UVAVU: WordNet Similarity and Lexical Patterns for Semantic Relation Classification

Willem Robert van Hage
TNO Science & Industry
Stieltjesweg 1, 2628CK Delft
the Netherlands
wrvhage@few.vu.nl

Sophia Katrenko
HCSL, University of Amsterdam
Kruislaan 419, 1098VA Amsterdam
the Netherlands
katrenko@science.uva.nl

Abstract

The system we propose to learning semantic relations consists of two parallel components. For our final submission we used components based on the similarity measures defined over WordNet and the patterns extracted from the Web and WMTS. Other components using syntactic structures were explored but not used for the final run.

1 Experimental Set-up

The system we used to classify the semantic relations consists of two parallel binary classifiers. We ran this system for each of the seven semantic relations separately. Each classifier predicts for each instance of the relation whether it holds or not. The predictions of all the classifiers are aggregated for each instance by disjunction. That is to say, each instance is predicted to be false by default unless any of the classifiers gives evidence against this.

To generate the submitted predictions we used two parallel classifiers: (1) a classifier that combines eleven WordNet-based similarity measures, see Sec. 2.1, and (2) a classifier that learns lexical patterns from Google and the Waterloo Multi-Text System (WMTS)(Turney, 2004) snippets and applies these on the same corpora, see Sec. 2.2.

Three other classifiers we experimented with, but that were not used to generate the submitted predictions: (3) a classifier that uses string kernel methods on the dependency paths of the training sentences, see Sec. 3.1, (4) a classifier that uses string kernels on the local context of the subject and object nominals in the training sentences, see Sec. 3.2 and (5) a classifier that uses hand-made lexical patterns on Google and WMTS, see Sec. 3.3.

2 Submitted Run

2.1 WordNet-based Similarity Measures

WordNet 3.0 (Fellbaum, 1998) is the most frequently used lexical database of English. As this resource consists of lexical and semantic relations, its use constitutes an appealing option to learning relations. In particular, we believe that given two mentions of the same semantic relation, their arguments should also be similar. Or, in analogy learning terms, if $R_1(X_1,Y_1)$ and $R_2(X_2,Y_2)$ are relation mentions of the same type, then $X_1 :: Y_1$ as $X_2 :: Y_2$. Our preliminary experiments with WordNet suggested that few arguments of each relation are connected by immediate hyperonymy or meronymy relations. As a result, we decided to use similarity measures defined over WordNet (Pedersen et al., 2004). The WordNet::Similarity package (Pedersen et al., 2004) includes 11 different measures, which mostly use either the WordNet glosses (*lesk* or *vector* measures) or the paths between a pair of concepts (*lch*; *wup*) to determine their relatedness.

To be able to use WordNet::Similarity, we mapped all WordNet sense keys from the training and test sets to the earlier WordNet version (2.1). Given a relation $R(X,Y)$, we computed the relatedness scores for each pair of arguments X and Y. The scores together with the sense keys of arguments were further used as features for the machine learning method. As there is no a priori knowledge on what measures are the most important for each relation, all of them were used and no feature selection step has been taken.

We experimented with a number of machine learning methods such as k-nearest neighbour algorithm, logistic regression, bayesian networks and others. For each relation a method performing best on the training set was selected (using 5-fold cross-validation).

2.2 Learnt Lexical Patterns

This classifier models the intuition that when a pair of nominals is used in similar phrases as another pair they share at least one relation, and when no such phrases can be found they do not share any relation. Applied to the semantic relation classification problem this means that when a pair in the test set can be found in the same patterns as pairs from the training set, the classification for the pair will be true.

To find the patterns we followed step 1 to 6 described in (Turney, 2006), with the exception that we used both Google and the WMTS to compute pattern frequency.

First we extracted the pairs of nominals $\langle X, Y \rangle$ from the training sentences and created one Google query and a set of WMTS queries for each pair. The Google queries were of the form `"X * Y" OR "Y * X"`. Currently, Google performs morphological normalization on every query, so we did not make separate queries for various endings of the nominals. For the WMTS we did make separate queries for various morphological variations. We used the following set of suffixes: '-tion(s|al)', '-ly', '-ist', '-ical', '-y', '-ing', '-ed', '-ies', and '-s'. For this we used Peter Turney's `pairs` Perl package. The WMTS queries looked like `[n]>([5].."X"..[i].."Y"..[5])` and `[n]>([5].."Y"..[i].."X"..[5])` for $i = 1,2,3$ and $n = i + 12$, and for each variation of X and Y. Then we extracted sentences from the Google snippets and cut out a context of size 5, so that we were left with similar text segments as those returned by the WMTS queries. We merged the lists of text segments and counted all n-grams that contained both nominals for $n = 1$ to 6. We substituted the nominals by variables in the n-grams with a count greater than 10 and used these as patterns for the classifier. An example of such a pattern for the Cause-Effect relation is `"generation of Y by X"`. After this we followed step 3 to 6 of (Turney, 2006), which left us with a matrix for each of the seven semantic relations, where each row represented a pair of nominals and each column represented the frequency of a pattern, and where each pair was classified as either true or false. The straightforward way to find pattern frequencies for the pairs in the test set would be to fill in these patterns with the pairs of nominals from the test set. This was not feasible given the time limitation on the task. So instead, for each pair of nominals in the test set we gathered the top-1000 snippets and computed pattern frequencies by counting how often the nominals occur in every pattern on this set text segments. We constructed a matrix from these frequencies in the same way as for the training set, but without classifications for the pairs. We experimented with various machine learning algorithms to predict the classes of the pairs. We chose to use k-nearest neighbors, because it was the only algorithm that gave more subtle predictions than true for every pair or false for every pair. For each semantic relation we used the value of k that produced the highest F_1 score on 5-fold cross validation on the training data.

3 Additional Runs

3.1 String Kernels on Dependency Paths

It has been a long tradition to use syntactic structures for relation extraction task. Some of the methods as in (Katrenko and Adriaans, 2004) have used information extracted from the dependency trees. We followed similar approach by considering the paths between each pair of arguments X and Y. Ideally, if each syntactic structure is a tree, there is only one path from one node to the other. After we have extracted paths, we used them as input for the string kernel methods (Hal Daumé III, 2004). The advantage of using string kernels is that they can handle sequences of different lengths and already proved to be efficient for a number of tasks.

All sentences in the training data were parsed using MINIPAR (Lin, 1998). From each dependency tree we extracted a dependency path (if any) between the arguments by collecting all lemmas (nodes) and syntactic functions (edges). The sequences we obtained were fed into string kernel.

To assess the results, we carried out 5-fold cross-validation. Even by optimizing the parameters of the kernel (such as the length of subsequences) for each relation, the highest accuracy we obtained was equal 61,54% (on Origin-Entity relation) and the lowest was accuracy for the Instrument-Agency relation (50,48%).

3.2 String Kernels on Local Context

Alternatively to syntactic information, we also extracted the snippets of the fixed length from each sentence. For each relation mention of $R(X,Y)$, all tokens between the relation arguments X and Y were collected along with at most three tokens to the left and to the right. Unfortunately, the results we obtained on the training set were comparable to those obtained by string kernels on dependency paths and less accurate than the results provided by WordNet similarity measures or patterns extracted from the Web and WMTS. As a consequence, string kernel methods were not used for the final submission.

3.3 Manually-created Lexical Patterns

The results of the method described in Sec. 2.2 are quite far below what we expected given earlier results in the literature (Turney, 2006; van Hage, Katrenko, and Schreiber, 2005; van Hage, Kolb, and Schreiber, 2006; Berland and Charniak, 2006; Etzioni et al., 2004). We think this is caused by the fact that many pairs in the training set are non-stereotypical examples. So often the most commonly described relation of such a pair is not the relation we try to classify with the pair. For example, common associations with the pair ⟨body,parents⟩ are that it is the parents' body, or that the parents are member of some organizing body, while it is a positive example for the Product-Producer relation. We wanted to see if this could be the case by testing whether more intuitive patterns give better results on the test set. The patterns we manually created for each relation are shown in Table 1. If a pair gives any results for these patterns on Google or WMTS, we classify the pair as true, otherwise we classify it as false. The results are shown in Table 2. We did not use these results for the submitted run, because only automatic runs were permitted. The manual patterns did not yield many useful results at all. Apparently intuitive patterns do not capture what is required to classify the relations in the test set. The patterns we used for the Part-Whole (6) relation had an average Precision of .50, which is much lower than the average Precision found in (van Hage, Kolb, and Schreiber, 2006), which was around 0.88. We conclude that both the sets of training and test examples capture different semantics of the relations than the intuitive ones, which causes common sense background knowledge, such as Google to produce bad results.

rel.	patterns
1.	X causes Y, X caused by Y, X * cause Y
2.	X used Y, X uses Y, X * with a Y
3.	X made by Y, X produced by Y, Y makes X, Y produces X
4.	Y comes from X, X * source of Y, Y * from * X
5.	Y * to * X, Y * for * X, used Y for * X
6.	X in Y, Y contains X, X from Y
7.	Y contains X, X in Y, X containing Y, X into Y

Table 1: Hand-written patterns.

relation	N	Prec.	Recall	F_1	Acc.
1. Cause-Effect	6	1	0.15	0.25	0.56
2. Instr.-Agency	2	1	0.05	0.10	0.54
3. Prod.-Prod.	4	0.75	0.05	0.09	0.35
4. Origin-Ent.	6	0.33	0.05	0.09	0.35
5. Theme-Tool	2	0	0	0	0.56
6. Part-Whole	16	0.50	0.31	0.38	0.64
7. Cont.-Cont.	11	0.54	0.16	0.24	0.50

Table 2: Results for hand-written lexical patterns on Google and WMTS.

4 Results

4.1 WordNet-based Similarity Measures

Table 3 shows the results of the WordNet-based similarity measure method. In the 'methods' column, the abbreviation LR stands for logistic regression, K-NN stands for k-nearest neighbour, and DT stands for decision trees.

relation	method	Prec.	Recall	F_1	Acc.
1. Cause-Effect	LR	0.48	0.51	0.49	0.45
2. Instr.-Agency	DT	0.65	0.63	0.64	0.62
3. Prod.-Prod.	DT	0.67	0.50	0.57	0.46
4. Origin-Ent.	LR	0.50	0.47	0.49	0.49
5. Theme-Tool	LR	0.54	0.52	0.53	0.62
6. Part-Whole	DT	0.54	0.73	0.62	0.67
7. Cont.-Cont.	2-NN	0.66	0.55	0.60	0.62

Table 3: Results for similarity-measure methods.

4.2 Learnt Lexical Patterns

Table 4 shows the results of the learnt lexical patterns method. For all relations we used the *k*-nearest neighbour method.

relation	method	Prec.	Recall	F_1	Acc.
1. Cause-Effect	3-NN	0.53	0.76	0.63	0.54
2. Instr.-Agency	2-NN	0.47	0.89	0.62	0.46
3. Prod.-Prod.	2-NN	0	0	0	0.33
4. Origin-Ent.	2-NN	0.47	0.22	0.30	0.54
5. Theme-Tool	3-NN	0.39	0.93	0.55	0.38
6. Part-Whole	2-NN	0.36	1	0.53	0.36
7. Cont.-Cont.	2-NN	0.51	0.97	0.67	0.51

Table 4: Results for learnt lexical patterns on Google and WMTS.

5 Discussion

Our methods had the most difficulty with classifying relation 1, 3 and 4. We wanted to see if human assessors perform less consistent for those relations. If so, then those relations would simply be harder to classify. Otherwise, our system performed worse for those relations. We manually assessed ten sample sentences from the test set, five of which were positive examples and five were false examples. The result of a comparison with the test set is shown in Table 5. The numbers listed there represent the fraction of examples on which we agreed with the judges of the test set. There was quite a

	inter-judge agreement	
relation	judge 1	judge 2
1. Cause-Effect	0.93	0.93
2. Instrument-Agency	0.77	0.77
3. Product-Producer	0.87	0.80
4. Origin-Entity	0.80	0.77
5. Theme-Tool	0.80	0.77
6. Part-Whole	0.97	1.00
7. Content-Container	0.77	0.77

Table 5: Inter-judge agreement.

large variation in the inter-judge agreement, but for relation 1 and 3 the consensus was high. We conclude that the reason for our low performance on those relations are not caused by the difficulty of the sentences, but due to other reasons. Our intuition is that the sentences, especially those of relation 1 and 3, are easily decidable by humans, but that they are non-stereotypical examples of the relation, and thus hard to learn. The following example sentence breaks common-sense domain and range restrictions: Product-Producer #142 *"And, of course, everyone wants to prove the truth of their beliefs through experience, but the <e1>belief</e1> begets the <e2>experience</e2>."* The common-sense domain and range restriction of the Product-Producer relation are respectively something like 'Entity' and 'Agent'. However, 'belief' is generally not considered to be an entity, and 'experience' not an agent. The definition of Product-Producer relation used for the Challenge is more flexible and allows therefore many examples which are difficult to find by such common-sense resources as Google or WordNet.

References

Matthew Berland and Eugene Charniak. 1999. Finding Parts in Very Large Corpora. *In Proceedings of ACL 1999.*

Christiane Fellbaum (ed.). 1998. WordNet: An Electronic Lexical Database. *MIT Press.*

Hal Daumé III. 2004. SVMsequel Tutorial Manual. *Available at* `http://www.cs.utah.edu/~hal/SVMsequel/svmsequel.pdf`

Oren Etzioni et al. 2004. Methods for Domain-Independent Information Extraction from the Web: An Experimental Comparison. *In Proceedings of AAAI 2004.*

Willem Robert van Hage, Sophia Katrenko, and Guus Schreiber. 2005. A Method to Combine Linguistic Ontology-Mapping Techniques. *In Proceedings of ISWC 2005.*

Willem Robert van Hage, Hap Kolb, and Guus Schreiber. 2006. A Method for Learning Part-Whole Relations. *In Proceedings of ISWC 2006.*

Sophia Katrenko and Pieter Adriaans. 2007. Learning Relations from Biomedical Corpora Using Dependency Trees. *In KDECB, LNBI, vol. 4366.*

Dekang Lin. 1998. Dependency-based Evaluation of MINIPAR. *In Workshop on the Evaluation of Parsing Systems, Granada, Spain.*

Ted Pedersen, Patwardhan, and Michelizzi. 2004. WordNet::Similarity - Measuring the Relatedness of Concepts. *In the Proceedings of AAAI-04, San Jose, CA.*

Peter Turney. 2006. Expressing Implicit Semantic Relations without Supervision. *In Proceedings of COLING-ACL 2006.*

Peter Turney. 2004. The MultiText Project Home Page, University of Waterloo, School of Computer Science, `http://www.multitext.uwaterloo.ca`