Classification of Semantic Relationships between Nominals Using Pattern Clusters

Dmitry Davidov
ICNC
Hebrew University of Jerusalem
dmitry@alice.nc.huji.ac.il

Ari Rappoport
Institute of Computer Science
Hebrew University of Jerusalem
arir@cs.huji.ac.il

Abstract

There are many possible different semantic relationships between nominals. Classification of such relationships is an important and difficult task (for example, the well known noun compound classification task is a special case of this problem). We propose a novel *pattern clusters* method for nominal relationship (NR) classification. Pattern clusters are discovered in a large corpus independently of any particular training set, in an unsupervised manner. Each of the extracted clusters corresponds to some unspecified semantic relationship. The pattern clusters are then used to construct features for training and classification of specific inter-nominal relationships. Our NR classification evaluation strictly follows the ACL SemEval-07 Task 4 datasets and protocol, obtaining an f-score of 70.6, as opposed to 64.8 of the best previous work that did not use the manually provided WordNet sense disambiguation tags.

1 Introduction

Automatic extraction and classification of semantic relationships is a major field of activity, of both practical and theoretical interest. A prominent type of semantic relationships is that holding between nominals[1]. For example, in noun compounds many different semantic relationships are encoded by the same simple form (Girju et al., 2005): ‘dog food’ denotes food consumed by dogs, while ‘summer morning’ denotes a morning that happens in the summer. These two relationships are completely different semantically but are similar syntactically, and distinguishing between them could be essential for NLP applications such as question answering and machine translation.

[1]Our use of the term ‘nominal’ follows (Girju et al., 2007), and includes simple nouns, noun compounds and multiword expressions serving as nouns.

Relation classification usually relies on a training set in the form of tagged data. To improve results, some systems utilize additional manually constructed semantic resources such as WordNet (WN) (Beamer et al., 2007). However, in many domains and languages such resources are not available. Furthermore, usage of such resources frequently requires disambiguation and connection of the data to the resource (word sense disambiguation in the case of WordNet). Manual disambiguation is unfeasible in many practical tasks, and an automatic one may introduce errors and greatly degrade performance. It thus makes sense to try to minimize the usage of such resources, and utilize only corpus contexts in which the relevant words appear.

A leading method for utilizing context information for classification and extraction of relationships is that of patterns (Hearst, 1992; Pantel and Pennacchiotti, 2006). The standard classification process is to find in an auxiliary corpus a set of patterns in which a given training word pair co-appears, and use pattern-word pair co-appearance statistics as features for machine learning algorithms.

In this paper we introduce a novel approach, based on utilizing pattern clusters that are prepared separately and independently of the training set. We do not utilize any manually constructed resource or any manual tagging of training data beyond the correct classification, thus making our method applicable to fully automated tasks and less domain and language dependent. Moreover, our pattern clustering algorithm is fully unsupervised.

Our method is based on the observation that while each lexical pattern can be highly ambiguous, several patterns in conjunction can reliably define and represent a lexical relationship. Accordingly, we construct pattern clusters from a large generic corpus, each such cluster potentially representing some important generic relationship. This step is done without accessing any training data, anticipating that most meaningful relationships, including those in a given classification problem, will be represented by some of the discovered clusters. We then use the training set to label some of the clusters, and the labeled clusters to assign classes to tested items. One of the advantages of our method is that it can be used not only for classification, but also for further analysis and retrieval of the observed relationships[2].

The semantic relationships between the components of noun compounds and between nominals in general are not easy to categorize rigorously. Several different relationship hierarchies have been proposed (Nastase and Szpakowicz, 2003; Moldovan et al., 2004). Some classes, like Container-Contained, Time-Event and Product-Producer, appear in several classification schemes, while classes like Tool-Object are more vaguely defined and are subdivided differently. Recently, SemEval-07 Task 4 (Girju et al., 2007) proposed a benchmark dataset that includes a subset of 7 widely accepted nominal relationship (NR) classes, allowing consistent evaluation of different NR classification algorithms. In the SemEval event, 14 research teams evaluated their algorithms using this benchmark. Some of the teams have used the manually annotated WN labels provided with the dataset, and some have not.

We evaluated our algorithm on SemEval-07 Task 4 data, showing superior results over participating algorithms that did not utilize WordNet disambiguation tags. We also show how pattern clusters can be used for a completely unsupervised classification of the test set. Since in this case no training data is used, this allows the automated discovery of a potentially unbiased classification scheme.

[2]In (Davidov and Rappoport, 2008) we focus on the pattern cluster resource type itself, presenting an evaluation of its intrinsic quality based on SAT tests. In the present paper we focus on showing how the resource can be used to improve a known NLP task.

Section 2 discusses related work, Section 3 outlines the pattern clustering algorithm, Section 4 details three classification methods, and Sections 5 and 6 describe the evaluation protocol and results.

2 Related Work

Numerous methods have been devised for classification of semantic relationships, among which those holding between nominals constitute a prominent category. Major differences between these methods include available resources, degree of preprocessing, features used, classification algorithm and the nature of training/test data.

2.1 Available Resources

Many relation classification algorithms utilize WordNet. Among the 15 systems presented by the 14 SemEval teams, some utilized the manually provided WordNet tags for the dataset pairs (e.g., (Beamer et al., 2007)). In all cases, usage of WN tags improves the results significantly. Some other systems that avoided using the labels used WN as a supporting resource for their algorithms (Costello, 2007; Nakov and Hearst, 2007; Kim and Baldwin, 2007). Only three avoided WN altogether (Hendrickx et al., 2007; Bedmar et al., 2007; Aramaki et al., 2006).

Other resources used for relationship discovery include Wikipedia (Strube and Ponzetto, 2006), thesauri or synonym sets (Turney, 2005) and domain-specific semantic hierarchies like MeSH (Rosario and Hearst, 2001).

While usage of these resources is beneficial in many cases, high quality word sense annotation is not easily available. Besides, lexical resources are not available for many languages, and their coverage is limited even for English when applied to some restricted domains. In this paper we do not use any manually annotated resources apart from the classification training set.

2.2 Degree of Preprocessing

Many relationship classification methods utilize some language-dependent preprocessing, like deep or shallow parsing, part of speech tagging and named entity annotation (Pantel et al., 2004). While the obtained features were shown to improve classification performance, they tend to be language dependent and error-prone when working on unusual text domains and are also highly computationally intensive when processing large corpora. To make our approach as language independent and efficient as possible, we avoided using any such preprocessing techniques.

2.3 Classification Features

A wide variety of features are used by different algorithms, ranging from simple bag-of-words frequencies to WordNet-based features (Moldovan et al., 2004). Several studies utilize syntactic features. Many other works manually develop a set of heuristic features devised with some specific relationship in mind, like a WordNet-based meronymy feature (Bedmar et al., 2007) or size-of feature (Aramaki et al., 2006). However, the most prominent feature type is based on lexico-syntactic patterns in which the related words co-appear.

Since (Hearst, 1992), numerous works have used patterns for discovery and identification of instances of semantic relationships (e.g., (Girju et al., 2006; Snow et al., 2006; Banko et al, 2007)). Rosenfeld and Feldman (2007) discover relationship instances by clustering entities appearing in similar contexts. Strategies were developed for discovery of multiple patterns for some specified lexical relationship (Pantel and Pennacchiotti, 2006) and for unsupervised pattern ranking (Turney, 2006). Davidov et al. (2007) use pattern clusters to define general relationships, but these are specific to a given concept. No study so far has proposed a method to define, discover and represent general relationships present in an arbitrary corpus.

In (Davidov and Rappoport, 2008) we present an approach to extract pattern clusters from an untagged corpus. Each such cluster represents some unspecified lexical relationship. In this paper, we use these pattern clusters as the (only) source of machine learning features for a nominal relationship classification problem. Unlike the majority of current studies, we avoid using any other features that require some language-specific information or are devised for specific relationship types.

2.4 Classification Algorithm

Various learning algorithms have been used for relation classification. Common choices include variations of SVM (Girju et al., 2004; Nastase et al., 2006), decision trees and memory-based learners. Freely available tools like Weka (Witten and Frank, 1999) allow easy experimentation with common learning algorithms (Hendrickx et al., 2007). In this paper we did not focus on a single ML algorithm, letting algorithm selection be automatically based on cross-validation results on the training set, as in (Hendrickx et al., 2007) but using more algorithms and allowing a more flexible parameter choice.

2.5 Training Data

As stated above, several categorization schemes for nominals have been proposed. Nastase and Szpakowicz (2003) proposed a two-level hierarchy with 5 (30) classes at the top (bottom) levels[3]. This hierarchy and a corresponding dataset were used in (Turney, 2005; Turney, 2006) and (Nastase et al., 2006) for evaluation of their algorithms. Moldovan et al. (2004) proposed a different scheme with 35 classes. The most recent dataset has been developed for SemEval 07 Task 4 (Girju et al., 2007). This manually annotated dataset includes a representative rather than exhaustive list of 7 important nominal relationships. We have used this dataset, strictly following the evaluation protocol. This made it possible to meaningfully compare our method to state-of-the-art methods for relation classification.

3 Pattern Clustering Algorithm

Our pattern clustering algorithm is designed for the unsupervised definition and discovery of generic semantic relationships. The algorithm first discovers and clusters patterns in which a single ('hook') word participates, and then merges the resulting clusters to form the final structure. In (Davidov and Rappoport, 2008) we describe the algorithm at length, discuss its behavior and parameters in detail, and evaluate its intrinsic quality. To assist readers of the present paper, in this section we provide an overview. Examples of some resulting pattern clusters are given in Section 6. We refer to a pattern

[3]Actually, there were 50 relationships at the bottom level, but valid nominal instances were found only for 30.

contained in our clusters (a pattern type) as a 'pattern' and to an occurrence of a pattern in the corpus (a pattern token) as a 'pattern instance'.

The algorithm does not rely on any data from the classification training set, hence we do not need to repeat its execution for different classification problems. To calibrate its parameters, we ran it a few times with varied parameters settings, producing several different configurations of pattern clusters with different degrees of noise, coverage and granularity. We then chose the best configuration for our task automatically without re-running pattern clustering for each specific problem (see Section 5.3).

3.1 Hook Words and Hook Corpora

As a first step, we randomly sample a set of hook words, which will be used in order to discover relationships that generally occur in the corpus. To avoid selection of ambiguous words or typos, we do not select words with frequency higher than a parameter F_C and lower than a threshold F_B. We also limit the total number N of hook words. For each hook word, we now create a *hook corpus*, the set of the contexts in which the word appears. Each context is a window containing W words or punctuation characters before and after the hook word.

3.2 Pattern Specification

To specify patterns, following (Davidov and Rappoport, 2006) we classify words into high-frequency words (HFWs) and content words (CWs). A word whose frequency is more (less) than F_H (F_C) is considered to be a HFW (CW). Our patterns have the general form

[Prefix] CW_1 **[Infix]** CW_2 **[Postfix]**

where Prefix, Infix and Postfix contain only HFWs. We require Prefix and Postfix to be a single HFW, while Infix can contain any number of HFWs (limiting pattern length by window size). This form may include patterns like '*such X as Y and*'. At this stage, the pattern slots can contain only single words; however, when using the final pattern clusters for nominal relationship classification, slots can contain multiword nominals.

3.3 Discovery of Target Words

For each of the hook corpora, we now extract all pattern instances where one CW slot contains the hook word and the other CW slot contains some other ('target') word. To avoid the selection of common words as target words, and to avoid targets appearing in pattern instances that are relatively fixed multiword expressions, we sort all target words in a given hook corpus by pointwise mutual information between hook and target, and drop patterns obtained from pattern instances containing the lowest and highest L percent of target words.

3.4 Pattern Clustering

We now have for each hook corpus a set of patterns, together with the target words used for their extraction, and we want to cluster pattern types. First, we group in clusters all patterns extracted using the same target word. Second, we merge clusters that share more than S percent of their patterns. Some patterns can appear in more than a single cluster. Finally, we merge pattern clusters from different hook corpora, to avoid clusters specific to a single hook word. During merging, we define and utilize *core patterns* and *unconfirmed patterns*, which are weighed differently during cluster labeling (see Section 4.2). We merge clusters from different hook corpora using the following algorithm:

```
1. Remove all patterns originating from a single hook corpus only.
2. Mark all patterns of all present clusters as unconfirmed.
3. While there exists some cluster $C_1$ from corpus $D_X$ containing only unconfirmed patterns:
   (a) Select a cluster with a minimal number of patterns.
   (b) For each corpus $D$ different from $D_X$:
       i. Scan $D$ for clusters $C_2$ that share at least $S$ percent of their patterns, and all of their core patterns, with $C_1$.
       ii. Add all patterns of $C_2$ to $C_1$, setting all shared patterns as core and all others as unconfirmed.
       iii. Remove cluster $C_2$.
   (c) If all of $C_1$'s patterns remain unconfirmed remove $C_1$.
4. If several clusters have the same set of core patterns merge them according to rules (i,ii).
```

At the end of this stage, we have a set of pattern clusters where for each cluster there are two subsets, core patterns and unconfirmed patterns.

4 Relationship Classification

Up to this stage we did not access the training set in any way and we did not use the fact that the target relations are those holding between nominals. Hence, only a small part of the acquired pattern clusters may be relevant for a given NR classification task, while other clusters can represent completely different relationships (e.g., between verbs). We now use the acquired clusters to learn a model for the given labeled training set and to use this model for classification of the test set. First we describe how we deal with data sparseness. Then we propose a HITS measure used for cluster labeling, and finally we present three different classification methods that utilize pattern clusters.

4.1 Enrichment of Provided Data

Our classification algorithm is based on contexts of given nominal pairs. Co-appearance of nominal pairs can be very rare (in fact, some word pairs in the Task 4 set co-appear only once in Yahoo web search). Hence we need more contexts where the given nominals or nominals similar to them co-appear. This step does not require the training labels (the correct classifications), so we do it for both training and test pairs. We do it in two stages: extracting similar nominals, and obtaining more contexts.

4.1.1 Extracting more words

For each nominal pair (w_1, w_2) in a given sentence S, we use a method similar to (Davidov and Rappoport, 2006) to extract words that have a shared meaning with w_1 or w_2. We discover such words by scanning our corpora and querying the web for symmetric patterns (obtained automatically from the corpus as in (Davidov and Rappoport, 2006)) that contain w_1 or w_2. To avoid getting instances of $w_{1,2}$ with a different meaning, we also require that the second word will appear in the same text paragraph or the same web page. For example, if we are given a pair <*loans*, *students*> and we see a sentence '*... loans and scholarships for students and professionals ...*', we use the symmetric pattern 'X and Y' to add the word *scholarships* to the group of *loans* and to add the word *professionals* to the group of *students*. We do not take words from the sentence '*In European soccer there are transfers and loans...*' since its context does not contain the word *students*. In cases where there are only several or zero instances where the two nominals co-appear, we dismiss the latter rule and scan for each nominal separately. Note that 'loans' can also be a verb, so usage of a part-of-speech tagger might reduce noise.

If the number of instances for a desired nominal is very low, our algorithm trims the first words in these nominal and repeats the search (e.g., <*simulation study*, *voluminous results*> becomes <*study*, *results*>). This step is the only one specific to English, using the nature of English noun compounds. Our desire in this case is to keep the head words.

4.1.2 Extracting more contexts using the new words

To find more instances where nominals similar to w_1 and w_2 co-appear in HFW patterns, we construct web queries using combinations of each nominal's group and extract patterns from the search result snapshots (the two line summary provided by search engines for each search result).

4.2 The HITS Measure

To use clusters for classification we define a HITS measure similar to that of (Davidov et al., 2007), reflecting the affinity of a given nominal pair to a given cluster. We use the pattern clusters from Section 3 and the additional data collected during the enrichment phase to estimate a HITS value for each cluster and each pair in the training and test sets. For a given nominal pair (w_1, w_2) and cluster C with n core patterns P_{core} and m unconfirmed patterns P_{unconf},

$$\text{HITS}(C, (w_1, w_2)) = \\ |\{p; (w_1, w_2) \text{ appears in } p \in P_{core}\}| / n + \\ \alpha \times |\{p; (w_1, w_2) \text{ appears in } p \in P_{unconf}\}| / m.$$

In this formula, 'appears in' means that the nominal pair appears in instances of this pattern extracted from the original corpus or retrieved from the web at the previous stage. Thus if some pair appears in most of the patterns of some cluster it receives a high HITS value for this cluster. α (0..1) is a parameter that lets us modify the relative weight of core and unconfirmed patterns.

4.3 Classification Using Pattern Clusters

We present three ways to use pattern clusters for relationship classification.

4.3.1 Classification by cluster labeling

One way to train a classifier in our case is to attach a single relationship label to each cluster during the training phase, and to assign each unlabeled pair to some labeled cluster during the test phase. We use the following normalized HITS measure to label the involved pattern clusters. Denote by k_i the number of training pairs in class i in training set T. Then

$$Label(C) = argmax_i \sum_{p \in T, Label(p)=i} hits(C,p)/k_i$$

Clusters where the above sum is zero remain unlabeled. In the test phase we assign to each test pair p the label of the labeled cluster C that received the highest HITS(C,p) value. If there are several clusters with a highest HITS value, then the algorithm selects a 'clarifying' set of patterns – patterns that are different in these best clusters. Then it constructs clarifying web queries that contain the test nominal pair inside the clarifying patterns. The effect is to increment the HITS value of the cluster containing a clarifying pattern if an appropriate pattern instance (including the target nominals) was found on the web. We start with the most frequent clarifying pattern and perform additional queries until no clarifying patterns are left or until some labeled cluster obtains a highest HITS value. If no patterns are left but there are still several winning clusters, we assign to the pair the label of the cluster with the largest number of pattern instances in the corpus.

One advantage of this method is that we get as a by-product a set of labeled pattern clusters. Examination of this set can help to distinguish and analyze (by means of patterns) which different relationships actually exist for each class in the training set. Furthermore, labeled pattern clusters can be used for web queries to obtain additional examples of the same relationship.

4.3.2 Classification by cluster HITS values as features

In this method we treat the HITS measure for a cluster as a feature for a machine learning classification algorithm. To do this, we construct feature vectors from each training pair, where each feature is the HITS measure corresponding to a single pattern cluster. We prepare test vectors similarly. Once we have feature vectors, we can use a variety of classifiers (we used those in Weka) to construct a model and to evaluate it on the test set.

4.3.3 Unsupervised clustering

If we are not given any training set, it is still possible to separate between different relationship types by grouping the feature vectors of Section 4.3.2 into clusters. This can be done by applying k-means or another clustering algorithm to the feature vectors described above. This makes the whole approach completely unsupervised. However, it does not provide any inherent labeling, making an evaluation difficult.

5 Experimental Setup

The main problem in a fair evaluation of NR classification is that there is no widely accepted list of possible relationships between nominals. In our evaluation we have selected the setup and data from SemEval-07 Task 4 (Girju et al., 2007). Selecting this type of dataset allowed us to compare to 6 submitted state-of-art systems that evaluated on exactly the same data and to 9 other systems that utilize additional information (WN labels). We have applied our three different classification methods on the given data set.

5.1 SemEval-07 Task 4 Overview

Task 4 (Girju et al., 2007) involves classification of relationships between simple nominals other than named entities. Seven distinct relationships were chosen: Cause-Effect, Instrument-Agency, Product-Producer, Origin-Entity, Theme-Tool, Part-Whole, and Content-Container. For each relationship, the provided dataset consists of 140 training and 70 test examples. Examples were binary tagged as belonging/not belonging to the tested relationship. The vast majority of negative examples were near-misses, acquired from the web using the same lexico-syntactic patterns as the positives. Examples appear as sentences with the nominal pair tagged. Nouns in this pair were manually labeled with their corresponding WordNet 3 labels and the web queries used to obtain the sentences. The 15 submitted systems were assigned into 4 categories according to whether they use the WordNet and Query tags (some systems were assigned to more than a single category, since they reported experiments in several settings). In our evaluation we do not utilize WordNet or Query tags, hence we compare ourselves with the corresponding group (A), containing 6 systems.

5.2 Corpus and Web Access

Our algorithm uses two corpora. We estimate frequencies and perform primary search on a local web corpus containing about 68GB untagged plain text. This corpus was extracted from the web starting from open directory links, comprising English web pages with varied topics and styles (Gabrilovich and Markovitch, 2005). To enrich the set of given word pairs and patterns as described in Section 4.1 and to perform clarifying queries, we utilize the Yahoo API for web queries. For each query, if the desired words/patterns were found in a page link's snapshot, we do not use the link, otherwise we download the page from the retrieved link and then extract the required data. If only several links were found for a given word pair we perform local crawling to depth 3 in an attempt to discover more instances.

5.3 Parameters and Learning Algorithm

Our algorithm utilizes several parameters. Instead of calibrating them manually, we only provided a desired range for each, and the final parameter values were obtained during selection of the best-performing setup using 10-fold cross-validation on the training set. For each parameter we have estimated its desired range using the (Nastase and Szpakowicz, 2003) set as a development set. Note that this set uses an entirely different relationship classification scheme. We ran the pattern clustering phase on 128 different sets of parameters, obtaining 128 different clustering schemes with varied granularity, noise and coverage.

The parameter ranges obtained are: F_C (meta-pattern content word frequency and upper bound for hook word selection): $100 - 5000$ words per million (wpm); F_H (meta-pattern HFW): $10 - 100$ wpm; F_B (low word count for hook word filtering): $1 - 50$ wpm; N (number of hook words): $100 - 1000$; W (window size): 5 or window = sentence; L (target word mutual information filter): $1/3 - 1/5$; S (cluster overlap filter for cluster merging): $2/3$; α (core vs. unconfirmed weight for HITS estimation): $0.1 - 0.01$; S (commonality for cluster merging): $2/3$. As designed, each parameter indeed influences a certain effect. Naturally, the parameters are not mutually independent. Selecting the best configuration in the cross-validation phase makes the algorithm flexible and less dependent on hard-coded parameter values.

Selection of learning algorithm and its algorithm-specific parameters were done as follows. For each of the 7 classification tasks (one per relationship type), for each of the 128 pattern clustering schemes, we prepared a list of most of the compatible algorithms available in Weka, and we automatically selected the model (a parameter set and an algorithm) which gave the best 10-fold cross-validation results. The winning algorithms were LWL (Atkeson et al., 1997), SMO (Platt, 1999), and K* (Cleary and Trigg, 1995) (there were 7 tasks, and different algorithms could be selected for each task). We then used the obtained model to classify the testing set. This allowed us to avoid fixing parameters that are best for a specific dataset but not for others. Since each dataset has only 140 examples, the computation time of each learning algorithm is negligible.

6 Results

The pattern clustering phase results in 90 to 3000 distinct pattern clusters, depending on the parameter setup. Manual sampling of these clusters indeed reveals that many clusters contain patterns specific to some apparent lexical relationship. For example, we have discovered such clusters as: $\{$*'buy Y accessory for X!', 'shipping Y for X', 'Y is available for X', 'Y are available for X', 'Y are available for X systems', 'Y for X'* $\}$ and $\{$*'best X for Y', 'X types for Y', 'Y with X', 'X is required for Y', 'X as required for Y', 'X for Y'*$\}$. Note that some patterns (*'Y for X'*) can appear in many clusters.

We applied the three classification methods described in Section 4.3 to Task 4 data. For supervised classification we strictly followed the SemEval datasets and rules. For unsupervised classification we did not use any training data. Using the k-means algorithm, we obtained two nearly equal unlabeled

Method	**P**	**R**	**F**	**Acc**
Unsupervised clustering (4.3.3)	64.5	61.3	62.0	64.5
Cluster Labeling (4.3.1)	65.1	69.0	67.2	68.5
HITS Features (4.3.2)	**69.1**	**70.6**	**70.6**	**70.1**
Best Task 4 (no WordNet)	66.1	66.7	64.8	66.0
Best Task 4 (with WordNet)	79.7	69.8	72.4	76.3

Table 1: Our SemEval-07 Task 4 results.

Relation Type	**F**	**Acc**	**C**
Cause-Effect	69.7	71.4	2
Instrument-Agency	76.5	74.2	1
Product-Producer	76.4	83.8	1
Origin-Entity	65.4	62.6	4
Theme-Tool	59.4	58.7	6
Part-Whole	74.3	70.9	1
Content-Container	72.6	69.2	2

Table 2: By-relation Task 4 HITS-based results. **C** is the number of clusters with positive labels.

clusters containing test samples. For evaluation we assigned a negative/positive label to these two clusters according to the best alignment with true labels.

Table 1 shows our results, along with the best Task 4 result not using WordNet labels (Costello, 2007). For reference, the best results overall (Beamer et al., 2007) are also shown. The table shows precision (P) recall (R), F-score (F), and Accuracy (Acc) (percentage of correctly classified examples).

We can see that while our algorithm is not as good as the best method that utilizes WordNet tags, results are superior to all participants who did not use these tags. We can also see that the unsupervised method results are above the random baseline (50%). In fact, our results (f-score 62.0, accuracy 64.5) are better than the averaged results (58.0, 61.1) of the group that did not utilize WN tags.

Table 2 shows the HITS-based classification results (F-score and Accuracy) and the number of positively labeled clusters (C) for each relation. As observed by participants of Task 4, we can see that different sets vary greatly in difficulty. However, we also obtain a nice insight as to why this happens – relations like Theme-Tool seem very ambiguous and are mapped to several clusters, while relations like Product-Producer seem to be well-defined by the obtained pattern clusters.

The SemEval dataset does not explicitly mark items whose correct classification requires analysis of the context of the whole sentence in which they appear. Since our algorithm does not utilize test sentence contextual information, we do not expect it to show exceptional performance on such items. This is a good topic for future research.

Since the SemEval dataset is of a very specific nature, we have also applied our classification framework to the (Nastase and Szpakowicz, 2003) dataset, which contains 600 pairs labeled with 5 main relationship types. We have used the exact evaluation procedure described in (Turney, 2006), achieving a class f-score average of 60.1, as opposed to 54.6 in (Turney, 2005) and 51.2 in (Nastase et al., 2006). This shows that our method produces superior results for rather differing datasets.

7 Conclusion

Relationship classification is known to improve many practical tasks, e.g., textual entailment (Tatu and Moldovan, 2005). We have presented a novel framework for relationship classification, based on pattern clusters prepared as a standalone resource independently of the training set.

Our method outperforms current state-of-the-art algorithms that do not utilize WordNet tags on Task 4 of SemEval-07. In practical situations, it would not be feasible to provide a large amount of such sense disambiguation tags manually. Our method also shows competitive performance compared to the majority of task participants that do utilize WN tags. Our method can produce labeled pattern clusters, which can be potentially useful for automatic discovery of additional instances for a given relationship. We intend to pursue this promising direction in future work.

Acknowledgement. We would like to thank the anonymous reviewers, whose comments have greatly improved the quality of this paper.

References

Aramaki, E., Imai, T., Miyo, K., and Ohe, K., 2007. UTH: semantic relation classification using physical sizes. *ACL SemEval '07 Workshop.*

Atkeson, C., Moore, A., and Schaal, S., 1997. Locally weighted learning. *Artificial Intelligence Review*, 11(1–5): 75–113.

Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M., and Etzioni, O., 2007. Open information extraction from the Web. *IJCAI '07.*

Beamer, B., Bhat, S., Chee, B., Fister, A., Rozovskaya A., and Girju, R., 2007. UIUC: A knowledge-rich approach to identifying semantic relations between nominals. *ACL SemEval '07 Workshop.*

Bedmar, I. S., Samy, D., and Martinez, J. L., 2007. UC3M: Classification of semantic relations between nominals using sequential minimal optimization. *ACL SemEval '07 Workshop.*

Cleary, J. G. , Trigg, L. E., 1995. K*: An instance-based learner using and entropic distance measure. *ICML '95.*

Costello, F. J., 2007. UCD-FC: Deducing semantic relations using WordNet senses that occur frequently in a database of noun-noun compounds. *ACL SemEval '07 Workshop.*

Davidov, D., Rappoport, A., 2006. Efficient unsupervised discovery of word categories using symmetric patterns and high frequency words. *COLING-ACL '06*

Davidov D., Rappoport A. and Koppel M., 2007. Fully unsupervised discovery of concept-specific relationships by Web mining. *ACL '07.*

Davidov, D., Rappoport, A., 2008. Unsupervised discovery of generic relationships using pattern clusters and its evaluation by automatically generated SAT analogy questions. *ACL '08.*

Gabrilovich, E., Markovitch, S., 2005. Feature generation for text categorization using world knowledge. *IJCAI '05.*

Girju, R., Giuglea, A., Olteanu, M., Fortu, O., Bolohan, O., and Moldovan, D., 2004. Support vector machines applied to the classification of semantic relations in nominalized noun phrases. *HLT/NAACL '04 Workshop on Computational Lexical Semantics.*

Girju, R., Moldovan, D., Tatu, M., and Antohe, D., 2005. On the semantics of noun compounds. *Computer Speech and Language,* 19(4):479-496.

Girju, R., Badulescu, A., and Moldovan, D., 2006. Automatic discovery of part-whole relations. *Computational Linguistics*, 32(1).

Girju, R., Hearst, M., Nakov, P., Nastase, V., Szpakowicz, S., Turney, P., and Yuret, D., 2007. Task 04: Classification of semantic relations between nominal at SemEval 2007. *4th Intl. Workshop on Semantic Evaluations (SemEval '07), in ACL '07.*

Hearst, M., 1992. Automatic acquisition of hyponyms from large text corpora. *COLING '92*

Hendrickx, I., Morante, R., Sporleder, C., and van den Bosch, A., 2007. Machine learning of semantic relations with shallow features and almost no data. *ACL SemEval '07 Workshop.*

Kim, S.N., Baldwin, T., 2007. MELB-KB: Nominal classification as noun compound interpretation. *ACL SemEval '07 Workshop.*

Moldovan, D., Badulescu, A., Tatu, M., Antohe, D., and Girju, R., 2004. Models for the semantic classification of noun phrases. *HLT-NAACL '04 Workshop on Computational Lexical Semantics.*

Nakov, P., and Hearst, M., 2007. UCB: System description for SemEval Task #4. *ACL SemEval '07 Workshop.*

Nastase, V., Szpakowicz, S., 2003. Exploring noun-modifier semantic relations. *In Fifth Intl. Workshop on Computational Semantics (IWCS-5).*

Nastase, V., Sayyad-Shirabad, J., Sokolova, M., and Szpakowicz, S., 2006. Learning noun-modifier semantic relations with corpus-based and WordNet-based features. *In Proceedings of the 21st National Conference on Artificial Intelligence, Boston, MA.*

Pantel, P., Ravichandran, D., and Hovy, E., 2004. Towards terascale knowledge acquisition. *COLING '04.*

Pantel, P., Pennacchiotti, M., 2006. Espresso: leveraging generic patterns for automatically harvesting semantic relations. *COLING-ACL '06.*

Platt, J., 1999. Fast training of support vector machines using sequential minimal optimization. In *Scholkopf, Burges, and Smola, Advances in Kernel Methods – Support Vector Learning,* pp. 185–208. MIT Press.

Rosario, B., Hearst, M., 2001. Classifying the semantic relations in noun compounds. *EMNLP '01.*

Rosenfeld, B., Feldman, R., 2007. Clustering for unsupervised relation identification. *CIKM '07.*

Snow, R., Jurafsky, D., Ng, A.Y., 2006. Semantic taxonomy induction from heterogeneous evidence. *COLING-ACL '06.*

Strube, M., Ponzetto, S., 2006. WikiRelate! computing semantic relatedness using Wikipedia. *AAAI '06.*

Tatu, M., Moldovan, D., 2005. A semantic approach to recognizing textual entailment. HLT/EMNLP '05.

Turney, P., 2005. Measuring semantic similarity by latent relational analysis. *IJCAI '05.*

Turney, P., 2006. Expressing implicit semantic relations without supervision. *COLING-ACL '06.*

Witten, H., Frank, E., 1999. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations.* Morgan Kaufman, San Francisco, CA.