Minimally Supervised Novel Relation Extraction using a Latent Relational Mapping

Danushka Bollegala *Member, IEEE*, Yutaka Matsuo, and Mitsuru Ishizuka *Member, IEEE*,

Abstract—The World Wide Web includes semantic relations of numerous types that exist among different entities. Extracting the relations that exist between two entities is an important step in various Web-related tasks such as information retrieval, information extraction, and social network extraction. A supervised relation extraction system that is trained to extract a particular relation type (source relation) might not accurately extract a new type of a relation (target relation) for which it has not been trained. However, it is costly to create training data manually for every new relation type that one might want to extract. We propose a method to *adapt* an existing relation extraction system to extract new relation types with minimum supervision. Our proposed method comprises two stages: *learning a lower-dimensional projection* between different relations, and *learning a relational classifier* for the target relation type with instance sampling. First, to represent a semantic relation that exist between two entities, we extract lexical and syntactic patterns from contexts in which those two entities co-occur. Then, we construct a bipartite graph between relation-specific and relation-independent patterns. Spectral clustering is performed on the bipartite graph to compute a lower-dimensional projection. Second, we train a classifier for the target relation type using a small number of labeled instances. To account for the lack of target relation training instances, we present a one-sided under-sampling method. We evaluate the proposed method using a dataset that contains 2000 instances for 20 different relation types. Our experimental results show that the proposed method achieves a statistically significant macro-average F-score of 62.77. Moreover, the proposed method outperforms numerous baselines and a previously proposed weakly-supervised relation extraction method.

Index Terms—Relation Extraction, Domain Adaptation, Web Mining

1 Introduction

The World Wide Web contains information related to numerous real-world entities (e.g. persons, locations, organizations, etc.) interconnected by various semantic relations. Accurately detecting the semantic relations that exist between two entities is of paramount importance for numerous tasks on the Web such as information retrieval (IR) [1], information extraction (IE) [2], and social network extraction [3]. For example, to improve coverage in information retrieval, a query about a particular person can return documents describing the various semantic relations that the person under consideration has with other related entities. Recent work on relation extraction has demonstrated that supervised machine learning algorithms coupled with intelligent feature engineering provide state-of-the-art solutions to this problem [4]–[6]. However, supervised learning algorithms depend heavily on the availability of adequate labeled data for the target relation types that must be extracted. Considering the potentially numerous semantic relations that exist among entities on the Web, it is costly to create labeled data manually for each new relation type that we want to extract. Instead of annotating a large set of training data manually for each new relation type, it would be cost effective if we could somehow *adapt* an existing relation extraction system to those new relation types using a small set of training instances. As described in this paper, we examine *relation adaptation* – how to adapt an existing relation extraction system that is trained to extract some specific relation types, to extract new relation types in a minimally-supervised setting. We designate the existing relation types on which a relation extraction system has been trained as *source* relations, whereas the novel relation type to which we must adapt is called the *target* relation.

We must overcome three fundamental challenges when adapting a relation extraction system to new relation types. First, a semantic relation that exists between two entities can be expressed using more than one lexical or syntactic pattern. For example, the acquiredBy relation that exist between two companies **X** and **Y** where the company **X** is acquired by the company **Y** can be expressed using lexical patterns such as **X** *acquires* **Y**, **X** *buys* **Y**, and **X** *purchases* **Y**. To classify a relation accurately, we must recognize the different ways in which it can be expressed on the Web. Second, the types of relations are strongly dependent on the application domain. For example, in the *financial* domain, we might be interested in extracting relations such as acquiredBy (between two companies) and ceoOf (between a company and a person), whereas, in the *movie* domain we might be interested in extracting relations such as actedIn (between an actor and a movie) and directed (between a director and a movie). Therefore, a classifier trained on the financial domain might not be applied directly to classify relations in the movie domain because the two domains have different sets of relations. Third, the labeled instances for the target relation are markedly fewer than those for

• *The University of Tokyo, Japan.*
danushka@iba.t.u-tokyo.ac.jp

the source relations. It is challenging to learn a classifier for the target relation type using such an unbalanced dataset.

We propose a two-stage approach to adapt an existing relation extraction system to new relation types. First, to represent a semantic relation $\mathcal{R}$ that exist between two entities A and B, we extract lexical and syntactic patterns from contexts in which those two entities co-occur. Our proposed method is inspired by the observation that different semantic relations share some lexical and syntactic patterns. For example, the lexical pattern **X** *direct* **Y** holds between a company **Y** and its CEO **X**, as well as between a country **Y** and its leader (e.g. prime minister or president) **X**. We designate patterns that appear in different relation types as *relation-independent* patterns, whereas patterns that appear only in a particular relation type are called *relation-specific* patterns. To identify relation-specific and relation-independent patterns, we propose the use of the entropy of a pattern over the distribution of entity pairs. If a pattern is distributed uniformly over entity pairs that belong to numerous semantic relations, then such patterns will have a high entropy. We then create a bipartite graph between relation-specific and relation-independent patterns and perform spectral clustering on this graph to compute a lower-dimensional mapping between relation-specific and relation-independent patterns. Spectral clustering attempts to minimize the normalized cut on the bi-partite graph between relation specific and relation independent patterns, thereby *aligning* the two types of patterns in a lower-dimensional space. The clusters formed by this process capture lexical patterns from source relation types as well as the target relation type. Consequently, we can use the lower-dimensional mapping created from this process to *project* feature vectors to train a relational classifier.

In the second stage, we train a classifier for the target relation type using training instances for both source and target relation types. A fundamental problem in training a relational classifier for a target relation type for which only a few labeled instances are available is that, because of the numerous source relation instances, the finally trained classifier becomes biased towards the source relation types. Any information related to the target relation type is overshadowed by the numerous source relation instances. To solve this problem, we propose a method that first samples a subset of source relation instances. Then we use that subset to train a classifier for the target relation type. This method reduces the imbalance between source and target relation datasets, thereby improving the classification accuracy for the target relation type.

To evaluate the performance of the proposed method to adapt to numerous relation types, we create a dataset that contains 2000 entity pairs for 20 different relation types such as actedIn (between an actor and a movie), leaderOf (between a leader and an organization/country), directedIn (between a film director and a movie), etc. We compare the proposed method against various baselines and a previously proposed weakly-supervised relation classification method. In our experiments, the proposed method significantly outperforms other methods compared herein, thereby demonstrating its capability of adapting accurately to numerous relation types.

The remainder of this paper is organized as follows. In Section 2.1, we formally define the *relation adaptation* problem. We provide a motivating example for the proposed method in Section 2.2. The procedure for representing semantic relations using lexical and syntactic patterns is described in Section 2.3. Three strategies to identify relation-specific and relation-independent patterns are presented in Section 2.4. In Section 2.5, we construct a bipartite graph between relation-specific and relation-independent pattern. In Section 2.6, we perform spectral clustering on the created bipartite graph to compute a latent relational mapping between relation-specific and relation-independent features. A one-sided under-sampling method is proposed in Section 2.7 to select a subset of source relation instances which are used together with target relation instances to train a classifier. In Section 3, we conduct a series of experiments using a dataset that contains numerous relation types to evaluate the ability of the proposed method to classify novel target relation types. Finally, we present the related work in Section 4 and conclude the paper.

2 Relation Adaptation

2.1 Problem Definition

Given two entities A and B, we define relation extraction as the task of selecting the relation $\mathcal{R}$, that exists between A and B, from a given set of relation types. Note that this definition of relation extraction is different from that used, for example in bootstrapping and Open IE systems (discussed later in Section 4), because we assume that we already know the set of relation types from which we must select a relation type for a given entity pair. Moreover, entity pair (A, B) is regarded as an *instance* of the relation $\mathcal{R}$. For example, the entity pair (*Steven Spielberg*, *Firelight*) is an instance of the relation directed. According to our definition, relation extraction can be modeled as a multi-class classification problem. For instance, we can label entity pairs for each of the relation types that we want to extract, and use the labeled entity pairs to train a supervised multi-class classifier.

Definition: We define **Relation Adaptation** as the problem of learning a classifier for a target relation type $\mathcal{T}$, for which we have a few entity pairs as training instances, given numerous entity pairs for some N source relation types, $\mathcal{S}_1, \ldots, \mathcal{S}_N$. We use the notation $\Omega = \{\mathcal{S}_1, \ldots, \mathcal{S}_N, \mathcal{T}\}$ to denote the set of all relations. A particular relation type from this set is denoted by $\mathcal{R}$ (i.e $\mathcal{R} \in \Omega$). An entity pair that consists of two entities A and B is denoted as (A, B). Moreover, we use the notation

$(A, B) \in \mathcal{R}$ to indicate that the relation $\mathcal{R}$ exists between two entities A and B.

The above-mentioned definition of relation adaptation assumes the availability of labeled data for source relations as well as for the target relation. However, the amount of labeled data available for the target relation is assumed to be very small, limited to several seed instances. We do not assume the availability of unlabeled data in this work.

Note that there might exist more than one relation between an entity pair. For example, both the ceoOf and founderOf relations exist between the two entities *Steve Jobs* and *Apple*. It is possible to extend our definition of relation adaptation to incorporate multiple relations between entities by considering multi-class multi-label classifiers. In this paper, we limit ourselves to assigning a single relation type to a given entity pair.

On the other hand, there might not exist any semantic relation between two entities. One solution to this problem of filtering out entity pairs that are not related is to train a binary classifier as a pre-processing step that determines whether a semantic relation exists or not between two entities. Subsequently, entity pairs that are marked to be related by this binary classifier can be further processed by a relation adaptation method to assign a relation type. In this paper, we assume that all entity pairs contain some semantic relation and do not attempt filter out entity pairs with no relations.

2.2 A Motivating Example

Before we explain the proposed method in greater detail, let us first consider a motivating example that portrays the intuition underlying the proposed method. Consider two relations, leaderOf and ceoOf, as shown in Table 1. The leaderOf relation exists between a country and its current leader, whereas the ceoOf relation exist between a company and the chief executive officer of that company. Assuming that we are given contexts in which instances of the relation leaderOf occurs, we intend to train a relational classifier for the ceoOf relation. The two relations under consideration have very different distributions. Consequently, a relational classifier trained on one relation might not correctly identify the other relation. In Table 1, two contexts are provided for the leaderOf relation instance entity pair (*George Bush*, *U.S.*) and for the ceoOf relation instance entity pair (*Steve Jobs*, *Apple*).

To represent a semantic relation, we extract lexical and syntactic patterns as described later in Section 2.3. To illustrate our example, we assume that we extract the lexical patterns shown within brackets alongside with contexts in Table 1. The pattern, **X** *direct* **Y** appears in both relation types. We designate such patterns as *relation-independent* (RI) patterns. However, patterns such as **Y** *president* **X** and **X** *ceo* **Y** appear in only one of the two relation types. We designate such patterns as *relation-specific* (RS) patterns. For relation adaptation, we assume

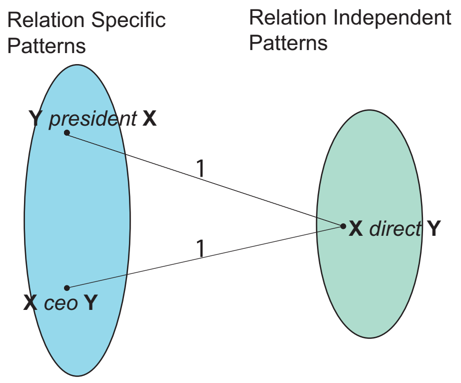

Fig. 1. A bipartite graph between relation-specific patterns and relation-independent patterns shown in Table 1

that we have sufficiently numerous source relation entity pairs, but only a few entity pairs for the target relation. Therefore, it is particularly challenging to learn proper weights for the target relation-specific patterns such as **X** *ceo* **Y**. However, relation-specific patterns in the target relation are extremely useful when determining whether a particular entity pair belongs to the target relation.

As a solution to this *mismatch* between source and target relation-specific patterns, we propose a method to find a mapping between source relations and the target relation using relation-independent patterns as pivots. First, Figure 1 shows that we create a bipartite graph between relation-specific patterns and relation-independent patterns. Each pattern is represented as a vertex in the bipartite graph. Two vertices are connected by a weighted undirected edge if the corresponding patterns are extracted from the same entity pair. For instance, in Table 1, the two patterns **X** *direct* **Y** and **Y** *president* **X** are extracted from contexts for the entity pair (*George Bush*, *U.S.*). Therefore, those patterns are connected by an edge in the bipartite graph portrayed in Figure 1. Similarly, the relation specific pattern **X** *ceo* **Y** and the relation independent pattern **X** *direct* **Y** are connected by an edge because those two patterns are extracted from the contexts for the same entity pair (**Steve Jobs**, **Apple**). Next, we perform spectral clustering on this bipartite graph to compute a latent mapping between relation-specific and relation-independent patterns. This mapping is then used to project feature vectors to train a relation classifier.

Several important points must be discussed. We must determine how to extract lexical and syntactic patterns to represent the semantic relations expressed by an entity pair, how to identify relation-specific and relation-independent patterns, how to weight the edges on the bipartite graph and how to construct a low-dimensional mapping using relation-specific and relation-independent patterns, and how to train a classifier with a few target relation instances. The following sections present discussions of these points.

TABLE 1
Example contexts for two relation types: leaderOf and ceoOf. Entities between which the specified relation exist are marked in **boldface**. Words that contribute to important lexical patterns are shown in *italic*. Some lexical patterns extracted by the proposed method are shown within squared brackets.

leaderOf (source relation)	ceoOf (target relation)
President **George Bush** directed **U.S.** to an unnecessary war against Iraq. [**X** *direct* **Y**]	**Steve Jobs** personally directs **Apple** and makes final decisions on various UI designs. [**X** *direct* **Y**]
U.S. president *George Bush* attended the G8 summit last month. [**Y** *president* **X**]	**Steve Jobs** is the CEO of **Apple**, which he co-founded in 1976. [**X** *ceo* **Y**]

2.3 Relation Representation

The contexts in which two entities A and B co-occur on the Web provide useful clues to the relations existing between those entities. We use the term *context* to refer a window of text in which two entities co-occur. A context might not necessarily be a complete sentence. Retrieving contexts in which two entities co-occur has been studied in previous works investigating the relations between entities on the Web [7]–[9]. Two main approaches are identifiable. First, given a large Web crawl, we can select textual windows that contain the two entities A and B in web documents [7], [9] However, disadvantages of this method include the high costs of crawling, storing, and processing a large text corpus [10]. Moreover, if the crawled data is insufficient, then the entities might not co-occur, which in turn engenders data sparseness. A second approach is to issue various queries including the two entities to an existing Web search engine and to retrieve search engine snippets (or entire web pages) that contain both entities [8]. This approach is cheaper because it obviates the need to crawl, store or index Web documents. Unfortunately however, the results that can be retrieved from a Web search engine are often of a limited number. Numerous solutions have been proposed in previous work to circumvent this problem [8], [10]. In our work, we assume that we are provided with contexts in which entities co-occur and only specifically examine the relation adaptation problem. For the experiments described in this paper we use the Yahoo BOSS API to retrieve contexts from the Web following the method described in [8].

Given a pair of entities (A, B), the first step is to express the relation between A and B using some feature representation. Lexical or syntactic patterns have been successfully used in numerous natural language processing tasks involving relation extraction such as extracting hypernyms [11], [12] or meronyms [13], question answering [14], and paraphrase extraction [15]. Following the previous work on relation extraction between entities, we use lexical and syntactic patterns extracted from the contexts in which two entities co-occur to represent the semantic relation that exists between those entities.

First, we lemmatize and part-of-speech (POS) tag the contexts using Python Natural Language Processing Tool Kit (NLTK)[1]. Table 2 presents an example in which we extract patterns from a context selected for the two entities, *Adobe Systems* and *Macromedia*, between which the relation acquiredBy exist. Next, both in the surface form and POS tag sequence, we replace the first entity (i.e. *Adobe Systems*) with a placeholder variable **X**, and the second entity (i.e. *Macromedia*) with a different placeholder variable **Y**. In relation adaptation, the entity pairs are given as input. We need only to detect the relations between those entities. Therefore, we need not recognize entities in a context. We use the subsequence pattern extraction algorithm proposed by Bollegala et al. [16] to extract lexical and syntactic patterns from contexts. Next, we briefly outline the steps in this algorithm. (refer to [16] for additional details).

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

(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 τ tokens.
(iii). A subsequence is allowed to have gaps (i.e. one or more skipped tokens). However, we do not allow gaps of more than g 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 pattern **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 τ, G, and g are set respectively to 5, 2, and 4 following [16].

The above-described subsequence pattern extraction algorithm presents several interesting properties. First, it considers all the words in a context, and is *not* limited to extracting patterns only from the mid-fix (i.e., the portion of text in a context that appears between a pair of entities). Moreover, the consideration of gaps enables us to capture relations between entities located at some distance in a context. We use *prefixspan* algorithm [17] to generate the subsequences efficiently. The constraints (i)-(iv) listed above are used to prune the search space, thereby reducing the number of subsequences generated

1. http://www.nltk.org

TABLE 2
Extracting lexical and syntactic patterns from a context retrieved for the entity pair *Adobe Systems* and *Macromedia*.

context	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 .
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**

by prefixspan. Some lexical and syntactic patterns extracted using the proposed method are shown in Table 2 (not all patterns are shown because of the limited availability of space). Finally, all lexical and syntactic patterns extracted from all contexts in which two entities A and B co-occur are arranged in a pattern frequency vector $\boldsymbol{x}_{AB}$ to represent the entity pair (A, B). The elements of $\boldsymbol{x}_{AB}$ correspond to, $f(\rho, A, B)$, the total number of times a pattern ρ is extracted from contexts in which A and B co-occur. It is analogous to the term frequency vector used in IR.

2.4 Relation-Specific vs. Relation-Independent Patterns

Once we express the relations that exist between entities using lexical and syntactic patterns as described in the previous section, we compute the correspondence between patterns that express different semantic relations. First, we must identify which patterns are specific to a particular relation type. We present three strategies to identify relation-specific patterns.

The first strategy is to select patterns that occur more than ζ times in all relation types as relation independent patterns. Here, the total frequency of a pattern ρ in a particular relation type $\mathcal{R}$ is defined as the sum of the frequencies of ρ in all entity pairs that belong to $\mathcal{R}$ (i.e. $\sum_{(A,B)\in\mathcal{R}} f(\rho, A, B)$). Given the number l of relation independent patterns to be selected, we set ζ to the largest number such that we can get at least l relation-independent patterns. Given l, ζ is uniquely determined.

The second strategy for selecting relation-independent patterns is based on the mutual information between a pattern and a relation type. Mutual information is a measure of the mutual dependence between two random variables. In previous works examining cross-domain sentiment classification, mutual information between a feature and a domain label is used as a criterion to select domain-independent features [18], [19]. The (pointwise) mutual information $I(\rho, \mathcal{R})$ between a pattern ρ and a relation type $\mathcal{R}$ is defined as follows:

$$I(\rho, \mathcal{R}) = p(\rho, \mathcal{R}) \log_2 \left(\frac{p(\rho, \mathcal{R})}{p(\rho)p(\mathcal{R})} \right). \tag{1}$$

Here, $p(\rho, \mathcal{R})$ is the joint probability of a pattern ρ and a relation $\mathcal{R}$; it is given as

$$p(\rho, \mathcal{R}) = \frac{\sum_{(A,B)\in\mathcal{R}} f(\rho, A, B)}{\sum_{\rho\in\Phi} \sum_{\mathcal{R}\in\Omega} \sum_{(A,B)\in\mathcal{R}} f(\rho, A, B)}. \tag{2}$$

Here, we use Φ to denote the set of all (lexical and syntactic) patterns extracted for all entity pairs in all relations in Ω. Moreover, the total number of patterns in Φ is denoted by n. We can compute the marginal probabilities $p(\rho)$ and $p(\mathcal{R})$ in Equation 1 by marginalizing the joint probability $p(\rho, \mathcal{R})$ in Equation 2 respectively over $\mathcal{R}$ and ρ. To select relation-independent patterns, we consider the sum of mutual information given by Equation 1 over the set of relations Ω (i.e. $\sum_{\mathcal{R}\in\Omega} I(\rho, \mathcal{R})$) for each pattern. The smaller this value is, the more likely that the pattern ρ is relation-independent. We select the l number of patterns with the lowest mutual information as relation-independent patterns.

We propose a third strategy for selecting relation independent patterns using the entropy of a pattern over the distribution of entity pairs. The proposed strategy is inspired by the fact that if a pattern is relation-independent, then its distribution over the entity pairs tends to become more uniform. However, if a pattern is relation-specific, then its distribution is concentrated over a small set of entity pairs that belong to a specific relation type. The entropy of a pattern increases as its distribution becomes more uniform. The entropy, $H(\rho)$, of a pattern ρ is computed as

$$H(\rho) = \sum_{\mathcal{R}\in\Omega} \sum_{(A,B)\in\mathcal{R}} p(\rho, A, B) \log_2 p(\rho, A, B). \tag{3}$$

Here, the joint probability between a pattern ρ and an entity pair (A, B) is given as

$$p(\rho, A, B) = \frac{f(\rho, A, B)}{\sum_{\rho\in\Phi} \sum_{\mathcal{R}\in\Omega} \sum_{(A,B)\in\mathcal{R}} f(\rho, A, B)}. \tag{4}$$

Figure 2 presents an example in which we plot the distributions over entity pairs (numeric ids are assigned to entity pairs and grouped by their relation types for illustrative purposes) for four lexical patterns. From Figure 2, it is apparent that relation-specific patterns such as **Y** *directed by* **X** (directed relation), and **Y** *wife* **X** (isMarriedTo relation) are concentrated over a small set of entity pairs, whereas relation-independent patterns such as **Y** *from* **X**, and **Y** *for* **X** are distributed over a large set of entity pairs. Consequently, relation-independent patterns have higher entropy values than relation-specific patterns do.

2.5 Bipartite Graph Construction

For the set of all patterns Φ (total no. of patterns, $|\Phi| = n$) extracted by the pattern extraction method described in

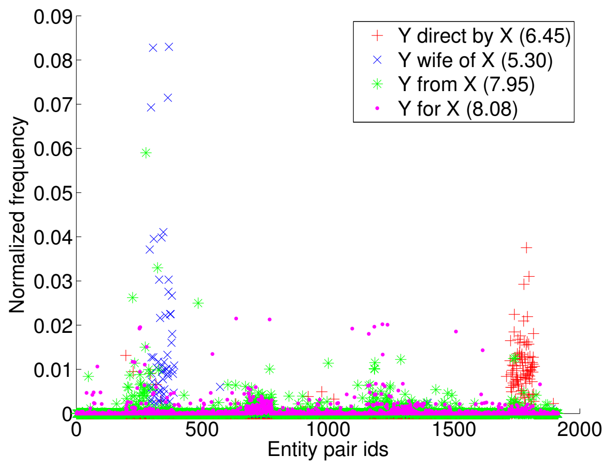

Fig. 2. Distributions of four patterns over entity pairs. Entropies are shown within brackets.

Section 2.3 for all entity pairs in source relations and the target relation (i.e. Ω), we can use one of the strategies described in Section 2.4 to identify a set $\Phi_{RS} \subseteq \Phi$ of relation-specific patterns, and a set $\Phi_{RI} \subseteq \Phi$ of relation-independent patterns. Here, $\Phi_{RS} \cup \Phi_{RI} = \Phi$ and $\Phi_{RS} \cap \Phi_{RI} = \phi$. In this section, we construct a bipartite graph, $G = (V_{RS} \cup V_{RI}, E)$ between relation-specific and relation-independent patterns to represent the intrinsic relationship between those patterns. Each vertex in V_{RS} corresponds to a relation-specific pattern, and each vertex in V_{RI} corresponds to a relation-independent pattern. A vertex in V_{RS} (corresponding to a relation-specific pattern $\rho_i \in \Phi_{RS}$) is connected to a vertex in V_{RI} (corresponding to a relation-independent pattern $\rho_j \in \Phi_{RI}$) by an undirected edge $e_{ij} \in E$. Note that there are no intra-set edges connecting vertices in V_{RS} and V_{RI}. Moreover, each edge $e_{ij} \in E$ is associated with a non-negative weight m_{ij}, that measures the strength of association between the corresponding patterns ρ_i and ρ_j. We set m_{ij} to the number of different entity pairs from which both ρ_i and ρ_j are extracted. If two patterns are extracted from numerous entity pairs, then those patterns can be considered as semantically similar according to the distributional hypothesis [20]. Previous work on pattern clustering [21] and relation extraction [16] have modeled patterns using their distributions over entity pairs to compute distributional similarity. Edge weights m_{ij} are represented collectively by an edge-weight matrix $\mathbf{M}$ of the bipartite graph G. Figure 1 portrays a bipartite graph constructed from the example shown in Table 1. We use the constructed bipartite graph to model the intrinsic relationship between relation-specific and relation-independent patterns.

It is noteworthy that, aside from using the number of different entity pairs from which two patterns are extracted, we can use numerous other methods to measure the strength of association, m_{ij}, between two patterns. For example, we can use the distributional similarity between the two patterns over the entity pair distributions, or use popular co-occurrence measures such as pointwise mutual information, Jaccard coefficient, Dice coefficient etc. In this paper, for simplicity, we use the number of different entity pairs from which two patterns are extracted as the edge-weighting measure. We want to show that by constructing a simple bipartite graph and applying spectral clustering techniques on it, we can accurately map patterns from source relations to the target relation.

2.6 Relational Mapping

In this section, we propose an algorithm based on spectral graph theory [22] to find a lower-dimensional mapping for patterns extracted from different relation types. This lower-dimensional mapping is used to project pattern frequency vectors created for entity pairs, thereby reducing the mismatch between patterns extracted for source relations and the target relation. There are two main assumptions in spectral graph theory: (1) if two vertices in a graph are connected to many common vertices, then those two vertices must be similar, and (2) there exists a low-dimensional latent space underlying a complex graph, in which two vertices are mutually similar if they are also similar in the original graph. Based on those two assumptions, spectral graph theory has been applied to widely various problems such as document clustering [23], dimensionality reduction [24], [25] and object recognition [26], [27]. In relation adaptation, we assume that: (1) if two relation-specific patterns are connected to many common relation-independent patterns, then those relation-specific patterns must be mutually similar, (2) if two relation-independent patterns are connected to many common relation-specific patterns, then those relation-independent patterns must be mutually similar, and (3) there exist a lower-dimensional latent space in which similar patterns in the original space are located close together in this lower-dimensional space. Under those assumptions, we can use spectral graph theory to find a latent mapping between patterns extracted for source and target relation types, as shown in Algorithm 1.

Given as input an edge-weight matrix $\mathbf{M}$ for the bipartite graph G constructed in Section 2.5, and dimensionality $k (< n)$ of the latent space, Algorithm 1 returns a projection matrix from the original n dimensional pattern space to a k dimensional latent space. The (i, j) element of the edge-weight matrix $\mathbf{M}$ represents the weight of the edge that connects a relation-specific pattern ρ_i to a relation-independent pattern ρ_j. The first step in Algorithm 1 is to construct the affinity matrix $\mathbf{A}$ of the overall bipartite graph G. Because no edges exist among vertices that belong to V_{RS} and V_{RI}, the affinity matrix $\mathbf{A}$ for the entire bipartite graph can be constructed using $\mathbf{M}$ and the zero matrix $\mathbf{0}$ as shown in Line 1 in Algorithm 1. Next, we compute the normalized Laplacian $\mathbf{L}$ for the bipartite graph (Line 2) and compute the eigenvectors corresponding to the k smallest eigenvalues

Algorithm 1 Mapping patterns extracted from source relations and the target relation to a lower-dimensional space.

```
Input: An edge-weight matrix, $\mathbf{M} \in \mathbb{R}^{(n-l)\times l}$ of a bipartite graph $G(V_{RS} \cup V_{RI}, E)$, and the number of clusters (latent dimensions) $k$.
Output: A projection matrix, $\mathbf{U} \in \mathbb{R}^{n\times k}$.
1: Compute the affinity matrix, $\mathbf{A} \in \mathbb{R}^{n\times n}$, of the bipartite graph $G$ as $\mathbf{A} = \begin{bmatrix} \mathbf{0} & \mathbf{M} \\ \mathbf{M}^\top & \mathbf{0} \end{bmatrix}$.
2: Compute the Laplacian, $\mathbf{L}$, of the bipartite graph $G$ as $\mathbf{L} = \mathbf{I} - \mathbf{D}^{-1}\mathbf{A}$, where the diagonal matrix $\mathbf{D}$ has elements $D_{ii} = \sum_j A_{ij}$, and $\mathbf{I} \in \mathbb{R}^{n\times n}$ is the unit matrix.
3: Find the eigenvectors corresponding to the $k$ smallest eigenvalues of $\mathbf{L}$, $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k$, and arrange them in columns to form the projection matrix $\mathbf{U} = [\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k] \in \mathbb{R}^{n\times k}$.
4: return U
```

of $\mathbf{L}$ (Line 3). In previous work on spectral clustering [28], it has been shown that the k smallest eigenvectors of the Laplacian matrix can be used to cluster a set of data points by mapping them into a k dimensional space spanned by those eigenvectors. Moreover, the k smallest eigenvectors act as the continuous solution of the cluster membership indicators. Consequently, we arrange the k smallest eigenvectors, $\boldsymbol{u}_1, \ldots, \boldsymbol{u}_k$ in columns to construct a projection matrix $\mathbf{U}$ (Line 3). It is noteworthy that the smallest eigenvector of $\mathbf{L}$ is always a constant unit vector and does not provide any useful information related to the cluster membership. Consequently, we ignore this constant eigenvector when selecting the k smallest eigenvectors to construct $\mathbf{U}$. Once the projection matrix $\mathbf{U}$ is computed as described in Algorithm 1, we use it to project entity pairs that belong to different relation types into a common latent subspace. Specifically, for an entity pair (A, B) represented by a pattern-frequency vector $\boldsymbol{x}_{AB}$, its projection into the k-dimensional latent space is given as $\mathbf{U}\boldsymbol{x}_{AB}$.

We implemented the stochastic matrix decomposition algorithm proposed by Halko et al. [29] to find the smallest k eigenvectors directly without computing the entire spectrum of the Laplacian and then selecting only the smallest k eigenvectors. Moreover, it can be shown that the eigenvectors of the Laplacian $\mathbf{L} \in \mathbb{R}^{n\times n}$ can be constructed by computing the left and right singular vectors in the singular value decomposition of the matrix $\mathbf{M} \in \mathbb{R}^{(n-l)\times l}$, which has much smaller dimensions than $\mathbf{L}$ (see [23] for the proof). Together these techniques enable us to compute the projection matrix $\mathbf{U}$ efficiently from numerous lexical and syntactic patterns.

2.7 Relation Classification

The low-dimensional projection computed in the previous section reduces the mismatch between patterns in source and target relation types, thereby enabling us to train a classifier for the target relation type using labeled entity pairs for both source and target relation types. However, we must overcome two challenges before we can use the projected vectors to train a classifier for a target relation type: *loss of information because of imperfect projections*, and *imbalance between source and target relation training datasets*. Next, we discuss each challenge in detail and propose solutions to overcome them.

Algorithm 2 One-sided under-sampling algorithm to select a subset of source relation instances.

```
Input: Set $\Lambda$ that contains entity pairs for all source relations $\mathcal{S}_1, \ldots, \mathcal{S}_N$ and the target relation $\mathcal{T}$.
Output: A set $\Gamma \subseteq \Lambda$.
1: Initialize $\Gamma$ to the set containing all entity pairs of $\mathcal{T}$.
2: Randomly select an entity pair from each source relation $\mathcal{S}_i$ and add to $\Gamma$.
3: Classify $\Lambda$ with the 1-NN rule using the instances in $\Gamma$, and compare the assigned relation labels with the original ones.
4: Move all misclassified instances from $\Lambda$ into $\Gamma$.
5: return $\Gamma$
```

First, the criterion for selecting relation-independent and relation-specific patterns might not be perfect, thereby introducing some noise to the created bipartite graph. For that reason, the computed projection matrix might not accurately project features in the dimensionality reduction step. To compensate for the loss of information because of imperfect feature projection, we augment all the patterns in the original vector $\boldsymbol{x}_{AB} \in \mathbb{R}^{n\times 1}$ to the projection $\mathbf{U}\boldsymbol{x}_{AB} \in \mathbb{R}^{k\times 1}$ to construct a new representation $\tilde{\boldsymbol{x}}_{AB} \times \mathbb{R}^{(n+k)\times 1}$ for an entity pair (A, B) as

$$\tilde{\boldsymbol{x}}_{AB} = [\boldsymbol{x}_{AB}, \lambda\mathbf{U}\boldsymbol{x}_{AB}]. \quad (5)$$

The single scalar parameter λ is useful to balance the tradeoff between original and projected features in the new representation. Using a set of heldout data, we set λ such that the average L_1 norm on the source relation projection vectors $\mathbf{U}\boldsymbol{x}$ is equal to that of the original vectors $\boldsymbol{x}$. This new representation retains all the features (pattern frequencies) in the original vector in addition to the projected features, thereby overcoming any disfluencies attributable to potential imperfect projections.

Second, in relation adaptation, the number of target relation training instances (entity pairs) is significantly smaller than that of the source relations. Given such an unbalanced training dataset, most supervised classification algorithms treat the minority class (target relation) instances as noise or outliers. Therefore, learning a classifier for a target relation type which has only a few instances is difficult in practice. To overcome this problem, we use a one-sided under-sampling algorithm (Algorithm 2), which first selects a subset of the source relation training data and then uses that subset to train a multi-class classifier. This algorithm was first proposed by Tomek [30] as a modification to the condensed nearest-neighbor (CNN) method in pattern recognition. One-sided under-sampling methods have been used to select a subset of the majority class in previous work investigating the problem of machine learning with unbalanced datasets [31], [32].

Algorithm 2 takes a set, $\Lambda = \{(A,B)|(A,B) \in \mathcal{R}, \forall \mathcal{R} \in \Omega\}$, of all the entity pairs for source relations and the target relation, and creates a set $\Gamma \subseteq \Lambda$ that contains all target relation entity pairs and a subset of the source relation entity pairs. In Algorithm 2, we first select all target relation entity pairs as the set Γ (Line 1). In relation adaptation problem setting, the number of target relation entity pairs is small and we do not sample from the target relation (hence the name *one-sided* sampling). Next, we randomly select an entity pair from each source relation type and include those entity pairs in Γ (Line 2). We then use the single nearest neighbor (1-NN) rule to classify the entity pairs in Λ using the entity pairs in Γ as the labeled instances (Line 3). We use the augmented representation given in Equation 5 to represent an entity pair and measure the Euclidean distance between feature vectors to identify the nearest neighbors. All misclassified entity pairs in Λ are then moved to Γ (Line 4). After this operation, we obtain a set Γ that is consistent with the set Λ, although it contains fewer source domain entity pairs, thereby decreasing the imbalance between source and target relation entity pairs. Note that the sampling process is conducted once for each target relation type.

After selecting a subset of entity pairs from each of the source relation types, we train a multi-class classifier for the source and target relation types. Specifically, we represent each entity pair by a feature vector where we use lexical and syntactic patterns that co-occur with that entity pair as features. For example, given an entity pair (A,B), we use the co-occurrence frequencies $f(\rho, A, B)$ for each pattern ρ as features. Feature vectors are normalized to unit length (L_2 norm) so that we can equally represent both entity pairs that co-occur a lot as well as entity pairs that co-occur only a few times. Each feature vector is assigned a relational label according to the relation that exist between the two entities in the entity pair. We then train a multi-class classifier to learn a classification model to classify each of the source relation types and the target relation type. In general, any classification algorithm can be used to train from the feature vectors. For simplicity, we use multi-class logistic regression as our classifier[2]. The L_2 regularization parameter is set to its default value of 1 and is not tuned for any of our experiments.

2. http://www.chokkan.org/software/classias/

3 Experiments

3.1 Dataset

To evaluate the proposed method, we select 20 relation types that have been used frequently for evaluating relation extraction systems [2], [16], [33] from the YAGO ontology[3]. YAGO (Yet Another Great Ontology) is a large semantic knowledge base that includes over two million entities such as persons, organizations and cities. Moreover, it contains over twenty million facts about those entities. YAGO is automatically created from Wikipedia and uses WordNet to structure information. The YAGO ontology has a high level (on average 95%) of manually confirmed accuracy, which makes it a suitable gold standard for evaluating relations between entity pairs on the Web [34]. For each selected relation, we randomly selected 100 entity pairs listed for that relation in the YAGO ontology. Overall, the dataset contains 2000 (20 relations $\times$ 100 instances) entity pairs. Some of those relation types are: actedIn (actor-movie), ceoOf (ceo-company), acquiredBy (company-company), and directed (director-movie).

The dataset contains various relations that exist between entities of numerous types on the Web. We use the Yahoo BOSS search API[4] to download contexts for the entity pairs in the dataset. Specifically, we construct numerous contextual queries that include the two entities in an entity pair and download snippets that contain those entities using the method proposed in [8]. On average, we have ca. 7000 snippets for a pair of entities in the dataset. The dataset and the source code for the proposed method is publicly available[5].

3.2 Experimental Settings

For each relation type $\mathcal{R}$, we randomly allocated its 100 instances (entity pairs) into three groups: 60 instances as training instances when $\mathcal{R}$ is a source relation, 10 instances as training instances when $\mathcal{R}$ is a target relation, and 30 instances as test instances for $\mathcal{R}$. For each target relation type, therefore we have 1140 (19×60) source relation training instances and 10 target relation training instances, which well simulates the problem setting in relation adaptation. We call this the *train dataset* for a particular target relation type, and the 30 instances set aside for that target relation type combined with the 30 instances set aside from each of the source relation types (19×30) as the *test dataset* for that target relation type.

For each target relation, we use the pattern extraction algorithm presented in Section 2.3 and extract lexical patterns from all the contexts from its train dataset. However, because of misspellings and fragmented snippets, patterns extracted from Web texts can be noisy. To remove noisy patterns, we select those patterns which occur at least 5 times in the dataset. We then use

3. http://www.mpi-inf.mpg.de/yago-naga/yago/
4. http://developer.yahoo.com/search/boss/
5. www.iba.t.u-tokyo.ac.jp/~danushka/RA/

TABLE 3
Macro-average results for various methods.

Method	Precision	Recall	F-score
Random	7.25	7.33	7.24
RS patterns	77.08	29.99	41.41
RI patterns	81.38	40.22	51.40
All patterns	79.34	37.11	47.94
Projected	78.48	33.56	44.86
Combined (all + projected)	84.21	45.11	56.99
RS patterns + Sampling	83.58	38.44	49.78
RI patterns + Sampling	75.60	45.11	54.83
All patterns + Sampling	80.94	47.11	57.62
Projected + Sampling	72.07	37.33	47.61
Jiang [35]	81.06	44.89	55.62
PROPOSED (Combined + Sampling)	**86.47**	**51.78**	**62.77**
Full Graph	79.37	37.77	49.18
Supervised Upper Bound	74.58	75.55	74.57

the entropy-based relation-independent pattern selection criterion and select the top 1000 ranked patterns as relation-independent patterns ($l = 1000$). The remaining patterns are selected as relation-specific patterns. Next, we construct a bipartite graph following the procedure described in Section 2.5 and apply Algorithm 1 on the created bipartite graph to compute feature vector projections. We set the number of clusters to $k = 1000$ in our experiments. Later in Section 3.4, we investigate the effect of the parameters l and k on the proposed method.

To evaluate the performance of a relation adaptation method, we select one relation type in the dataset as a target relation and train a multi-class classifier as described in Section 2. We compute precision, recall, and F-score on the selected target relation type $\mathcal{T}$ as follows:

$$\begin{aligned}\text{precision} &= \frac{\text{no. of correctly classified entity pairs}}{\text{total no. of entity pairs classified as } \mathcal{T}},\\ \text{recall} &= \frac{\text{no. of correctly classified entity pairs}}{\text{total no. of entity pairs in } \mathcal{T}},\\ \text{F} &= \frac{2 \times \text{precision} \times \text{recall}}{\text{precision} + \text{recall}}.\end{aligned}$$

This process is repeated with a different relation type as the target relation and the remaining relation types as the source relations. We report the macro-average scores over the 20 relation types in our benchmark dataset. Note that the macro-averages computed here consider only the target relations and not source relations, because in relation adaptation the objective is to obtain high performance on a target relation and not on source relations.

3.3 Overall Results of Comparisons

Table 3 presents the results obtained using the proposed method, 12 baselines, and a previously proposed weakly-supervised relation extraction system on our dataset as described next.

- **Random**: This baseline randomly infers a relation for an entity pair out of the 20 relation types in the dataset. The probability of randomly guessing the correct relation is as low as 0.05. Consequently, the macro-average F-measure for this baseline is the lowest in Table 3. This method can be considered as a lower baseline.
- **RS patterns**: We use the frequencies of relation specific (RS) patterns as features to represent an entity pair to train a multi-class classifier.
- **RI patterns**: We use the frequencies of relation independent (RI) patterns as features to represent an entity pair to train a multi-class classifier.
- **All patterns**: We use all (relation-specific and relation-independent) patterns and represent an entity-pair by the frequency of the patterns with the entity pair. This baseline is expected to show the level of performance we would obtain if we did not use the lower-dimensional mapping (Algorithm 1) or the one-sided under-sampling (Algorithm 2).
- **Projected**: We use the entropy-based relation-independent pattern selection criterion and construct a bipartite graph as described in Section 2.5. We then run Algorithm 1 to compute a projection matrix $\mathbf{U}$ and project a feature vector $\boldsymbol{x}_{AB}$ for an entity pair (A, B) to a lower-dimensional vector $\mathbf{U}\boldsymbol{x}_{AB}$. We use the projected feature vectors to train a multi-class classifier. This baseline is expected to show the level of performance we would have obtained if we had used only the lower-dimensional representation.
- **Combined**: We use both the original feature vectors as well as their projection into the k-dimensional latent space according to Equation 5 and train a multi-class classifier using those augmented feature vectors. This baseline is expected to demonstrate the level of performance that we would obtain if we did not perform the one-sided under-sampling (Algorithm 2).
- We perform one-sided undersampling to select a subset of source relation type entity pairs. For **RS patterns**, **RI patterns**, **All patterns**, and **Projected** baselines we denote their sampling enabled versions respectively by **RS patterns + Sampling**, **RI patterns + Sampling**, **All patterns + Sampling**, and **Projected + Sampling**.
- **Jiang** [35]: This is the current state-of-the-art cross-domain relation classification method [35]. In this method, first, an entity-pair is represented as a set of lexical and syntactic features. Second, a multi-class logistic regression model is trained using those feature vectors. Some features are shared across different relations and the weight parameters for those features are learned in a joint fashion. This method is further detailed in Section 4. We ran the original implementation that is publicly available[6] on our dataset.
- **PROPOSED**: This is the method proposed in this paper. This corresponds to **Combined + Sampling**.
- **Full Graph**: As an alternative to using a bipartite graph model, we construct a full graph that does not distinguish between relation-independent patterns and relation-specific patterns. All remaining processing steps in the proposed method are conducted exactly the same

6. www.mysmu.edu/faculty/jingjiang/software/DALR.html

way on this full graph. This baseline is expected to demonstrate the importance of identifying relation independent patterns and using a bipartite graph for relation adaptation.

- **Supervised Upper Bound**: The supervised upper bound corresponds to the level of performance that we would obtain, if we had access to not only a few seeds (i.e. 10 training seed instances) for the target relation, but an equal amount of training instances as we have for each source relation type (i.e. 60 training instances). The proposed method (combined + sampling) is run using this training dataset to simulate a fully-supervised relation extraction scenario. Note that this is *not* the relation adaptation scenario that we consider in this paper, where we assume only a few seed training instances for the target relation type.

From Table 3, we see that the proposed method has the best macro-average precision, recall, and F-measure among all the different methods, except for the supervised upper bound. In particular, the improvement against the previously proposed state-of-the-art weakly-supervised relation extraction method [35] is statistically significant (paired t-test with $p < 0.05$ inferred as significant). The **Random** baseline on this balanced dataset only yields a very low F-score of 7.25. The **RI patterns** baseline that uses only relation-independent patterns outperforms the **RS patterns** baseline that uses only relation-specific patterns (9.99% improvement in F-score). This result is particularly interesting considering that there exist only 1000 relation-independent, whereas on average there exist 67,822 relation-specific patterns. Even with a few relation-independent patterns, we can learn a better relational adaptation model than using many relation-specific patterns. Relation-specific patterns occur in only a few relation types. Therefore, a model trained using those patterns alone does not generalize well to a novel target relation type. Moreover, on average, for all target relation types, among the 1000 relation-independent patterns, we have 454 lexical patterns and 546 syntactic patterns, whereas among the 67,822 relation-specific patterns we have 65,771 lexical patterns and 2051 syntactic patterns. Considering the fact that part-of-speech tags abstract individual words, it is not surprising that a major proportion of the relation-independent patterns are indeed syntactic patterns.

Using all the patterns (i.e. **All patterns** baseline) performs slightly worse than when using only relation-independent patterns (3.46% drop in F-score). One reason for this is that the overall performance of the **All patterns** baseline is dominated by the numerous relation-specific patterns, which adapt poorly to a target relation. As explained already in Section 2.7, there can be errors in identifying relation-independent patterns using strategies such as mutual information, which engender some noise in the constructed bipartite graph. Consequently, using only the **Projected** features is not satisfactory (6.54% drop in F-score compared to **RI patterns**). However, by augmenting the original features to the projected features (i.e. **Combined** baseline), this problem can be overcome (5.59% improvement of F-score over **RI patterns**).

Next, we evaluate the effect of the one-sided undersampling method presented in Section 2.7 on top of the numerous baselines discussed above. From the experimental results presented in Table 3, it is apparent that, by sampling, we consistently improve all the baselines: **RS patterns** (8.37% increase in F-score), **RI patterns** (3.43% increase in F-score), **All patterns** (9.68% increase in F-score), and **Projected** (2.75% increase in F-score). In fact, the proposed method, which uses augmented feature vectors with sampling, shows a 5.78% improvement in F-score over not using sampling (i.e. **Combined**). A paired two-tailed t-test shows that the improvements in all measures due to sampling is statistically significant under the 0.05 significance level. This underscores the importance of selecting a subset of source relation instances when training a classifier for a target relation. In relation adaptation, the number of source relation labeled instances significantly outperforms that for a target relation. Without proper sampling, any information related to the target relation, encoded in the small number of target relation instances, get "washed-out" during training.

The **Full Graph** baseline has low performance (F-score of 49.18%), which is roughly equal to the **All patterns** baseline in terms of performance. This result shows that it is important to use a bipartite graph structure as done by the proposed method for relation adaptation instead of using a full graph that does not distinguish between relation-specific and relation-independent patterns. From a domain adaptation point-of-view, it is the existence of common features (i.e. relation-independent patterns in our case) that enables us to transfer the weights learnt from the source relation types to the target relation type. The bipartite graph structure enforces this constraint explicitly, thereby yielding a lower-dimensional latent space that helps towards relation adaptation.

It is noteworthy that the **Supervised Upper Bound** reports the highest F-score of 74.57 in Table 3. From this result we can see that there is still sufficient room for improvement for the proposed method. However, it is encouraging to observe that the **proposed** method attains an F-score of 62.77 merely using one sixth (10 vs 60) training instances for the target relation as used by the **Supervised Upper Bound**.

A future research direction in relation adaptation is how to improve recall without loosing precision. In fact, for most of the methods compared in Table 3, we see that the macro-averaged recall is much lower than the macro-averaged precision. This behavior can be understood considering the fact that we have only a few seed training instances for the target relation type in relation adaptation. Therefore, the classifier only learns a limited set of properties regarding the target relation

TABLE 4
Effect of relation-independent pattern selection criteria.

Method	Precision	Recall	F-measure
Frequency	77.49	44.83	54.72
MI Pan et al. [19]	74.29	46.83	55.39
Entropy (**PROPOSED**)	86.47	51.78	62.77

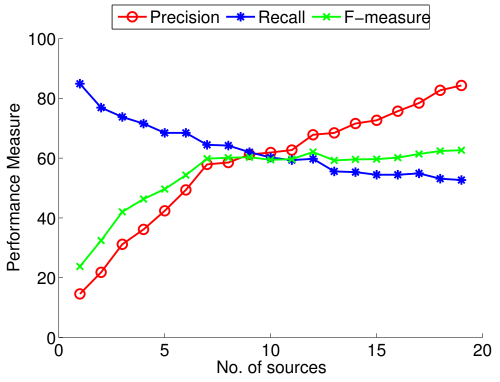

Fig. 3. Effect of varying the number of sources

and is unable generalize beyond this small set of training instances. Consequently, it cannot recognize most of the target relations in the test set, resulting in low recall. On the other hand, it will predict any test instance as belonging to the target relation if that test instance is highly similar to any of the training target instances, resulting in high precision.

In Table 4, we compare the three strategies presented in Section 2.4 to select a set of relation-independent patterns. With each of the three strategies, we select 1000 relation-independent patterns and created a bipartite graph as described in Section 2.5. Next, we use Algorithm 1 with dimensionality $k = 1000$ to augment the feature vectors to train a multi-class classifier with sampling as described in Section 2.7. Table 4 shows that the proposed entropy-based relation-independent pattern selection method outperforms both frequency-based and mutual information (MI)-based approaches. The frequency-based method tends to ignore low-frequent relation-independent patterns, whereas the MI-based approach sometimes select patterns that has a high mutual information with only a subset of relation types. Entropy-based relation-independent pattern selection addresses the entire distribution of a pattern over entity pairs and is less affected by the disfluencies described above.

For each target relation in our dataset, we use multiple source relation types with the proposed method and evaluate the effect on performance as shown in Figure 3. We see that as we increase the number of source relation types, the macro-average F-score improves steadily up to a certain point (ca. 8 sources) and then saturates. This result shows that we can improve the performance on a target relation up to a certain level simply by using multiple source relations.

Figure 4 depicts the performance of the proposed method as a function of the number of training instances

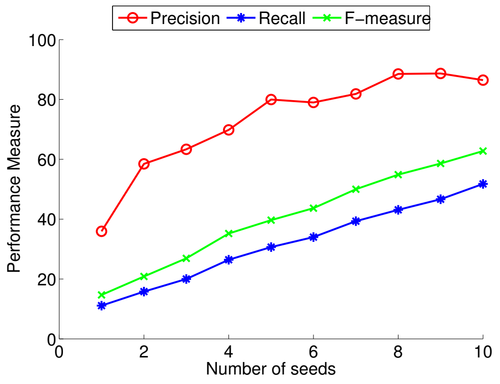

Fig. 4. Effect of the no. of target training instances.

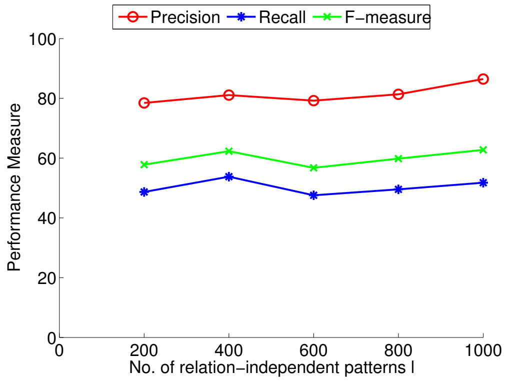

Fig. 5. Effect of varying the number of relation-independent patterns l.

for the target relation. The figure shows that the performance increases steadily with the number of training instances we have for the target relation. This result emphasizes the importance of target relation instances for relation adaptation, and justifies our decision to retain all target relation instances during sampling.

3.4 Parameter Sensitivity

We experimentally study the effect of the two parameters in our proposed method: the number of relation-independent patterns l, and the dimensionality k of the latent space. To study the effect of the number of relation-independent patterns l on the proposed method, we use the entropy-based strategy to select different quantities of relation-independent patterns and use those patterns in the proposed method. The dimensionality k of the latent space is fixed at 1000. In Figure 5, we present the performance of the proposed method against the number of relation-independent patterns used. We see that the proposed method is stable against the varying quantities of relation-independent patterns.

Next, we study the effect of varying the dimensionality k of the latent space by executing Algorithm 1 with different k values. The set of relation-independent features is fixed throughout this experiment (i.e. the total no. relation-independent patterns $l = 1000$). Figure 6 shows the performance of the proposed method against the value of k. From Figure 6, it is apparent that the

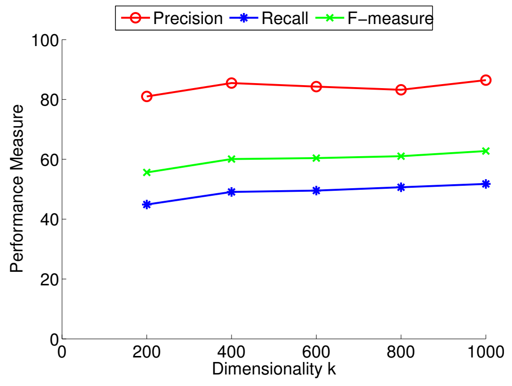

Fig. 6. Effect of varying the number of dimensions

performance of the proposed method remains almost constant across the range of k values.

4 Related Work and Discussion

The relation adaptation method proposed in this paper is motivated by work in three fields: relation extraction, domain adaptation, and transfer learning. Next, we discuss previous work in those fields and compare it to our proposed method.

Traditionally, relation extraction is framed as a binary classification problem: Given a sentence S and a relation $\mathcal{R}$, does S assert $\mathcal{R}$ between two entities in S? Kernel-based supervised methods such as dependency tree kernels [5], subsequence kernels [36], and convolution tree kernels [37] have been successfully used to learn relation extraction systems. In particular, kernel methods allow the use of a large set of features without the need to extract them explicitly. However, supervised relation extraction methods assume the availability of sufficient labeled training data, which is problematic when we want to extract new relation types, as we do in this paper.

Bootstrapping methods [9], [33], [38]–[40] 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 designated as seeds) of the target relation [9], [33], [39] or general extraction templates [38]. During subsequent iterations of the bootstrapping process, new extraction patterns are discovered and are used to extract new instances. However, the quality of the extracted relations depends heavily upon the initial seeds provided to the bootstrapping system [41]. Different from bootstrapping, we not only use target relation seeds, but also use the existing training instances for numerous source relations to train a robust relation extractor for a target relation. In information extraction, lexical patterns have been used for extracting class instances [40] and entity pairs with specific relations [33]. However, unlike those works that use vector space model-based patterns, the subsequence patterns used in our work can both preserve the relative ordering of words as well as consider long range dependencies.

Open Information Extraction (Open IE) [2], [7], [42] is a domain-independent information extraction paradigm that has been studied in both a natural language document corpus [42], and the Web environment [2], [7] 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. In contrast, we learn the domain-independent relation patterns using source and target relation instances. We do not require them to be provided manually. Moreover, open IE systems attempt to extract all relations that exist in a corpus; users cannot specify in advance which relation types (targets) they want to extract. Therefore, it is not guaranteed that we will be able to extract instances for the target relation type in which we are interested.

Domain adaptation methods can be classified broadly into fully supervised [43], [44] and semi-supervised adaptation [18], [19], [45], [46] . In the fully supervised scenario, we have labeled data for the source domain and also invest in labeling a few instances in the target domain whereas, the semi-supervised version does not assume the availability of labeled data from the target domain to use unlabeled data from the target domain. Domain adaptation methods first identify a set of common features in source and target domains and then use those features as pivots to map source domain features to the target domain. However, relation adaptation differs from domain adaptation because, in domain adaptation, it is assumed that the class labels remain the *same* in both source and target domains, only the distribution of data is different whereas, in relation adaptation, the source and target relation types are considered to be *different*.

Transfer learning is intended to transfer knowledge learned from one or more tasks to a new task. In the Alternating Structure Optimization (ASO) [47] framework, a learning algorithm is first trained on a set of *auxiliary* problems. The linear prediction vectors for those problems are arranged as a matrix. Next, Singular Value Decomposition is performed on this matrix to compute a lower-dimensional mapping between the features. The working hypothesis in ASO is that by jointly learning a set of related problems (auxiliary problems), we can learn some useful information related to the structure of the data, which is useful when learning a new task. Relation adaptation can be seen as a special instance of transfer learning, where the source relations act as auxiliary problems. We must *transfer* the structural knowledge about source relations to a target relation type. However, relation adaptation necessitates that we overcome the additional challenge of learning the target relation using only a few seed instances.

Spectral clustering of bipartite graphs have been studied in [23] and in this case it turns out to be producing co-clusters. If we represent the vertices of one partite of the bipartite graph in rows of a matrix and the vertices of the other partite in columns (the matrix M in Section 2.5 is such an example), then spectral clustering of the original bipartite graph gives co-clusters for the newly formed matrix. If we translate this result back to our scenario where we have relation specific patterns in one of the partites and relation independent patterns in the other partite of the bipartite graph, then we retrieve clusters which group relation specific and relation independent patterns that are semantically similar. The co-clusters can be considered as providing an alignment between relation specific and relation independent patterns. This enables us to perform relation adaptation because relation specific patterns (features) in both source relations and the target relation can be first mapped to relation independent patterns and then train a classifier in this common (lower-dimensional) feature space.

Methods proposed for learning from an unbalanced dataset can be broadly categorized into two: *undersampling* methods and *oversampling* methods. Undersampling methods, such as the method described in Algorithm 2, attempt to select a subset of training instances from the majority class, whereas oversampling methods synthetically generate instances from the minority class. For example, DataBoost-IM method [48] first detects hard-to-classify examples for each class and then separately generates synthetic instances for each class. More synthetic instances are generated for the minority class. Next, the frequency of classes in the combined (original + synthetic) dataset are re-balanced to alleviate the learning algorithm's bias towards the majority class. Finally, the total weights of instances of different classes are re-balanced so that the final classifier will focus on hard as well as minority class examples. Above procedure is iterated for a user specified number of iterations unless the error term exceeds some threshold value.

Synthetic Minority Oversampling Technique (SMOTE) [49] is an oversampling method that generates new minority class instances by interpolating between existing minority class instances. For example, with 200% sampling rate, for each minority class instance, two of its neighbors are selected uniformly at random from its neighborhood. Next, the difference vector between the selected two neighbors is multiplied by a random number between 0 and 1 and is added to the original minority class instance to generate a new minority class instance. One problem of oversampling methods is that the synthetically generated instances might not bring in new information to the training dataset. Moreover, it can lead to overfitting to the minority class. On the other hand, the 1-nearest neighbor-based one-sided undersampling method used in this paper does not generate any synthetic instances and has also shown to be robust to the noise in the training data [30].

The most computationally demanding steps in our proposed method are the bipartite graph construction and eigenvalue decomposition of the graph Laplacian. It is noteworthy that the bipartite graph that we construct contains lexical-syntactic patterns as its vertices. Therefore, the size of the affinity matrix of the bipartite graph (hence, the size of the Laplacian) directly depends on the number of patterns we extract. In practice, the computational complexity of eigenvalue decomposition of an $n \times n$ square matrix is $O(n^3)$. Therefore, there is a cubic time complexity dependence on the number of patterns we extract. However, it must be emphasized that the size of the affinity matrix does not depend on the number of relation types nor the number of entity pairs. For example, if we use a fixed set of patterns to represent all entity pairs, then the complexity of this step will remain constant regardless of the number of relation types or the entity pairs used in the system. We consider this is to be a desirable property of the proposed method that makes it scalable to thousands of relations and/or many source examples. Moreover, the stochastic matrix decomposition algorithm [29] that we use in our proposed method produces an approximate truncated eigenvalue decomposition in quadratic ($O(n^2)$) time complexity, which is sufficiently accurate for our task.

5 Conclusion

We proposed and investigated a method to learn a relational classifier for a target relation using multiple source relations. Our experimental results show that the proposed method significantly outperforms 11 baselines and a previously proposed weakly-supervised relation extraction method on a dataset that contains 2000 entity pairs for 20 different relation types. Both feature projection and sampling positively contribute to the proposed method. Moreover, the proposed method performs consistently under different parameter settings. An interesting future research direction of relation adaptation is to extend the current method to handle entities that are not related as well as entities with multiple semantic relations. Moreover, in our future work we intend to apply the proposed relation adaptation method in real-world relation extraction systems and evaluate its effectiveness in detecting novel relation types.

References

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

[2] M. Banko, M. Cafarella, S. Soderland, M. Broadhead, and O. Etzioni, "Open information extraction from the web," in *IJCAI'07*, 2007, pp. 2670–2676.

[3] 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.

[4] R. Bunescu and R. Mooney, "A shortest path dependency kernel for relation extraction," in *Proc. of EMNLP'05*, 2005, pp. 724 – 731.

[5] A. Culotta and J. Sorensen, "Dependency tree kernels for relation extraction," in *Proc. of ACL'04*, 2004, pp. 423–429.

[6] Z. GuoDong, S. Jian, Z. Jie, and Z. Min, "Exploring various knowledge in relation extraction," in *ACL'05*, 2005, pp. 427 – 434.

[7] M. Banko and O. Etzioni, "The tradeoffs between traditional and open relation extraction," in *ACL'08*, 2008, pp. 28–36.

[8] D. Bollegala, Y. Matsuo, and M. Ishizuka, “Measuring the similarity between implicit semantic relations from the web,” in *WWW’09*, 2009, pp. 651 – 660.
[9] J. Zhu, Z. Nie, X. Liu, B. Zhang, and J. R. Wen, “Statsnowball: a statistical approach to extracting entity relationships,” in *WWW’09*. ACM, 2009, pp. 101–110.
[10] M. Baroni and A. Kilgarriff, “Large linguistically-processed web corpora for multiple languages,” in *EACL’06*, 2006, pp. 87 – 90.
[11] M. Hearst, “Automatic acquisition of hyponyms from large text corpora,” in *COLING’92*, 1992, pp. 539–545.
[12] R. Snow, D. Jurafsky, and A. Ng, “Learning syntactic patterns for automatic hypernym discovery,” in *NIPS’05*, 2005, pp. 1297–1304.
[13] M. Berland and E. Charniak, “Finding parts in very large corpora,” in *ACL’99*, 1999, pp. 57–64.
[14] D. Ravichandran and E. Hovy, “Learning surface text patterns for a question answering system,” in *ACL’02*, 2001, pp. 41–47.
[15] R. Bhagat and D. Ravichandran, “Large scale acquisition of paraphrases for learning surface patterns,” in *ACL’08*, 2008, pp. 674–682.
[16] D. Bollegala, Y. Matsuo, and M. Ishizuka, “Relational duality: Unsupervised extraction of semantic relations between entities on the web,” in *WWW’10*, 2010, pp. 151 – 160.
[17] 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*, vol. 16, no. 11, pp. 1424–1440, 2004.
[18] J. Blitzer, M. Dredze, and F. Pereira, “Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification,” in *ACL’07*, 2007, pp. 440–447.
[19] S. J. Pan, X. Ni, J.-T. Sun, Q. Yang, and Z. Chen, “Cross-domain sentiment classification via spectral feature alignment,” in *WWW’10*, 2010.
[20] Z. Harris, “Distributional structure,” *Word*, vol. 10, pp. 146–162, 1954.
[21] D. Lin and P. Pantel, “Dirt: Discovery of inference rules from text,” in *SIGKDD’01*, 2001, pp. 323–328.
[22] F. R. K. Chung, *Spectral Graph Theory*. American Mathematical Society, 1997, vol. CBMS Regional Conference Series in Mathemaitcs Number 92.
[23] I. Dhillion, “Co-clustering documents and words using bipartite spectral graph partitioning,” in *KDD’01*, 2001, pp. 269–274.
[24] M. Belkin and P. Niyogi, “Laplacian eigenmaps for dimensionality reduction and data representation,” *Neural Computation*, vol. 15, no. 6, pp. 1373 – 1396, 2003.
[25] S. Ben-David, J. Blitzer, K. Crammer, and F. Pereira, “Analysis of representations for domain adaptation,” in *NIPS’06*, 2006.
[26] J. Shi and J. Malik, “Normalized cuts and image segmentation,” *IEEE TPAMI*, vol. 22, no. 8, pp. 888 – 905, 2000.
[27] X. Wang and I. Davidson, “Flexible constrained spectral clustering,” in *KDD’10*, 2010, pp. 563 – 572.
[28] C. Ding and X. He, “K-means clustering via principal component analysis,” in *ICML’04*, 2004, pp. 225 – 232.
[29] N. Halko, P. G. Martinsson, and J. A. Tropp, “Finding structure with randomness: stochastic algorithms for constructing approximate matrix decompositions,” California Institute of Technology, Tech. Rep., 2009.
[30] I. Tomek, “Two modifications of cnn,” *IEEE Transactions on System, Man and Cybernetics*, vol. 6, no. 11, pp. 769 – 772, 1976.
[31] M. Kubat and S. Matwin, “Addressing the curse of imbalanced training sets: one-sided selection,” in *ICML’97*, 1997, pp. 179 – 186.
[32] F. Provost, “Machine learning from imbalanced data sets,” in *AAAI’00 Workshop on Imbalanced Data Sets*, 2000.
[33] E. Agichtein and L. Gravano, “Snowball: Extracting relations from large plain-text collections,” in *ICDL’00*, 2000.
[34] F. M. Suchanek, G. Kasneci, and G. Weikum, “Yago: A core of semantic knowledge,” in *WWW’07*, 2007.
[35] J. Jiang, “Multi-task transfer learning for weakly-supervised relation extraction,” in *ACL’09*, 2009, pp. 1012–1020.
[36] R. C. Bunescu and R. J. Mooney, “Subsequence kernels for relation extraction,” in *NIPS’05*, 2005.
[37] L. Qian, G. Zhou, F. Kong, Q. Zhu, and P. Qian, “Exploiting constitutent dependencies for tree kernel-based semantic relation extraction,” in *COLING’08*, 2008, pp. 697 – 704.
[38] O. Etzioni, M. Cafarella, D. Downey, A.-M. Popescu, T. Shaked, S. Soderland, D. S. Weld, and A. Yates, “Unsupervised named-entity extraction from the web: An experimental study,” *Artificial Intelligence*, vol. 165, no. 1, pp. 91–134, 2005.
[39] 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 *Proc. of AAAI’06*, 2006, pp. 1400–1405.
[40] E. Riloff and R. Jones, “Learning dictionaries for information extraction by multi-level bootstrapping,” in *AAAI’99*, 1999, pp. 474 – 479.
[41] Z. Kozareva and E. Hovy, “Not all seeds are equal: Measuring the quality of text mining seeds,” in *NAACL’10*, 2010.
[42] Y. Shinyama and S. Sekine, “Preemptive information extraction using unrestricted relation discovery,” in *NACCL’06*, 2006.
[43] H. Daumé III, “Frustratingly easy domain adaptation,” in *ACL’07*, 2007, pp. 256–263.
[44] J. Jiang and C. Zhai, “Instance weighting for domain adaptation in nlp,” in *ACL’07*, 2007, pp. 264 – 271.
[45] J. Blitzer, R. McDonald, and F. Pereira, “Domain adaptation with structural correspondence learning,” in *EMNLP’06*, 2006.
[46] J. Jiang and C. Zhai, “A two-stage approach to domain adaptation for statistical classifiers,” in *CIKM’07*, 2007, pp. 401–410.
[47] R. K. Ando and T. Zhang, “A framework for learning predictive structures from multiple tasks and unlabeled data,” *Journal of Machine Learning Research*, vol. 6, pp. 1817–1853, 2005.
[48] H. Guo and H. L. Viktor, “Learning from imbalanced data sets with boosting and data generation: the databoost-im approach,” *SIGKDD Newsletters*, vol. 6, pp. 30 – 39, 2004.
[49] N. V. Chawla, K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer, “Smote: Synthetic minority over-sampling technique,” *Journal of Artificial Intelligence Research*, vol. 16, pp. 321 – 357, 2002.

Danushka Bollegala received the BS, MS and PhD degrees from the University of Tokyo, Japan in 2005, 2007, and 2009. He is an assistant professor at Graduate School of Information Science and Technology, the University of Tokyo, Japan. His research interests include natural language processing, Web mining and artificial intelligence. He is a member of IEEE and ACL.

Yutaka Matsuo is an associate professor at Institute of Engineering Innovation, the University of Tokyo, Japan. He received the BS, MS, and PhD degrees from the University of Tokyo in 1997, 1999, and 2002. He was at National Institute of Advanced Industrial Science and Technology (AIST) from 2002 to 2007. He is interested in social network mining, text processing, and semantic web in the context of artificial intelligence.

Mitsuru Ishizuka received the BS and PhD degrees in electronic engineering from the University of Tokyo, in 1971 and 1976, respectively. He is a professor at Graduate School of Information Science and Technology, University of Tokyo, Japan. Prior to this, he worked at NTT Yokosuka Lab., Univ. of Tokyo’s Institute of Industrial Science, and Purdue Univ. His research interests include artificial intelligence, Web intelligence, and lifelike agents. He is a member of the IEEE and a past president of JSAI (Japanese Soc. for Artificial Intelligence).