Recent approaches for classification of semantic relations are based on supervised learning using large training datasets. Due to the high cost of annotating such data and to the class imbalance problem, alternatives for minimizing the effort of full corpus annotation are required. In set expansion, one of such alternatives, given a small initial training set, new relevant instances are acquired from a large corpus. However, when dealing with contextual semantic relations, which are relations that are highly dependent on the context within the sentence, set expansion is not trivial, since instances are not directly queryable and filtering requires classification under a very restricted number of training instances. This work thus proposes a bootstrapped set expansion method for contextual semantic relations. It performs a best effort extraction using the Web, and a two-stage filtering of candidate instances, the first based on syntactic patterns and the second using a feature distance-based classifier designed for the low frequency setting. The relevance of the output is measured experimentally by using the expanded set as the training data of the supervised classification task, observing an incremental improvement in performance after each bootstrapping iteration when compared to values using the unexpanded training data.

**Keywords:** Semantic relations; set expansion; contextual semantics.

### 1. Introduction

Informally defining contextual semantic relations as links among conceptual entities which structure the meaning of a natural language text, the extraction and classification of such relations from natural language texts are of growing interest in academia, observed in tasks such as semantic role labeling [1–3] and discourse parsing [4]. Current approaches are based on supervised learning, which rely on large annotated corpora as training data. However, full corpus annotation is very resource intensive and costly.
consuming, requiring training of human annotators with linguistic background and an extensive annotation effort. The annotated corpora are also susceptible to error and bias, which may be partially addressed by applying intra- and inter-annotator consistency checking, further increasing the overall cost of the process. This scenario is aggravated by the fact that for a given classification scheme, some relation classes may present few instances even in large annotated corpora due to the class imbalance problem [5]. As a result, it is not unrealistic to consider that some classes will present very few instances in the training data for the semantic relation classification task. This produces a training feature space that generates an incomplete classification model, which may in turn compromise the results of the classification task since the model will not be able to recognize many of the features in the testing data.

This work thus proposes using set expansion as an alternative to full corpus annotation for the semantic relation classification task for contextual semantic relations. Given a seed dataset which consists of very few instances for each relation class, set expansion extracts new instances that are relevant to the seed from a large unannotated corpus such as the Web. This results in a new training set for the classification task whose coverage of the feature space is increased by adding instances with new or recombined features. This substantial increase is expected to improve overall classification, as observed in the 2005 CoNLL task for semantic role labeling [2].

Nevertheless, it should be noticed that although the ultimate objective of this work is to avoid the classifier of the original relation classification task dealing with small training data, the set expansion process itself requires a classifier whose training data produces an incomplete feature space. This is addressed by proposing a feature distance-based classifier, which does not penalize the classifier when there are features inexistent in the training set. For the set expansion, the loss of expressibility in the features when using distance metrics is compensated by the gain in classification, as observed in experimental results, and the controlled input solves restrictions imposed by the use of a classifier based in such metrics.

This paper is structured as follows. In Sec. 2, background on semantic relations is given. In Sec. 3, the method is proposed, elucidating the overall architecture of the bootstrapped process, and the extraction and filtering steps of the set expansion. In Sec. 4, experiments that evaluate the performance of the method are presented and some discussions are carried. Finally, in Sec. 5, the conclusion is given.

2. Background on Semantic Relations

Semantic relations are generally defined as meaningful associations among conceptual entities. Considering their nature, they can be categorized into two large groups: lexical semantic relations and contextual semantic relations.

Lexical semantic relations are subject of the field of Lexical Semantics and handle relations regarding the lexical properties of words. They are described in lexicons [6], and are the objective of ontology creation [7] and extraction of relations between
nominal entities [8]. Contextual semantic relations, on the other hand, are subject of Contextual Semantics and are concerned with the structure of texts, being roughly equivalent to syntagmatic relations [9, 10] and relations at higher levels of text [11]. They have schemes defined for tasks involving role labeling [12–14] and discourse relation parsing [15, 16]. Examples for both categories of semantic relations are given in Table 1.

These contextual semantic relations can be described by a head entity, a tail entity and the description of the context \( c \) which connects these two entities, i.e. the role that each of the head and tail entities have in the text. For different contexts, relations can be grouped in classes \( R_c \) according to a pre-defined scheme. A natural language text can be abstracted by these contextual relations through a directed hypergraph. This hypergraph \( H = (E, R) \) is defined by the set of hyperedges \( E \) and the set of vertices instances \( R \), which correspond respectively to the set of entities and relations. In order to accommodate phrases, clauses and sentences, an entity can be any connected subgraph of \( H \).

One example of contextual semantic relations is the Concept Description Language (CDL) [17, 18], a relation classification scheme that proposes an interlingual representation of natural language text for semantically describing media. An example is given in Fig. 1, in which a sentence is annotated using CDL and is represented by its hypergraph.

<table>
<thead>
<tr>
<th>Category</th>
<th>Subcategory</th>
<th>Examples</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lexical</td>
<td>Prototypical</td>
<td>Maltese → Dog (hypernym)</td>
</tr>
<tr>
<td></td>
<td>General</td>
<td>Tokyo → Japan (“capital of”)</td>
</tr>
<tr>
<td>Contextual</td>
<td>Within an event</td>
<td>went → Tokyo in “I went to Tokyo” (destination)</td>
</tr>
<tr>
<td></td>
<td>Between events</td>
<td>“If it rains, then I will not go” (conditional)</td>
</tr>
</tbody>
</table>

Table 1. Types of semantic relations.

Fig. 1. Hypergraph representation of a CDL annotation for the sentence “John reported to Alice that he bought a computer yesterday”.
an experimental setting of 13,487 instances evaluated using ten-fold cross validation. In this high-frequency setting, reported results were satisfactory, achieving 86.89% of $F$-value. For small training sets, on the other hand, our past work in [20] is extended herein.

Nevertheless, for low-frequency settings, [21] used set expansion for lexical semantic relations, and [22] used feature vector extension for discourse relations. While the first used properties specific of lexical semantic relations for the expansion, the latter work is based on the premise that features are somehow correlated, and this correlation information is used for the feature extension, assumption which cannot be made for this work. This work proposes using a classifier based on feature distances in order to account for features that are not present in the training data.

3. Proposed Method

The proposed method uses set expansion for contextual semantic relations, so that the feature space of the training data for a supervised algorithm is improved by adding new relevant instances. The overall architecture is illustrated in Fig. 2.

For each relation class $R_c$ in the dataset, sentences possibly containing new relation instances are extracted from the Web. The resulting candidate instances are filtered using syntactic patterns, and then classified using a feature distance-based classifier as belonging or not to the relation class $R_c$. This is done by comparing the confidence of the classification to a pre-defined threshold. Only the confidently classified instances are then added to the dataset, and this process is repeated in a bootstrapped manner [23]. The inclusion of a manual checking step is discussed in Sec. 4.

The following subsections give more details on how contextual semantic relations are modeled (Sec. 3.1), as well as each of the extraction (Sec. 3.2) and filtering (Sec. 3.3) steps of the set expansion process.

Fig. 2. Architecture of the bootstrapped set expansion.
3.1. Modeling

Recent works on the classification task of contextual semantic relations use feature vectors for describing linguistic properties of the relations [19]. The used features include morphological, syntactic and lexico-semantic features. For this work, the extracted features, together with their extraction software, are described below. The part-of-speech tag, named entity tag and WordNet sense are extracted for head and tail entities, resulting in eight feature types overall.

- Part-of-speech (POS) tag\textsuperscript{a}: Morphological feature that indicates word class
- Phrase structure tree shortest path\textsuperscript{a}: Shortest path between two entities in the phrase structure tree
- Dependency tree shortest path\textsuperscript{a}: Shortest path between two entities in the dependency tree
- Named entity (NE) tag\textsuperscript{a}: Lexical feature that provides labels for proper nouns, classifying them as people, places or institutions
- WordNet (WN) sense\textsuperscript{b}: Lexical feature that provides the sense of a word

The syntax of a sentence can be represented either by the phrase structure, which is composed by rules that describe a language’s syntax, breaking a sentence into constituent parts, or by dependencies, which indicates the syntactic functions of words. These two representations of syntax can be used as features by determining the shortest path between two entities of the tree [24], as illustrated in Fig. 3.

![Phrase structure and dependency as semantic relation features.](image)

\textsuperscript{a}Stanford CoreNLP: http://nlp.stanford.edu/software/corenlp.shtml.
\textsuperscript{b}WordNet: http://wordnet.princeton.edu.
When using feature vectors, each of the aforementioned feature type corresponds to several elements within the feature vector, ultimately producing sparsity. For example, the POS tag of the head entity produces 36 vector elements (one for each tag defined in the Penn Treebank [25]), but only one of them will have a non-zero value at a time.

Under the constraint of small datasets, this sparsity leads to an incomplete feature space, since the classifier will not be able to recognize many of the features in the testing data. Considering a very simplistic example, the CDL Agent relation is characterized by the association between a verb and its subject in the active voice. If the training dataset contains only verbs in the base form (VB) and no verbs in the past tense (VBD), any VBD might be incorrectly penalized by the classifier.

In order to address this problem, feature distance measures are used. Feature distance is a function \( \delta^k \) that given any two features of a given type \( k \), outputs a value in the range between 0 and 1, according to the similarity between these features. By using such metrics, it is guaranteed that a feature will have a value, regardless if it is present or not in the training set. For the previously presented feature types, the following distance measures are used:

- **POS tag**: Pre-defined distances among POS tags
- **Phrase structure tree shortest path**: Levenshtein distance
- **Dependency tree shortest path**: Levenshtein distance
- **NE tag**: Binary
- **WN sense**: Hierarchy tree-based distance

The Levenshtein distance, which is used for feature types based on shortest path, is originally a string distance metric. It was generalized so that it could handle any array, and the final result was normalized using array length. As for the WordNet sense distance, the distance measure is calculated using the distances of each sense \( s_1 \) and \( s_2 \) to their common parent \( cp \) of the WordNet hierarchy tree, and the distance between the common parent to the root \( r \), as expressed in Eq. (1).

\[
\delta_{wn} = \frac{\min(d(s_1, cp), d(s_2, cp))}{d(cp, r) + \min(d(s_1, cp), d(s_2, cp))}.
\] (1)

Table 2 exemplifies the eight feature types for two relation instances, **categorizes** \( \rightarrow \) **products** from the sentence “the new definition categorizes genes by functional products” and **training** \( \rightarrow \) **product** from the sentence “certification training by product”, giving corresponding distance values between the pair of instances for each feature type.

As a result, a set of relation instances can be represented regarding each of the eight feature types by diagonal matrices \( D_k = \delta^k_{ij} \), which represent the distance values between every pair of instances \( i \) and \( j \) from this set of relation instances for a certain feature type \( k \). These matrices are called single-view matrices, since they only
provide partial information concerning the relation set. Examples of single-view matrices are given in Fig. 4, in which distances are represented by greyscale colors, with darker colors being equivalent to distance closer to zero (similar features), and whiter to distance closer to one (different features).

In the example illustrated in the figure, it is expected that a feature type relevant for separating the considered relation classes is one with more darker points along the main diagonal and whiter points outside it. Numerically, this block diagonality indicates that for a given feature type $k$, instances from the same class have lower distances and instances from different classes have higher distances among them, according to the distance metric defined for that feature type. For example, there are five darker clusters in the tail POS tag matrix: three of which along the main diagonal, indicating that relation instances within these clusters are similar, and two clusters outside the main diagonal, indicating that the first and third clusters are not easily separable using only the tail entity POS tag. A simple graphical analysis enables one to conclude that feature types such as tail POS tag and phrase structure are better for classifying elements for this specific relation set than feature types such as NE tags.

It should be observed that this kind of analysis is possible using the numerical and graphical representations of single-view matrices. This is a desirable property of the model, motivated by the fact that for set expansion, the quality of the initial seed is of utmost importance. By analyzing these matrices, it is possible to know how well each feature type separates the given dataset and if the relation instances are easily separable given a feature set, facilitating the evaluation of prototypes of contextual semantic relation classification systems.

### Table 2. Examples of feature types and distance values for the relation instance `categorizes` → `products` and `training` → `product`.

<table>
<thead>
<tr>
<th>Feature type</th>
<th><code>Categorizes</code> → <code>products</code></th>
<th><code>Training</code> → <code>product</code></th>
<th>Distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Head POS tag</td>
<td>VBZ</td>
<td>NN</td>
<td>1.0</td>
</tr>
<tr>
<td>Tail POS tag</td>
<td>NNS</td>
<td>NN</td>
<td>0.3</td>
</tr>
<tr>
<td>Phrase structure</td>
<td>[head] [[VP]] [PP by] [NP] [tail] [head] [NP] [[FRAG]] [PP by] [NP] [tail]</td>
<td>0.5</td>
<td></td>
</tr>
<tr>
<td>Dependency</td>
<td>[head] [prep by] [tail]</td>
<td>[head] [dep] [*] [pobj] [tail]</td>
<td>1.0</td>
</tr>
<tr>
<td>Head NE tag</td>
<td>NONE</td>
<td>NONE</td>
<td>0.0</td>
</tr>
<tr>
<td>Tail NE tag</td>
<td>NONE</td>
<td>NONE</td>
<td>0.0</td>
</tr>
<tr>
<td>Head WN tag</td>
<td><code>categorize</code>#1</td>
<td>training#1, educate#3, discipline#1, etc.</td>
<td>1.0</td>
</tr>
<tr>
<td>Tail WN tag</td>
<td>merchandise#1, product#2, etc.</td>
<td>merchandise#1, product#2, etc.</td>
<td>0.0</td>
</tr>
</tbody>
</table>

3.2. **Extraction**

The extraction step is responsible for constructing Web queries given a relation instance and processing the search result, so that candidate relation instances are
passed forward to the filtering step. For example, the destination relation \textit{travel → Japan} from “\textit{travel to Japan}” generates the following two queries:

\begin{verbatim}
“travel to *”
“* to Japan”
\end{verbatim}

The first query produces results such as \textit{travel → anywhere} from the sentence “\textit{that may not sound like much of a reason to travel to anywhere...}”. This relation, if
properly classified positively as a destination relation, becomes seed for the
next iteration of the bootstrapped process, and may lead to relations such as
move → anywhere and move → Portugal after some iterations.

It should be noticed, however, that the use of a bootstrapping is motivated by the
fact that the expansion for contextual relations is expected to be slower when
compared to lexical relations. For lexical semantic relations [21], the definition of a
relation class can be explicitly expressed in a query, following the ideas that there are
contexts that explicitly state the relation properties [26] and that these contexts can
be used for querying [27]. This results that, in the example of the capital relation,
entity pair queries such as “Tokyo * Japan” are able to extract new contexts “Y,
capital of X”, and context queries such as “* capital of *” are able to extract new
pairs “Paris * France”.

However, for contextual semantic relations, it is not possible to obtain candidate
instances using context queries “destination” or entity pair queries “* to *” to
expand travel → Japan, since the first does not produce relevant results in an
unannotated corpus, and the second produces too much noise. Consequently, by
using only one wildcard in the query at a time, the expansion is expected to be slower,
which enforces the need for bootstrapping.

Another challenge for the extraction of contextual relations is that contextual
semantic relations rely on morphological, syntactic and semantic features, which is
information that cannot be carried by the query. This leads to extraction being
performed on a best-effort basis, and the noise outputted from the extraction step
being filtered later. The filtering step is presented in the next subsection.

3.3. Filtering

The first filtering step, carried in order to remove noise and to decrease the amount of
data to be processed, is based on syntactic patterns. Because the query in the
extraction step is not able to carry syntactic information, which is one of the most
defining characteristic of contextual relations, a filtering based on syntactic patterns
is applied. Two types of patterns, one which use phrase structure and one which use
dependency, are presented herein and compared in the experiments (Sec. 4).

For phrase structure-based patterns, the following information is used for pattern
matching: (1) part-of-speech tags of head and tail entities, (2) phrasal categories of
intermediate nodes, (3) highest node of the tree path, and (4) indication if the head
or tail entity comes first in the sentence. The matching of part-of-speech tags of head
and tail entities is relaxed, so that similar tags are also matched. An example of
pattern, obtained from the phrase structure tree in Fig. 3(a), is as follows:

[head][[VP]][PP by][NP][tail]

For dependency-based patterns, the pattern extraction is more straight-forward,
since prepositions and conjunctions are explicitly stated for collapsed dependencies.
An example of such pattern extracted from Fig. 3(b) is given below. It should be
noticed that because nodes of the dependency tree are words, and not constituent units as is the case of phrase structure, wildcards should be used to replace them.

\[
[[\text{head}]] [\text{prep by}] [\text{tail}]
\]

This syntactic pattern-based filtering step is quite restrictive, and assumes that the small training data is able to describe the possible syntactic structures for the relation class reasonably well.

The second filtering step consists of classifying the remaining instances as belonging or not to a relation class \( R_c \). For set expansion, it is expected that the filtering process prioritizes precision over recall, focusing on removing false positives while extracting true positives. As a result, a classifier with numerical score output is necessary, so that only confidently classified instances are considered for the next iteration of the bootstrapped process.

It should also be noticed that although set expansion prevents that the classifier for the semantic relation classification task \( C_{\text{task}} \) from dealing with training data with few instances, the classifier used within set expansion \( C_{\text{expansion}} \) itself has to work under this low-frequency constraint. However, the requirements for each of the classifiers are different. For \( C_{\text{task}} \), the classification is a multiclass problem with high performance requirements due to the large amount of training data. For \( C_{\text{expansion}} \), on the other hand, the classification has controlled input and is one-vs-all, which can ultimately be reduced to a positive-vs-negative classification. Moreover, due to the expected small amount of training data, performance requirements can be disregarded.

The proposed classifier \( C_{\text{expansion}} \) is as follows. Given the \( n \times n \) single-view feature distance matrices \( D_k \) for each of the \( k \) feature types as presented in Sec. 3.1, a multiview feature distance matrix \( D \) is defined as a linear combination of these single-view matrices:

\[
D = \beta_0 + \sum_{i=1}^{k} \beta_i \cdot D_i. \tag{2}
\]

The constant term \( \beta_0 \) can also be included within the sum by defining the matrix \( D_0 \) as a matrix of ones. As a result, the previous equation can be written as follows, in order to simplify notation:

\[
D = \sum_{i=0}^{k} \beta_i \cdot D_i. \tag{3}
\]

The \( \beta \) coefficients represent the weights of each feature type for the representation of a given relation class \( R_c \). Therefore, defining an expected multiview feature distance matrix \( D' \), the best \( \beta \) values are given by minimizing the quadratic error \((D' - D)^2\):

\[
\frac{\partial}{\partial \beta_x} \left[ D' - \sum_{i=0}^{k} \beta_i D_i \right]^2 = 0, \quad x = 0, \ldots, k. \tag{4}
\]
This results in Eq. (5), which states that the task of minimizing the error in Eq. (4) is equivalent to finding \( \beta \) coefficients for the system of equations below.

\[
D' = \sum_{i=0}^{k} \beta_i \cdot D_i.
\] (5)

The values of \( \beta \) can be easily achieved using least squares multiple regression algorithms for the expected matrix \( D' \). Although such algorithms use single vector decomposition (SVD), which has a complexity of \( O(n(k + 1)^2) \), this complexity is not an issue since the number of feature types \( k \) is small and fixed. The training thus has a linear complexity regarding the size \( n \) of the training data.

As for the expected matrix \( D' \), given that the training data is composed of positive instances (instances that belong to the positive class \( R_c \)) and negative instances (instances that do not belong to \( R_c \)), the value of each element \( D'_{ij} \) is equals to 0 if instances corresponding to \( i \) and \( j \) are positive and 1 otherwise. Figure 5(a) illustrates the resulting expected matrix \( D' \), with black lines separating positive from negative instances.

It should be noticed that a training element is only relevant for classification if (1) it is above the main diagonal, since the feature distance matrices are symmetric, and (2) if relation instances in positions \( i \) and \( j \) are not both negative, since the classifier should focus on separating positive relations in order to achieve maximum precision. Figure 5(b) shows elements that are relevant for the multiple regression calculation.

For the single-view matrices given as example in Fig. 5, the obtained multiview feature distance matrix is given in Fig. 6. The highest absolute values for \( \beta \) coefficients calculated for this specific example are for the head word sense feature and phrase structure (0.564 and 0.539), and the lowest for head and tail named entity tags (0.000 and 0.108 respectively), which are values that follow intuitive analysis on the graphical representations of the single-view matrices.

The classification proposed by \( C_{\text{expansion}} \) is distance-based. First, the distance between a relation instance in the training set \( r_{\text{train}} \) and a relation instance in the

![Fig. 5. (a) Expected multiview matrix \( D' \) and (b) elements to be considered for classification.](image-url)
testing set $r_{test}$ is defined with the following equation, which uses the previously calculated $\beta$ coefficients:

$$\delta(r_{train}, r_{test}) = \beta_0 + \sum_{i=1}^{k} \beta_i \cdot \delta^i(f_{train}, f_{test}).$$

As a result, $r_{test}$ can be evaluated as belonging or not to the relation class $R_c$ if it is closer to the positive training instances. However, in order to diminish the impact of noise and not to penalize classes that have more than one way of being defined, a clustering step is added. Spectral clustering [28, 29], which is widely used for image segmentation, is applied to the multiview distance matrix. The $a^*$ parameter of the clustering algorithm, which is responsible for controlling recursive partitioning, is found by grid search, as its optimal value is the one that (1) separates relation instances from different relation classes, (2) better separates instances from the same class if they are similar, and (3) better groups similar instances from the same class.

Figure 7 illustrates examples of good and bad clusterings. A good clustering, shown in Fig. 7(a), more closely resembles a block diagonal matrix, such that when condition (1) is met, there are only blocks in the main diagonal, and when
conditions (2) and (3) are met, these blocks are larger. Figure 7(b) exemplifies the case in which different classes do not seem to be completely separated, and similar relation instances are placed in different clusters.

The distance \( \delta(\text{r}_{\text{test}}, C) \) between \( \text{r}_{\text{test}} \) and a cluster \( C \) is given by the mean between the distances to all cluster members. Given that the classification is the closest cluster, the confidence measure \( \theta' \) is given by:

\[
\theta' = \begin{cases} 
1 - \frac{\delta(\text{r}_{\text{test}}, C_{\text{closest}})}{\sum_j (\text{r}_j, C_j)} & \text{if } C_{\text{cluster}} \text{ is a positive cluster} \\
- \left( 1 - \frac{\delta(\text{r}_{\text{test}}, C_{\text{closest}})}{\sum_j (\text{r}_j, C_j)} \right) & \text{otherwise.}
\end{cases}
\]

(7)

The confidence measure \( \theta' \) ranges from \(-1\) to \(+1\), with values closer to \(-1\) indicating that \( \text{r}_{\text{test}} \) is more confidently classified as negative, values closer to \(+1\) as positive, and values closer to 0 indicating that the classification is inconclusive. For the set expansion, \( \theta' \) is then compared against a pre-defined threshold \( \theta \), and only test relations for which \( \theta' \geq \theta \) holds true are carried for the next step of the bootstrapped process. This aims to increase precision of the classification.

4. Experiments and Discussions

In order to evaluate the proposed method, some experiments were conducted. The first set of experiments measured the quality of the filtering process using a training dataset composed of at most 10 randomly chosen instances per relation class from CDL-annotated Wikipedia articles. The testing dataset was composed by extracted instances that were confidently classified as positive. These instances were manually checked, and then evaluated under the metrics of macro-average precision and macro-average accuracy, described in the equations below. It should be noted that recall is not calculated for this set of experiments because false negatives should not negatively affect the process.

\[
\text{Macro-Average Precision} = \frac{1}{|R_c|} \sum_{R_c} \frac{\text{Correct positive predictions for } R_c}{\text{Total positive predictions for } R_c}
\]

(8)

\[
\text{Macro-Average Accuracy} = \frac{1}{|R_c|} \sum_{R_c} \frac{\text{Correct predictions for } R_c}{\text{Total predictions for } R_c}
\]

(9)

Macro-average is calculated on a per-class basis, in which each class has the same weight, as opposed to the regular micro-average measures, in which each instance has the same weight. As a result, macro-average gives more weight to classes that occur less frequently. In a simple example, if \( R_1 \) has 99 instances of value 1 and \( R_2 \) has one instance of value 0, then the macro-average would be equal to 0.5 and the micro-average to 0.99.
Different classifiers were compared for the low-frequency setting for one iteration of the extraction and filtering tasks. The proposed feature distance-based classifier was compared to the baseline SVM classifier using feature vectors as input [19] and sigmoidal probabilistic output [30, 31] as the confidence measure.

Table 3 presents macro-average precision and accuracy values for the optimal threshold, which was determined experimentally. The baseline classifier is outperformed in this task because the few number of training instances and the sparsity of the feature vector produced an incomplete classification model for the SVM.

Still using the same dataset, different syntactic patterns were also compared, one based on the phrase structure and another on dependencies. For most threshold values used, the values for macro-average precision are higher using dependency, but accuracy values are lower. This is an indication that phrase structure-based filtering extracts more candidate instances, allowing more noise. Accuracy in this case is increased because negative prediction is high. On the other hand, dependency-based filtering extracts more relevant instances, increasing precision but decreasing negative prediction values.

It is important to notice that even with the usage of confidence thresholds, it is not possible to eliminate classification errors. As a result, if error propagation in the bootstrapped process is to be completely avoided, a manual checking step should be included. Although this increases resources consumption, the required effort for this manual checking step is considerably less than that of full corpus annotation. Furthermore, error and bias is expected to be decreased considerably, and so are requirements for annotators’ linguistic backgrounds.

The second set of experiments evaluates the contribution of an expanded training set for the overall semantic relation classification task, compared to when using an unexpanded set. The training data is composed of manually generated CDL relations, with an average of 4.54 instances per each of the 46 classes, and the testing data consists of all semantic relations found in nine fully annotated Wikipedia articles, accounting for overall 12,277 instances from 39 CDL relations. The bootstrapped process for this task includes the manual checking step in order to avoid error propagation, and two iterations of the process were run.

First, a comparison between the relation classification task using large or small training sets is made. The work in [19] used a training data of 13,487 instances, and was evaluated using ten-fold cross validation, resulting in 86.35%, 87.43% and 86.89% of precision, recall and $F$-value respectively for the classification. However, when using the small training set mentioned previously, these figures decreased to

| Method     | Phrase structure |  | Dependency |
|------------|------------------|  |            |
|            | MA Precision     | MA Accuracy |  | MA Precision | MA Accuracy |
| Proposed   | 0.30             | 0.62        |  | 0.45         | 0.50        |
| Baseline   | 0.18             | 0.59        |  | 0.25         | 0.45        |
51.80%, 40.36% and 45.37%. Although this difference is partly explained by the use of a less complete feature set, it is largely due to the smaller training set. Furthermore, if considering macro-average precision, recall and F-value instead of micro-average values, these numbers become 29.15%, 27.19% and 28.14%.

The overall observed improvement after each bootstrapped iteration is given in Tables 4 and 5, with the first providing micro-average and the latter providing macro-average values. Applying only two iterations of the process, the initial set increases more than ten times, from 209 to 2357 training instances after the second iteration. The contribution of the expanded instances in the overall classification task also raises micro-average F-value by 15% and macro-average F-value by 10%. Finally, it is also noticeable that the improvement caused by the first iteration is greater than the improvement from the first to the second iteration, because many of the features that are added in the second iteration were already added in the first.

<table>
<thead>
<tr>
<th>Bootstrapped iterations</th>
<th>Precision %</th>
<th>Recall %</th>
<th>F-Value %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unexpanded</td>
<td>51.80</td>
<td>40.36</td>
<td>45.37</td>
</tr>
<tr>
<td>After 1st</td>
<td>62.04</td>
<td>51.35</td>
<td>56.19</td>
</tr>
<tr>
<td>After 2nd</td>
<td>62.30</td>
<td>59.05</td>
<td>60.63</td>
</tr>
</tbody>
</table>

Table 4. Contribution of expanded instances (micro-average values).

<table>
<thead>
<tr>
<th>Bootstrapped iterations</th>
<th>Precision %</th>
<th>Recall %</th>
<th>F-Value %</th>
</tr>
</thead>
<tbody>
<tr>
<td>Unexpanded</td>
<td>29.15</td>
<td>27.19</td>
<td>28.14</td>
</tr>
<tr>
<td>After 1st</td>
<td>36.74</td>
<td>34.95</td>
<td>35.82</td>
</tr>
<tr>
<td>After 2nd</td>
<td>42.30</td>
<td>35.55</td>
<td>38.64</td>
</tr>
</tbody>
</table>

Table 5. Contribution of expanded instances (macro-average values).

5. Conclusion

For the supervised classification task of semantic relations, one alternative to full corpus annotation is set expansion. This work presented a bootstrapped set expansion method, applying it for contextual semantic relations, whose modeling is done by various morphological, syntactic and lexico-semantic features. By using a double-layered filtering step in order to address the impossibility of applying syntactic features into the Web query, and by using a feature distance-based classifier with controlled input in order to classify instances even in small training set settings, the proposed method was able to improve the quality of the semantic relation classification task when using the expanded training set, if compared to when using datasets with few instances per relation class. This strongly suggests that the relation instances that were added to that initial dataset are not only relevant to classification, but also provide new and recombined features to the feature space of the classifying model.
Moreover, the single and multiview matrices calculated using feature distances allow analysis of the quality of the separation that a feature set provides given a training dataset prior to running the classification process. This is a desirable characteristic for the set expansion process and for prototyping classification systems.

The set expansion, nevertheless, was not able to guarantee that errors are not passed forward to the next bootstrapping iterations even by setting high threshold values. As a result, a manual checking step needs to be introduced. Although this step is more resource consuming than a completely automatic approach, it still loosens resource requirements when compared to full corpus annotation. A quantitative comparison between this semi-automatic approach and manual annotation needs further investigation.

Finally, although this research was applied specifically to CDL, using set expansion and a feature distance-based classifier for training sets with few instances can greatly decrease the effort that would be employed for full corpus manual annotation of contextual semantic relations. This work also provides an indication that distance-based classification may yield to better classification in settings in which the testing data presents many features that are not found in the training data.

References


