Defining a concept by stating all the properties that are required to come to that definition is called an *intensional* definition of the concept. Both $E(R)$ and $P(R)$ define the same semantic relation R. Therefore, a duality exists between the two definitions for R.

For example, we can extensionally define the ACQUISITION relation by specifying all pairs of companies between which an acquisition has taken place, such as (Google,YouTube), (Microsoft, Powerset), (Yahoo, Inktomi), etc. An intensional definition of ACQUISITION specifies the different expressions that indicate an acquisition has taken place between two companies **X** and **Y**, such as

X** acquires **Y**, **Y** is bought by **X**, **X** purchases **Y, etc. It is not possible for a human to enumerate all the different expressions that describe a particular semantic relation. Consequently, in Section 2.2, we propose a single-pass extraction method that extracts both entity pairs and numerous lexical-syntactic patterns that describe different relations that exist in a given corpus.

The dual representation of semantic relations can be captured efficiently within a co-clustering framework. We represent relational data in a matrix in which the rows correspond to entity pairs, and columns correspond to lexical-syntactic patterns. The row vectors can be considered as defining the distribution of a particular entity pair over the space spanned by lexical-syntactic patterns. Similarly, each column vector represents the distribution of patterns over the space spanned by the entity pairs. Distributional similarity [19, 21] is useful to identify the different patterns that describe the same semantic relation R (i.e.$P(R)$), and different entity pairs between which the same semantic relation R exists (i.e.$E(R)$). In Section 2.3, we propose a sequential co-clustering algorithm to cluster a large data matrix efficiently. By framing relational duality as a co-clustering problem, we can identify the variants of a particular entity. For example, if *Redmond software giant* is a variant of *Microsoft*, then we would expect both entity pairs (*Microsoft*,*Powerset*) and (*Redmond software giant*,*Powerset*) to appear within the same cluster. By merging the variants of an entity, we can reduce the sparsity in pattern vectors, thereby improving the accuracy of clustering.

Each cluster produced by the co-clustering algorithm expresses a particular semantic relation. However, in this paper, we do not assume any prior knowledge related to the number or the type of relations that exist in a given corpus. Therefore, it is extremely useful to identify the relation expressed by each cluster both for evaluation purposes, and for presentation purposes. In Section 2.4, we propose a self-supervised relation detection algorithm that labels each entity pair (row) cluster with representative lexical patterns.

2.2 Single-Pass Extraction

To extract entity pairs and lexical-syntactic patterns from a given text corpus, we propose a single-pass extraction method. Because the proposed extraction method requires only a single traversal over the corpus, visiting each sentence once, it is scalable to large datasets. We resort to shallow linguistic processing techniques such as sentence boundary detection, part-of-speech (POS) tagging, and noun phrase chunking, for which efficient and accurate tools are available for many languages.

First, we split the given text corpus into sentences using a sentence boundary detection tool[2]. We then run a part-of-speech (POS) tagger[3] and annotate each sentence with POS tags. To detect potential entities in sentences, we use a noun phrase chunking tool[4] and extract noun phrase chunks containing at least one proper noun (NP). It is noteworthy that we require no deep linguistic analysis, as do Open IE systems, which require dependency parsing [4] or coreference resolution [31]. Moreover, we do not assume the availability of named entity recognition (NER) tools that can tag entities of different types in a text. Instead, we use a name phrase chunking tool that can group multi-word entities such as *Adobe Systems* or *Microsoft Corporation*. No additional information, such as the type of the entity (i.e., person, company, or location) is required in the subsequent processing.

[2] `http://stp.ling.uu.se/~gustav/java/classes/MXTERMINATOR.html`

[3] `http://nlp.stanford.edu/software/tagger.shtml`

[4] `http://chasen.org/~taku/software/yamcha/`

An example is presented in Table 1, from which we extract the entity pair (*Adobe Systems*,*Macromedia*). Next, we replace the two entities respectively with two variables **X** and **Y** in a sentence. The entity that occurs first in the sentence is replaced by **X**, whereas the entity that occurs second is replaced by **Y**. The corresponding POS tags in the POS tag sequence is also replaced by **X** and **Y**, as presented in Table 1.

Lexical-syntactic patterns have been used successfully in various natural language processing tasks such as extracting hypernyms [20, 32], or meronyms [6], question answering [27], and paraphrase extraction [7]. Following those previous works, we present a shallow lexical pattern extraction algorithm to represent the semantic relations between two entities.

We generate subsequence patterns from both surface forms of the sentences and POS tag sequences that satisfy the following conditions.

(i). A subsequence must contain exactly one occurrence of each **X** and **Y** (i.e., exactly one **X** and one **Y** must exist in a subsequence).

(ii). The maximum length of a subsequence is L tokens.

(iii). A subsequence is allowed to have gaps. However, we do not allow gaps of more than g number of tokens. Moreover, the total length of all gaps in a subsequence should not exceed G tokens.

(iv). We expand all negation contractions in a sentence. For example, *didn't* is expanded to *did not*. We do not skip the word *not* when generating subsequences. For example, this condition ensures that from *X is not a Y*, we do not produce the subsequence *X is a Y*.

We designate the subsequences of surface forms produced by the procedure described above as *lexical* patterns. The corresponding POS tags of a lexical pattern is called a *syntactic* pattern. The values of parameters L, G, and g are set experimentally, as explained later in Section 3.

The proposed lexical-syntactic pattern extraction algorithm considers all the words in a sentence, and is *not* limited to extracting patterns only from the mid-fix (i.e., the portion of text in a sentence that appears between a pair of entities). Moreover, the consideration of gaps enables us to capture relations between entities located at a distance in a sentence. We use *prefixspan* algorithm [26] to generate subsequences. The constraints listed above are used to prune the search space, thereby reducing the number of subsequences generated by prefixspan. Some lexical-syntactic patterns extracted using the proposed method are shown in Table 1 (not all patterns are shown because of the limited availability of space).

We consider all entity pairs in a sentence, and extract lexical-syntactic patterns using the procedure described above. We then aggregate entity pairs and lexical-syntactic patterns we extract from each sentence. It is noteworthy that the proposed extraction method visits each sentence only once. Once we have completed processing the entire corpus, we select the most frequently occurring entity pairs and lexical-syntactic patterns for the co-clustering algorithm described in Section 2.3.

2.3 Sequential Co-clustering

Assuming that the set of all extracted entity pairs is E, and that the set of all extracted lexical-syntactic patterns is P, then we propose a sequential co-clustering algorithm to simultaneously identify the subset of lexical-syntactic patterns $P(R)$ that describes a particular semantic relation R, and the subset of entity pairs, $E(R)$, between which R holds. First, we represent the dyadic relation between entity pairs and lexical-syntactic patterns as matrix A. Each

Table 1: Extracting entity pairs and lexical-syntactic patterns from text.

sentence	another example of a statutory merger is software maker Adobe Systems acquisition of Macromedia .
POS tagged	DT NN IN DT JJ NN VBZ NN NN NNP NN NN IN NP .
Entity chunked	another example of a statutory merger is software maker [Adobe Systems] acquisition of [Macromedia].
Substitution	*Adobe Systems* = **X**, *Macromedia* = **Y**
surface form	another example of a statutory merger is software maker **X** acquisition of **Y** .
POS sequence	DT NN IN DT JJ NN VBZ NN NN **X** NN IN **Y** .
lexical patterns	**X** acquisition of **Y**, software maker **X** acquisition of **Y**, **X** of **Y**, software **X** acqusition **Y**
syntactic patterns	**X** NN IN **Y**, NN NN **X** NN IN **Y**, **X** IN **Y**, NN **X** NN **Y**

Algorithm 1 Sequential co-clustering algorithm.

Input: sets E, P, matrix A, thresholds θ, ϕ
Output: row clusters C_E, column clusters C_P

1: SORT(E)
2: SORT(P)
3: $C_E \leftarrow \{\}, C_P \leftarrow \{\}$
4: **while** $E \neq \{\}$ AND $P \neq \{\}$ **do**
5: $\quad \mathbf{p} \leftarrow \text{POP}(P)$
6: $\quad \text{ASSIGN}(\mathbf{p}, C_p, \theta)$
7: $\quad \mathbf{e} \leftarrow \text{POP}(E)$
8: $\quad \text{ASSIGN}(\mathbf{e}, C_E, \phi)$
9: **end while**
10: **return** C_E, C_P

11: **function** ASSIGN($\mathbf{x}, C, \lambda$)
12: $max \leftarrow -\infty$
13: $\mathbf{c}^* \leftarrow null$
14: **for** cluster $\mathbf{c}_j \in C$ **do**
15: $\quad sim \leftarrow \text{cosine}(\mathbf{x}, \mathbf{c_j})$
16: $\quad$ **if** $sim > max$ **then**
17: $\quad\quad max \leftarrow sim$
18: $\quad\quad \mathbf{c}^* \leftarrow \mathbf{c}_j$
19: $\quad$ **end if**
20: **end for**
21: **if** $max > \lambda$ **then**
22: $\quad \mathbf{c}^* \leftarrow \mathbf{c}^* \oplus \mathbf{x}$
23: **else**
24: $\quad C \leftarrow C \cup \{\mathbf{x}\}$
25: **end if**

extracted entity pair in Section 2.2 is represented as a row in this matrix, whereas each lexical-syntactic pattern is represented as a column. The A_{ij} element of the data matrix denotes the number of times the lexical-syntactic pattern p_j was extracted for the entity pair e_i. Each normalized row vector $\mathbf{e}_i$ in matrix A denotes the distribution of an entity pair e_i over lexical-syntactic patterns. Similarly, each normalized column vector $\mathbf{p_j}$ in matrix A denotes the distribution of a lexical-syntactic pattern p_j over entity pairs. From distributional hypothesis [19], it follows that if two entity pairs are distributed similarly over a set of lexical-syntactic patterns, then those entity pairs must be relationally similar. We use distributional similarity to cluster entity pairs and lexical-syntactic patterns simultaneously.

The pseudo-code of the proposed sequential co-clustering algorithm is presented in Algorithm 1. The algorithm takes as its input E, P, A, and two clustering thresholds: row (entity pair) clustering threshold, ϕ, and column (lexical-syntactic pattern) clustering threshold, θ. The output of the clustering algorithm is the set of row clusters, C_E, and column clusters, C_P. First, in Line 1, we sort the set of entity pairs E in the descending order of total frequency, $\sum_j A_{ij}$, of each entity pair e_i with all lexical-syntactic patterns in P. Similarly, in Line 2, we sort the set of lexical-syntactic patterns P in the descending order of total frequency, $\sum_i A_{ij}$, of each pattern with all entity pairs in E. After sorting, the most common entity pairs and patterns in the corpus appear respectively at the beginning of E and P, whereas rare instances are shifted to the end. In Line 3, we initialize both row and column cluster sets to the empty set. The function, POP(P) in Line 5, returns the first pattern $p \in P$ and removes p from P, thereby reducing the size of P by one. Next, the function, ASSIGN, measures the similarity between the vector $\mathbf{p}$ that corresponds to pattern p and each column cluster $\mathbf{c}_j$ in C_P. Here, $\mathbf{c}_j$ denotes the centroid vector of the j-th column cluster. Similarity between $\mathbf{p}$ and $\mathbf{c}_j$ is measured using cosine similarity. If the similarity between p and the most similar cluster $\mathbf{c}^*$ is greater than the column clustering threshold θ, then we merge $\mathbf{p}$ to $\mathbf{c}^*$. Here, the operator $\oplus$ denotes vector addition. Otherwise, we form a new column cluster that contains $\mathbf{p}$ and append it to C_P. This procedure is repeated for entity pairs in Lines 7 and 8. The *while-loop* in Algorithm 1 is repeated until both E and P are empty. It is noteworthy that the operation of merging rows or columns in Line 22 changes the distributions of patterns and entity pairs, thereby directly influencing the subsequent similarity computations. For example, if a pattern p is merged into a column cluster c_j, then, in the next iteration, when we compute cosine similarity between entity pairs, all patterns in cluster c_j will be considered as forming a single dimension.

Algorithm 1 can be considered as a co-clustering extension of the one-sided sequential clustering proposed by Bollegala et al. [8]. However, as we demonstrate experimentally in Section 3.1, the co-clustering version produces better results than its one-sided counterpart that does not cluster entity pairs. The sorting operations in Algorithm 1 require respectively, $O(|E|\log|E|)$ and $O(|P|\log|P|)$ complexities for entity pairs and lexical-syntactic patterns, where $|S|$, denotes the cardinality of a set S. This sorting operation is required only once at the start. The while-loop starting from Line 4 in Algorithm 1 terminates after $\max(|E|, |P|)$ iterations. The greedy nature of the algorithm avoids combinatorial pairwise comparisons among all entity pairs or all lexical-syntactic patterns.

Most co-clustering algorithms [14, 17] require the number of row and column clusters to be given as inputs. However, we do not know the exact number of relations in the given corpus from which we must extract relations. Alternatively, Algorithm 1 has two parameters, θ and ϕ, which indirectly specify the number of clusters. A popular approach to setting the number of clusters, cross-validation, tries different combinations of parameter values and evaluates some goodness criteria on a set of held out data. However, an exhaustive search in parameter space is infeasible for large datasets. Consequently, we propose an efficient heuristic to determine the optimal values of the clustering thresholds.

First, we consider the distribution of similarity scores in a dataset of T points, portrayed in Figure 1. There are $T(T-1)/2$ pairs in a dataset of T points, between which we compute the similarity scores. Figure 1 plots the normalized frequency (i.e., dividing the frequency counts by $T(T-1)/2$) such that the total area of the blue bars equals one). The solid green line connects the midpoints of the bars in the histogram and represents the distribution of similarity scores. Figure 1 shows results obtained using the ENT dataset

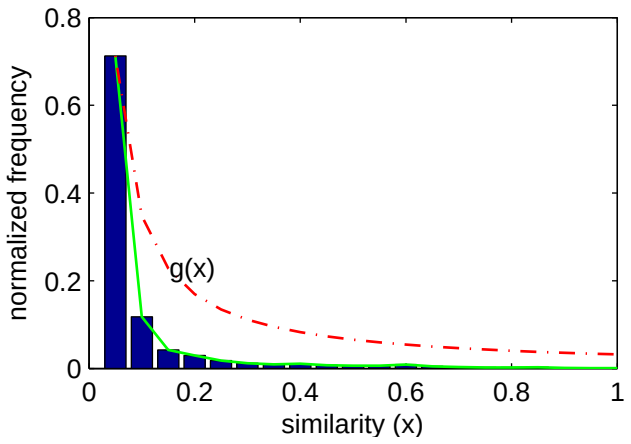

Figure 1: Approximating the distribution of similarity scores between data points. Bars show the actual histogram and the solid green line connects the mid-points of the bars. Approximation $g(x)$ is shown as a dotted red line.

(described later in Section 3.1), thereby showing the distribution of similarity in real data. The similarity among most data points tends to be very small in Open IE because, generally, there exist numerous semantic relations in a corpus and only a few between a given pair of entities. We approximate the actual distribution of similarity scores as a power-law curve and use this approximation to estimate the optimum values for clustering thresholds. This is analogous to Zipf's law [22], which is observed with word counts in large text corpora.

We define the power-law approximation $g(x)$ of the actual distribution of similarity scores x as

$$g(x) = ax^{-k}. \tag{1}$$

Here, a and k $(1 < k < 2)$ are real-valued constants. Assuming the bin size in the histogram in Figure 1 to be $\delta(< 1)$. Then, by fixing $g(x)$ at $x = \delta$, we have

$$g(\delta) = a\delta^{-k}. \tag{2}$$

By considering the area under the curve for g, we have

$$\int_{\delta}^{1} g(x)\,dx = 1, \tag{3}$$

From Formulas 2 and 3, we obtain

$$a = g(\delta)\delta^{k}, \tag{4}$$

$$k = \delta g(\delta) + 1. \tag{5}$$

Algorithm 1 adds a new entity pair (or a pattern) to an existing cluster only if the similarity between the cluster and the entity pair (or the pattern) exceeds the corresponding row or column clustering threshold. Therefore, with numerous data points (e.g., entity pairs or patterns), the average similarity between data points inside a cluster converges to the clustering threshold (proof omitted). In the case of optimum clustering, the similarity between data points in different clusters will be zero. Therefore, the average similarity among all data points (i.e., the mean $E_g(x)$ of the distribution $g(x)$) will be equal to the average similarity between data points inside clusters (i.e., the clustering threshold). For column clusters, the optimal threshold $\hat{\theta}$ is given as

$$\hat{\theta} \quad = \quad E_g(x) = \int_{\delta}^{1} xg(x)\,dx = \frac{a(1-\delta^{2-k})}{2-k}. \tag{6}$$

A similar expression can be derived for the optimum row clustering threshold $\hat{\phi}$ by considering the similarity distribution for entity pairs. It is noteworthy that the optimum value of the threshold given as Formula 6 depends only on δ and $g(\delta)$ (dependence on a and k can be eliminated using Formulas 4 and 5) . In practice, we set a small bin size (e.g. $\delta = 0.05$) and compute $g(\delta)$ as the fraction of data points that have similarity less than δ. Efficient algorithms have been proposed [30] that do not require pairwise comparisons of all data points to find the number of data points that have a similarity score that is less than a given threshold.

2.4 Self-supervised Relation Detection

In unsupervised relation extraction, the relation types to be extracted are an unknown prior. Co-clusters generated by Algorithm 1 capture numerous semantic relations that exist in a given corpus. For evaluation and presentation purposes, it is useful to label clusters with representative lexical patterns in them, which is particularly challenging for two reasons. First, we have no labeled data to train a supervised classifier to discriminate one cluster from another. Second, from among numerous lexical patterns, we must select the representative ones. We describe a self-supervised classification approach that overcomes both challenges.

The objective of Algorithm 1 is to produce a set of clusters where each cluster represents a different semantic relation between entity pairs. We assign a unique cluster i.d., y_k, to all entity-pairs e_i in a cluster C_k. We represent an entity pair e_i as a feature vector $\mathbf{e}_i$, in which the j-th feature is set to A_{ij}. We then model the problem of identifying representative lexical patterns in a cluster as one of discriminative feature selection in multi-class classification. Specifically, we use L1 regularized multi-class logistic regression, where the posterior probability of an entity pair e_i is given as

$$p(y_k|\mathbf{e_i}) = \frac{\exp(\mathbf{w}_k^{\mathrm{T}}\mathbf{e}_i)}{\sum_{y\in Y}\exp(\mathbf{w}_k^{\mathrm{T}}\mathbf{e}_i)}. \tag{7}$$

Here, Y denotes the set of cluster i.d.s; $\mathbf{w}_k$ is the weight vector associated with cluster C_k. We maximize the following w.r.t. $\mathbf{w}_k$, as

$$\sum_{(\mathbf{e}_i,y_i)} \log p(y_i|\mathbf{e}_i) - \sigma\sum_{k}||\mathbf{w}_k||_1. \tag{8}$$

The first term in expression 8 is the log-likelihood of entity pairs; the second (regularization) term is the sum of L1 norms of weight vectors $\mathbf{w}_k$. We use the notation, $||\mathbf{x}||_1$ to denote the L1 norm of a vector $\mathbf{x}$. The effect of regularization towards overall training process is adjusted using the regularization coefficient $\sigma(> 0)$. We use Orthant–Wise Limited-memory Quasi-Newton (OWL-QN) method to solve the optimization problem defined in the expression shown in 8. Actually, L1 regularization is known to produce sparse weight vectors, where most of the weights are set to zero [24]. This feature is particularly useful for the current task because it enables us to identify the lexical patterns that discriminate one cluster from another. After training, we select the highest non-zero weighted lexical patterns from a cluster as the representatives of the semantic relation described by that cluster.

3. EXPERIMENTS

We evaluate the proposed method in three tasks. First, in Section 3.1, we use the pattern clusters produced using the proposed method to measure the relational similarity between entity pairs in the ENT benchmark dataset [8], and compare it to the previously proposed relational similarity measures. We empirically study the behavior of Algorithm 1 over the complete parameter space. More-

over, we subjectively evaluate the patterns selected by the self-supervised relation detection method.

Second, in Section 3.2, we compare the proposed method against three previously proposed Open IE systems on SENT500 benchmark dataset. This dataset contains 500 manually annotated sentences that describe four semantic relations between named-entity pairs. It has been used extensively in previous work on Open IE. This experiment is intended to demonstrate the effectiveness of the proposed method in Open IE. Previous work using the ENT dataset has not used syntactic patterns. Consequently, to compare the proposed method against results of previous works that used the ENT dataset, we limit ourselves to lexical patterns. We evaluate the benefit of using syntactic patterns with the SENT500 dataset.

Third, in Section 3.3, we employ the proposed method to identify 53 different relations in a social network system containing 35 million nodes, thereby demonstrating the accuracy and scalability of the proposed method in real-world relation extraction tasks. In all experiments, we fix the values for the one-pass extraction to $L = 5$, $g = 2$, and $G = 4$.

3.1 Relational Similarity

Relational similarity between two pairs of words is defined as the correspondence between semantic relations that exist between the two words in each word pair. For example, the two pairs, (*ostrich*,*bird*) and (*lion*,*cat*) are considered relationally similar because the relation ***X is a large Y*** holds between the two words **X** and **Y**, in each of those word pairs. Bollegala et al. [8] proposed a *supervised* method (RELSIM) to measure the relational similarity between two word pairs using a set of automatically extracted lexical pattern clusters. We employ the lexical-syntactic pattern clusters extracted using the proposed *unsupervised* method to measure the relational similarity between two word pairs. If the pattern clusters produced by the proposed method are useful to predict the relational similarity between given two word pairs, then it not only justifies the proposed method; it also demonstrates a useful application of it.

Following [8], we represent a word pair (a, b) as an n-dimensional vector $\mathbf{f}_{(a,b)}$, in which the k-th element, $\mathbf{f}^k_{(a,b)}$, is set to the total frequency of all lexical-syntactic patterns in a pattern cluster C_k with word pair (a, b). Under this representation, each pattern cluster C_k contributes a single feature to the feature vector for a word pair. Considering the fact that each pattern cluster is expected to represent a unique semantic relation, this feature representation can be regarded as a projection of a word pair over the space defined by semantic relations. We then measure the relational similarity (alternatively the relational distance) between two word pairs (a, b) and (c, d) using Mahalanobis distance between the corresponding feature vectors $\mathbf{f}_{(a,b)}$ and $\mathbf{f}_{(c,d)}$, which is given as

$$(\mathbf{f}_{(a,b)} - \mathbf{f}_{(c,d)})^{\mathrm{T}} \Gamma^{-1} (\mathbf{f}_{(a,b)} - \mathbf{f}_{(c,d)}). \quad (9)$$

Here, Γ^{-1} denotes the inverse of the inter-cluster correlation matrix computed for pattern clusters. The (i, j) element of Γ is computed as the inner product between the centroid vectors of pattern clusters i and j. In contrast to the Euclidean distance, the Mahalanobis distance has shown to be more appropriate for the task of measuring relational similarity [8] because semantic relations are not independent.

We use the ENT dataset [8] as a gold standard of relational similarity. The ENT dataset contains 100 entity pairs describing the five semantic relations: **ACQUISITION** (between two companies, where one company is acquired by the other, e.g. (*Google*,*YouTube*)), **HEADQUARTERS** (between a company and the location of its headquarters, e.g. (*Microsoft*,*Redmond*)), **FIELD** (between a person and his field of expertise, e.g. (*Albert Einstein*,*Physics*)), **CEO** (between a company and its current CEO, e.g. (*Steve Jobs*,*Apple*)), and **BIRTHPLACE** (between a person and his place of birth, e.g. (*Charlie Chaplin*,*London*)). For each entity pair (a, b) of relation R in the ENT dataset, we measure the relational similarity between (a, b) and the remaining 99 entity pairs. A good relational similarity measure must assign higher similarity scores to entity pairs with similar semantic relations. Consequently, we evaluate the top k similar pairs to each entity pair in the dataset using average precision given as

$$\text{Average Precision} = \frac{\sum_{r=1}^{k} \mathrm{Pre}(r) \times \mathrm{Rel}(r)}{\text{no. of relevant entity pairs}}. \quad (10)$$

Here, $\mathrm{Rel}(r)$ is a binary valued function that returns 1 if the entity pair at rank r has the same relation (i.e. R) as in (a, b). Furthermore, $\mathrm{Pre}(r)$ is the precision at rank r, which is given as

$$\mathrm{Pre}(r) = \frac{\text{no. of entity pairs with relation } R \text{ in top } r \text{ pairs}}{r}. \quad (11)$$

In fact, the ENT dataset contains 20 entity pairs for each relation. Following previous work, we evaluate using the top 10 ranked results (i.e. $k = 10$).

The pattern extraction algorithm described in Section 2.2 extracts 142,655 lexical patterns from the text snippets provided in the ENT dataset. We then use Algorithm 1 to co-cluster both entity pairs and the extracted patterns. Clustering thresholds θ and ϕ are estimated respectively as 0.67 and 0.83 using Formula 6. This process produces 9 row (entity pair) clusters and 139 column (pattern) clusters. Table 2 presents a comparison of the relational similarity measured using the pattern clusters produced using the proposed method (**PROP**) against four others: **VSM** (Vector Space Model-based approach [33]), **LRA** (Latent Relational Analysis [33]), **EUC** (Euclidean distance between feature vectors [8]), and **RELSIM** (Mahalanobis distance between feature vectors [8]). Except for the proposed method, all other figures in Table 2 are obtained from previously published results obtained using the ENT dataset. Overall, PROP shows the highest average precision score (0.76) in Table 2. Moreover, for three out of the five relations in the ENT dataset, PROP outperforms all existing relational similarity measures. It is noteworthy that although RELSIM has a similar average precision score (0.74) to that of PROP, unlike PROP, which is unsupervised, RELSIM is a supervised method that requires labeled data for training. Moreover, both EUC and RELSIM use one-sided sequential clustering in which only patterns are clustered. In contrast, PROP clusters both patterns and entity pairs simultaneously using Algorithm 1, which exploits the dyadic structure in the data more effectively.

Table 2: Performance of the proposed method and previous work on relational similarity measures.

Relation	VSM	LRA	EUC	RELSIM	PROP
ACQUISITION	0.92	0.92	0.91	**0.94**	0.89
HEADQUARTERS	0.84	0.82	0.79	0.86	**0.97**
FIELD	0.44	0.43	0.51	**0.57**	0.42
CEO	0.95	0.96	0.90	0.95	**0.99**
BIRTHPLACE	0.27	0.27	0.33	0.36	**0.53**
Overall Average Precision	0.68	0.68	0.69	0.74	**0.76**

To study the behavior of the proposed sequential co-clustering algorithm empirically, we vary θ and ϕ in range $[0, 1]$, and measure the average precision over entity pairs in the ENT dataset using five-fold cross-validation. Average precision scores for various combinations of θ and ϕ values are shown in Figure 2. From Figure 2 it is readily apparent that for fixed low values of θ, average

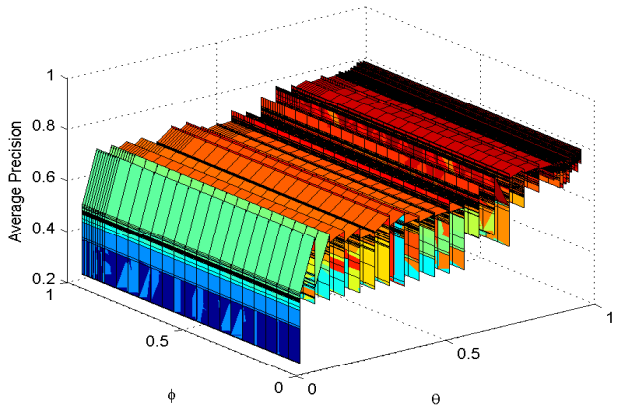

Figure 2: Average precision as a function of theta and phi.

Table 4: Subjective evaluation of patterns.

Relation	A	B	C	D
ACQUISITION	16.7%	40%	40%	3.3%
HEADQUARTERS	20%	40%	23.3%	16.7%
CEO	6.7%	53.3%	20%	20%
FIELD	13.3%	56.7%	23.3%	6.7%
BIRTHPLACE	13.3%	36.7%	10%	40%
overall	14%	45.3%	23.3%	17.3%

precision is insensitive w.r.t. ϕ because low θ values produce large and noisy pattern clusters that contain patterns for more than one semantic relation. However, when θ increases, the average precision also increases and the highest average precision score (0.78) is obtained for $\theta = 0.70$ and $\phi = 0.85$. When θ is increased further, we obtain numerous pattern clusters with very few patterns in them, thereby yielding sparse feature vectors. Consequently, the average precision drops for high θ values. The predicted values of θ (0.67) and ϕ (0.83) closely approximate the optimal parameter values obtained using cross-validation. However, conducting cross-validation over the full parameter space to determine threshold values is time consuming and is not scalable to large datasets. For example, on a computer with a Quad-Core processor (Intel Corp.) with 8GB of RAM, it takes more than 10 days to perform cross-validation over the parameter space as shown in Figure 2 for the ENT dataset. In stark contrast, the estimation method described in Section 2.3 requires less than 30 min to estimate the threshold values using the same computer.

To select the representative lexical patterns that describe the semantic relations in ENT dataset, we run the self-supervised relation detection method described in Section 2.4 as follows. First, the $142,655$ patterns extracted by the single-pass extraction method described in Section 2.2, are clustered into 139 pattern clusters and 9 entity pair clusters using Algorithm 1. Next, each entity pair is assigned with a pseudo-class label that indicates its entity pair (row) cluster. Next, multi-class logistic regression with L1 regularization is run on pseudo-labeled training instances. The regularization coefficient σ is set to its default value of 1. The self-supervised relation detection method assigns non-zero weights to 263 patterns, thereby producing a small subset of representative patterns. Table 3 shows the top 10 patterns with the highest weights in five clusters describing the relations in ENT dataset. For explanatory purposes, on the first row of Table 3 we have indicated the relation that is assigned in the ENT dataset for most entity pairs in a row cluster. The total number of patterns in a cluster is shown within brackets following the name of the relation. Two entities are indicated as **X** and **Y**, and N denotes a single numeric digit. From Table 3 it is readily apparent that lexical patterns that describe the target relations are identified correctly using the proposed method.

We subjectively compare the patterns presented in Table 3 against a baseline pattern selection method, which simply selects the most frequent pattern in each cluster. We asked three judges (excluding the authors of the paper) to grade the top 10 patterns extracted using the proposed method against the top 10 frequent patterns in each cluster using four grades: **A** (the pattern selected using the baseline method is better), **B** (the pattern selected using the proposed method is better), **C** (patterns selected using both methods are equally good), and **D** (patterns selected using both methods are bad). We elicit human judgments for 50 pairs of patterns (top 10 patterns for each of the 5 relations in the ENT dataset). Judges are not informed of the underlying nature of the baseline or the proposed method. Moreover, the ordering between the two patterns in each example is set randomly to avoid any bias in judgment towards a particular system. The inter-judge agreement measured using Fleiss's kappa is 0.4497 and is statistically significant ($p < 0.05$). Results of the subjective evaluation are presented in Table 4. Overall, patterns selected using the proposed method are preferred three times more than those selected using the baseline method. Particularly for CEO and FIELD relations, the patterns selected using the proposed method are preferred more than four times more than those obtained using the baseline method. Selecting frequent patterns from clusters usually produces ambiguous common patterns such as ***X and Y***, ***X or Y***, ***X is a Y***, etc. In contrast, the proposed method identifies discriminative patterns that uniquely identify a particular semantic relation.

3.2 Open Information Extraction

We evaluate the proposed relation extraction method on Open IE using the SENT500 [5] published benchmark corpus. This corpus contains 500 sentences; each sentence has one pair of entities. In fact, the SENT500 dataset includes 26 unique entity pairs. Some entities are represented by more than one name variant (e.g. *Adobe Systems Inc.* vs. *Adobe Systems*), thereby producing 65 unique name variant pairs in the corpus. Each entity pair in SENT500 dataset describes one of the four relation types: **ACQUISITION** (an acquisition relation between two companies), **BIRTHPLACE** (a person and that person's place of birth), **INVENTOR** (a person and that person's invention), and **WONAWARD** (a person and an award that person won). Previous works on Open IE have measured micro-averaged precision, recall, and the F-score of relation extraction using the SENT500 dataset. Therefore, by evaluating using the SENT500 dataset, we can directly compare the proposed method to previous work on Open IE.

We run the one-pass extraction method (Section 2.2) on SENT500 dataset, and extract both lexical and syntactic patterns. The extracted lexical and syntactic patterns are respectively, 947 and 384. The estimated values of the clustering thresholds θ and ϕ are respectively, 0.0005 and 0.01153. For those threshold values, Algorithm 1 produces 4 row clusters (corresponding to entity pairs) and 14 column clusters (corresponding to lexical-syntactic patterns). For evaluation purposes, we label each row cluster with the relation that exists between most entity pairs in the cluster. The proposed method (PROP) is compared against previous works on Open IE using the micro-average precision, recall and F-scores, as shown in Table 5. In Table 5, **O-NB** is the naive Bayes relation classifier described in [4], **O-CRF** is the conditional random field-based Open IE system described in [5], and **MLN** is the Markov logic network-based Open IE system described in [38]. To study the contribution of lexical and syntactic information for Open IE, we

Table 3: Lexical patterns selected using the self-supervised classifier

ACQUISITION (41)	HEADQUARTERS (30)	CEO (27)	FIELD (29)	BIRTHPLACE (26)
X acquires **Y**	**X** headquarters in **Y**	**Y** chairman and ceo **X**	**Y** legend **X**	**X** was born in **Y**
X to purchase **Y**	**X** head-office in **Y**	**X** , ceo of **Y**	**X** revolutionized **Y**	**Y** born **X**
X has purchased the **Y**	**X** laboratories **Y**	**Y** founder **X**	**Y** great **X**	**X** composer **Y**
X bids $ N.Nb for **Y**	**Y** -based **X**	**Y** -chef **X**	**Y** champion **X**	**X**'s birthplace in **Y**
X (formerly **Y**)	**X** in **Y**	**Y** ceo **X**, at	**X** retires from **Y**	**Y** native **X**
Y (now **X**	**X** building in **Y**	ceo **X** has **Y** on	**Y** star **X**	**X** left **Y** to
when **X** bought **Y**	**Y** office of **X**	**Y** chairman **X**	**X** and modern **Y**	**X**'s childhood in **Y**
X to acquire **Y**	past **X** offices in **Y**	**Y** ceo **X**	**X** referred to **Y**	**X** walk, **Y**
X buys **Y**	**Y** calif.-based **X**	**Y** . **X**	**Y** player **X**	**X** of **Y**
X buys ad firm **Y**	**X** .com **Y**	**X** has **Y** on the	**X** pga tour **Y**	**X** wird in **Y** city

Table 5: Evaluation of open IE on SENT500 dataset.

Method	Precision	Recall	F
O-NB [4]	0.866	0.232	0.366
O-CRF [5]	0.883	0.452	0.598
MLN [38]	0.798	0.733	0.764
PROP (lexical patterns)	**0.943**	0.647	0.767
PROP (syntactic patterns)	0.752	**0.860**	**0.802**
PROP (lexical+syntactic patterns)	0.751	0.857	0.801

Table 6: Classifying relations in a social network.

Relation	P	R	F	Relation	P	R	F
colleagues	0.76	0.87	0.81	friends	0.58	0.77	0.66
alumni	0.83	0.68	0.75	co-actors	0.75	0.74	0.74
fan	0.91	0.50	0.64	teacher	0.83	0.73	0.78
husband	0.89	0.57	0.74	wife	0.67	0.34	0.45
brother	0.79	0.60	0.68	sister	0.90	0.52	0.66
Micro	0.72	0.68	0.70	**Macro**	0.78	0.52	0.63

implement the proposed method in three flavors: using only lexical patterns (**PROP (lexical patterns)**), using only syntactic patterns (**PROP (syntactic patterns)**), and using both (**PROP (lexical+syntactic patterns)**). All figures presented in Table 5, except for the proposed method, are obtained from original publications. The highest precision among the different methods compared in Table 5 is reported using the proposed method using only lexical patterns. However, the use of syntactic information improves recall and consequently the best F-score is reported using the proposed method using only syntactic patterns. Lexical patterns are useful to detect specific relations between entities. On the other hand, syntactic patterns can generalize the relations that exist between entities, thereby improving the recall. Combining both lexical and syntactic patterns does not improve the performance beyond the mere use of syntactic patterns.

3.3 Relation Classification

To evaluate the scalability and performance of the proposed method in large real-world systems, we use the proposed method to classify relations between entities in an online social network mining system, SPYSEE[5]. This network contains $470,671$ nodes (people) and $35,652,475$ edges describing numerous relations between nodes. In fact, SPYSEE uses the automatic social network extraction method described by Matsuo et al.[23]; it is the largest of such systems in Japan. To mine the social network of a person P, first, SPYSEE uses the name of that person as the query to a Web search engine and downloads the top ranked search results. Subsequently, for each person name Q that appears in the downloaded search results, SPYSEE determines whether a relation exists between P and Q using various co-occurrence statistics such as the Jaccard coefficient and the overlap coefficient.

To apply the proposed method to identify relations between people in SPYSEE, we first run the single-pass extraction method described in Section 2.2 on the web pages that had been downloaded by SPYSEE for all personal names in the system. We then perform sequential co-clustering (Algorithm 1) to identify the entity pairs between which the same relation holds. Self-supervised relation detection method (Section 2.4) is used to label each entity pair cluster with representative lexical patterns that describe the relation between entities in the cluster. Manually evaluating the extracted relations among all the entity pairs in SPYSEE is impossible because of the extremely large number of entity pairs. Consequently, we randomly selected $50,000$ entity pairs (edges) from SPYSEE and evaluate on this subset. The proposed pattern extraction algorithm (Section 2.2) extracts $38,076$ unique lexical-syntactic patterns, out of which $11,193$ appear for more than two entity pairs. To avoid using noisy and rare patterns, which frequently contain misspellings and other irregularities, we consider only those $11,193$ patterns in the subsequent processing. Clustering thresholds θ and ϕ are estimated respectively, as 0.007 and 0.0131 using Formula 6. For those values of thresholds, Algorithm 1 produces 383 pattern clusters and 664 entity pair clusters.

We manually classified the entity pairs into 53 different relations (designated as the gold-standard), and evaluate the performance of the proposed method using micro- and macro-averaged precision (P), recall (R) and F-scores (F). Because of the limited availability of space, we show the clustering performance for randomly selected 10 relation types in Table 6. Two entities can share more than one relation. For example, colleagues can also be friends. Consequently, the gold standard assigns multiple relation types for such entity pairs. Table 6 shows that the proposed method correctly identifies numerous relations existing among people in a social network. In Figure 3, we present a comparison of the proposed co-clustering algorithm (Algorithm 1) against two other co-clustering algorithms: **MinSQD** (minimum sum-squared residue co-clustering) [11], and **ITCC** (information theoretic co-clustering) [14], using *Co-cluster*[6], a publicly available co-clustering tool. As shown in Figure 3, the proposed method is several orders of magnitude faster than MinSQD and ITCC over widely various dataset sizes. The end-to-end processing time taken by the proposed method to extract relations, cluster, and classify for the SPYSEE dataset is ca. 2 hr using a Quad-Core processor (Intel Corp.) with 8 GB of RAM.

4. RELATED WORK AND DISCUSSION

The approach proposed in this paper is motivated by work in three fields: relation extraction, relational similarity measurement, and co-clustering. Next, we discuss the previous work in those fields and compare it to the proposed method.

[5] http://spysee.jp/

[6] http://www.cs.utexas.edu/users/dml/Software/cocluster.html

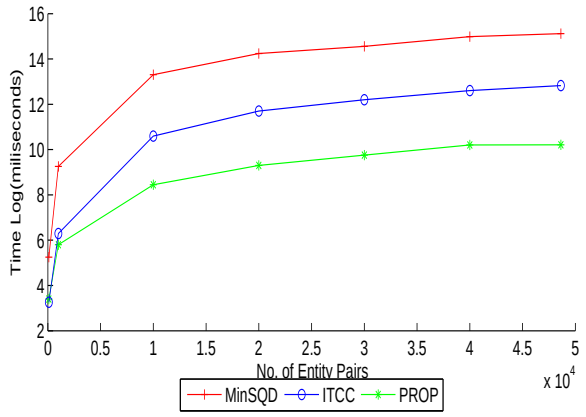

Figure 3: Comparison of processing times.

Traditionally, relation extraction is framed as a binary classification problem: Given a sentence S and a relation R, does S assert R between two entities in S? Supervised classification methods such as support vector machines (SVMs) with language-oriented kernels have been used to learn binary classifiers [10, 16, 18, 36, 37]. Roth and Yih [28] present a classification-based framework in which they jointly learn to identify named entities and relations. Culotta et al. [13] model the problem of relation extraction as a one of sequence labeling and used conditional random fields to identify the relations in a given document. Specifically, they perform relation extraction on biographical text in which the topic of each document is known in advance. Then, for each entity found in a document, their goal is to predict the relation between that entity and the topic of the document from a finite set of pre-defined relations. In our setting however, we do not know the relations that must be extracted beforehand. Moreover, the need for manually annotated training data by these supervised relation extraction systems makes it difficult to apply them to large-scale heterogeneous relation extraction tasks such as relation extraction from the web.

Bootstrapping methods [1, 9, 15, 25, 38] to relation extraction are attractive because they require markedly fewer training instances than supervised approaches do. Bootstrapping methods are initialized with a few instances (often referred to as seeds) of the target relation [1, 25, 38] or general extraction templates [15]. During subsequent iterations of the bootstrapping process, new extraction patterns are discovered and used to extract new instances. The quality of the extracted relations depends heavily upon the initial seeds provided to the bootstrapping system. If the extracted relations are of low quality, then we must restart with a different set of seeds and re-run the bootstrapping process. It might not be readily apparent to a non-expert user to devise good seeds of the target relation. Moreover, in a setting such as ours, in which we attempt to process a heterogeneous corpus such as Web text, it is not possible to know the target relations in advance, let alone provide seeds or extraction patterns for each relation.

Open Information Extraction (Open IE) [4, 5, 31] is a domain independent information extraction paradigm and has been studied in both the natural language document corpus [31], and the Web environment [4, 5] to extract relation tuples. Open IE systems are initialized with a few manually provided domain independent extraction patterns. To produce training data for the algorithm, dependency parsing is conducted on a text corpus; domain independent extraction patterns are used to identify correct extractions. Using the created training data, a classifier is trained to identify the correct instances of target relations. Although Open IE is closely related to the proposed method in the sense that they both extract unknown relations from heterogeneous corpora, the proposed method differs from the Open IE methods in several respects. First, unlike the proposed method, Open IE systems require human-selected features to learn a good extractor. Second, Open IE systems use deep linguistic parsing techniques to label training examples. In contrast, the proposed method uses cheaper and more robust linguistic processing and depends on an efficient co-clustering algorithm to produce training data for relation classification. Although the proposed method extracts unknown relations in a given text as done by Open IE systems, we go one step ahead and attempt to label the extracted relations. This step of identifying the extracted relations with informative lexical patterns is particularly important in any relation extraction system that attempts to extract unknown relations. It not only enables us to evaluate the accuracy of the extraction; it also provides a useful insight into which relations exist in a given corpus.

Relational similarity [8, 33, 34, 35] measures the correspondence between semantic relations existing between two pairs of words. For example, the relation ***X** is a large **Y*** exists between the two words in each word pair (*lion*, *cat*) and (*ostrich*,*bird*). Consequently, the two word pairs are considered to be relationally similar. The row clusters produced by the proposed co-clustering algorithm group entity pairs with the same semantic relation. Relational similarity measures first represent a pair of words as a vector of lexical patterns and then measure the distance between those vectors. A fundamental difference between relational similarity measures and the proposed method is that in relational similarity measures the two word pairs between which relational similarity must be computed are given in advance; the proposed method has the additional challenge of finding related word pairs. As Section 3.1 shows, the pattern clusters produced by the proposed method are useful to improve existing relational similarity measures.

The goal in co-clustering [3] is to cluster the rows and the columns of a given matrix simultaneously, such that the difference between the original matrix and the clustered matrix is optimal with respect to some objective function. Co-clustering algorithms [11, 14] that optimize different objective functions have been developed and used in a wide array of applications such as simultaneously clustering words and documents in information retrieval [14], and clustering genes and expression data for biological data analysis [11]. Co-clustering is an NP-hard problem [2] for which approximate algorithms have been proposed. However, most existing co-clustering algorithms require the number of row and column clusters as inputs, which is unknown in an open relation extraction setting. Moreover, the high computational complexities of these algorithms prohibit their application in large-scale relation extraction tasks. The sequential co-clustering algorithm presented in Section 2.3 does not require the number of clusters as input. It can efficiently cluster numerous data points, as described in Section 3.3.

5. CONCLUSION

We proposed a relation extraction method that exploits the dyadic nature in semantic relations within a co-clustering framework. Semantic relations that exist between numerous entities were represented using lexical-syntactic patterns. A one-pass extraction algorithm that can efficiently extract numerous expressive patterns was introduced. To cluster the extracted entity pairs and lexical-syntactic patterns simultaneously, we proposed an efficient sequential co-clustering algorithm. To identify the representative patterns that describe a particular semantic relation, we proposed a self-

supervised classification method. We evaluated the proposed method for three tasks: measuring relational similarity between entity pairs, Open IE, and classifying relations in an online social network. The pattern clusters extracted using the proposed method improved previously proposed relational similarity measures on the ENT benchmark dataset. Moreover, a subjective evaluation showed that the proposed self-supervised relation detection method can identify representative lexical patterns of a semantic relation. Experiments investigating the ability of the proposed method to conduct Open IE revealed that the proposed method outperforms all existing Open IE systems on a benchmark dataset of 500 sentences. The proposed method correctly classified 53 relation types in an online social network system with over 35 million edges, thereby proving its applicability in large real-world datasets.

6. REFERENCES

[1] E. Agichtein and L. Gravano. Snowball: Extracting relations from large plain-text collections. In *ICDL'00*, 2000.

[2] A. Anagnostopoulos, A. Dasgupta, and R. Kumar. Approximation algorithms for co-clustering. In *PODS '08*, pages 201–210, 2008.

[3] A. Banerjee, I. Dhillon, J. Ghosh, S. Merugu, and D. S. Modha. A generalized maximum entropy approach to bregman co-clustering and matrix approximation. *JAIR*, pages 1919–?1986, 2007.

[4] M. Banko, M. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni. Open information extraction from the web. In *IJCAI'07*, pages 2670–2676, 2007.

[5] M. Banko and O. Etzioni. The tradeoffs between traditional and open relation extraction. In *ACL'08*, pages 28–36, 2008.

[6] M. Berland and E. Charniak. Finding parts in very large corpora. In *ACL'99*, pages 57–64, 1999.

[7] R. Bhagat and D. Ravichandran. Large scale acquisition of paraphrases for learning surface patterns. In *ACL'08*, pages 674–682, 2008.

[8] D. Bollegala, Y. Matsuo, and M. Ishizuka. Measuring the similarity between implicit semantic relations from the web. In *WWW'09*, pages 651–660, 2009.

[9] S. Brin. Extracting patterns and relations from the world wide web. In *WebDB Workshop at EDBT'98*, pages 172 – 183, 1998.

[10] R. Bunescu and R. Mooney. Subsequence kernels for relation extraction. In *NIPS'06*, pages 171–178, 2006.

[11] H. Cho, I. Dhillon, Y. Guan, and S. Sra. Minimum sum-squared residue co-clustering of gene expression data. In *fourth SIAM Intl. Conf. on Data Mining*, pages 114–125, 2004.

[12] I. Copi. *Introduction to Logic*. Prentice Hall College Div, 1998.

[13] A. Culotta, A. McCallum, and J. Betz. Integrating probabilistic extraction models and data mining to discover relations and patterns in text. In *HLT/NAACL'06*, pages 296–303, 2006.

[14] I. Dhillon, S. Mallela, and D. Modha. Information-theoretic co-clustering. In *KDD'01*, pages 89–98, 2003.

[15] O. Etzioni, M. Cafarella, D. Downey, A. Popescu, T. Shaked, S. Soderl, D. S. Weld, and E. Yates. Unsupervised named-entity extraction from the web: An experimental study. *Artificial Intelligence*, 165:91–134, 2005.

[16] C. Giuliano, A. Lavelli, and L. Romano. Exploiting shallow linguistic information for relation extraction from biomedical literature. In *EACL'06*, pages 401–408, 2006.

[17] Q. Gu and J. Zhou. Co-clustering on manifolds. In *Proc. of KDD'09*, pages 359–367, 2009.

[18] A. Harabagiu, C. A. Bejan, and P. Mor?arescu. Shallow semantics for relation extraction. In *IJCAI'05*, pages 1061–1066, 2005.

[19] Z. Harris. Distributional structure. *Word*, 10:146–162, 1954.

[20] M. Hearst. Automatic acquisition of hyponyms from large text corpora. In *COLING'92*, pages 539–545, 1992.

[21] D. Lin and P. Pantel. Dirt: Discovery of inference rules from text. In *SIGKDD'01*, pages 323–328, 2001.

[22] C. D. Manning and H. Schütze. *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts, 2002.

[23] Y. Matsuo, J. Mori, M. Hamasaki, K. Ishida, T. Nishimura, H. Takeda, K. Hasida, and M. Ishizuka. Polyphonet: An advanced social network extraction system. In *WWW'06*, 2006.

[24] A. Y. Ng. Feature selection, l1 vs. l2 regularization, and rotational invariance. In *ICML'04*, pages 78–85, 2004.

[25] M. Pasca, D. Lin, J. Bigham, A. Lifchits, and A. Jain. Organizing and searching the world wide web of facts - step one: the one-million fact extraction challenge. In *AAAI'06*, pages 1400–1405, 2006.

[26] J. Pei, J. Han, B. Mortazavi-Asi, J. Wang, H. Pinto, Q. Chen, U. Dayal, and M. Hsu. Mining sequential patterns by pattern-growth: the prefixspan approach. *IEEE TKDE*, 16(11):1424–1440, 2004.

[27] D. Ravichandran and E. Hovy. Learning surface text patterns for a question answering system. In *ACL '02*, pages 41–47, 2001.

[28] D. Roth and W. Yih. A linear programming formulation for global inference in natural language tasks. In *CoNLL'04*, pages 1–8, 2004.

[29] G. Salton and C. Buckley. *Introduction to Modern Information Retreival*. McGraw-Hill Book Company, 1983.

[30] S. Sarawagi and A. Kirpal. Efficient set joins on similarity predicates. In *SIGMOD '04*, pages 743–754, 2004.

[31] Y. Shinyama and S. Sekine. Preemptive information extraction using unrestricted relation discovery. In *HLT-NAACL'06*, pages 304–311, 2006.

[32] R. Snow, D. Jurafsky, and A. Ng. Learning syntactic patterns for automatic hypernym discovery. In *NIPS'05*, pages 1297–1304, 2005.

[33] P. Turney. Measuring semantic similarity by latent relational analysis. In *IJCAI'05*, pages 1136–1141, 2005.

[34] P. Turney. Expressing implicit semantic relations without supervision. In *COLING-ACL'06*, pages 313–320, 2006.

[35] T. Veale. Wordnet sits the sat: A knowledge-based approach to lexical analogy. In *ECAI'04*, pages 606–612, 2004.

[36] D. Zelenko, C. Aone, and A. Richardella. Kernel methods for relation extraction. *JMLR*, 3:1083–1106, 2003.

[37] G. Zhou, M. Zhang, D. H. Ji, and Q. Zhu. Tree kernel-based relation extraction with context-sensitive structured parse tree information. In *EMNLP-CoNLL*, pages 728 – 736, 2005.

[38] J. Zhu, Z. Nie, X. Liu, B. Zhang, and J. R. Wen. Statsnowball: a statistical approach to extracting entity relationships. In *WWW'09*, pages 101–110. ACM, 2009.