Detecting compositionality using semantic vector space models based on syntactic context. Shared task system description*

Guillermo Garrido
NLP & IR Group at UNED
Madrid, Spain
ggarrido@lsi.uned.es

Anselmo Peñas
NLP & IR Group at UNED
Madrid, Spain
anselmo@lsi.uned.es

Abstract

This paper reports on the participation of the NLP GROUP at UNED in the DiSCo'2011 compositionality evaluation task. The aim of the task is to predict compositionality judgements assigned by human raters to candidate phrases, in English and German, from three common grammatical relations: adjective-noun, subject-verb and subject-object.

Our participation is restricted to adjective-noun relations in English. We explore the use of syntactic-based contexts obtained from large corpora to build classifiers that model the compositionality of the semantics of such pairs.

1 Introduction

This paper reports on the NLP GROUP at UNED 's participation in DiSCo'2011 Shared Task. We attempt to model the notion of compositionality from analyzing language use in large corpora. In doing this, we are assuming the distributional hypothesis: *words that occur in similar contexts tend to have similar meanings* (Harris, 1954). For a review of the field, see (Turney and Pantel, 2010).

1.1 Approach

In previous approaches to compositionality detection, different kinds of information have been used: morphological, lexical, syntactic, and distributional. For our participation, we are interested in exploring, exclusively, the reach of pure syntactic information to explain semantics.

Our approach draws from the Background Knowledge Base representation of texts introduced in (Peñas and Hovy, 2010). We hypothesize that behind syntactic dependencies in natural language there are semantic relations; and that syntactic contexts can be leveraged to represent meaning, particularly of nouns. A system could learn these semantic relations from large quantities of natural language text, to build an independent semantic resource, a Background Knowledge Base (BKB) (Peñas and Hovy, 2010).

From a dependency-parsed corpus, we automatically harvest meaning-bearing patterns, matching the dependency trees to a set of pre-specified syntactic patterns, similarly to (Pado and Lapata, 2007). Patterns are matched to dependency trees to produce propositions, carriers of minimal semantic units. Their frequency in the collection is the fundamental source of our representation.

Our participation, due to time constraints, is restricted to adjective-noun pairs in English.

2 System Description

Our hypothesis can be spelled out as: words (or word compounds) with similar syntactic contexts are semantically similar.

The intuition behind our approach is that non-compositional compounds are units of meaning. Then, the meaning of an adjective-noun combination that is not compositional should be different from the meaning of the noun alone; for similar

* This work has been partially supported by the Spanish Ministry of Science and Innovation, through the project Holopedia (TIN2010-21128-C02), and the Regional Government of Madrid, through the project MA2VICMR (S2009/TIC1542).

Proceedings of the Workshop on Distributional Semantics and Compositionality (DiSCo'2011), pages 43–47,

Portland, Oregon, 24 June 2011.

approaches, see (Baldwin et al., 2003; Katz and Giesbrecht, 2006; Mitchell and Lapata, 2010). We propose studying the distributional semantics of a *adjective-noun compound*; in particular, we will represent it via its syntactic contexts.

2.1 Adjective-noun compounds

Given a particular adjective-noun compound, denoted $\langle a, n\rangle$, we want to measure its compositionality by comparing its syntactic contexts to those of the noun: $\langle n\rangle$. After exploring the dataset we realized that considering nouns alone introduced noise, as contexts of the target and different meanings of the noun might be hard to separate; in order to soften this problem we decided to compare the occurrences of the $\langle a, n\rangle$ pair to those of the noun with a *different adjective*.

Given a dependency-parsed corpus C, we denote N the set of all nouns occurring in C and A the set of all adjectives. An adjective-noun pair, $\langle a, n\rangle$, is an occurrence in the dependency parse of the sentence of an arc (a, n), where n is the governor of an adjectival relation, with a as modifier. We define the *complementary* of $\langle a, n\rangle$ as the set of all adjective-noun pairs with the same noun but a different adjective:

$$\langle a^c, n\rangle = \{\langle b, n\rangle \text{ such that } b \in A, b \neq a\}$$

In order to detect compositionality, we compare the semantics of $\langle a, n\rangle$ to those of its complementary $\langle a^c, n\rangle$. We use syntactic context as the representation of these compounds' semantics.

We call *target pairs* those $\langle a, n\rangle$ in which we are interested, as they appear in the training, validation, or test sets for the task. For each of them, its complementary target is: $\langle a^c, n\rangle$.

We model the syntactic contexts of any $\langle a, n\rangle$ pair as a *set* of vectors in a set of vector spaces defined as follows. After inspection of the corpus, and its dependency parse annotation layer, we manually specified a few syntactic relations, which we consider codify the relevant syntactic relations in which an $\langle a, n\rangle$ takes part. For each of these syntactic relations, we built a vector space model, and we represented as a vector in it each of the target patterns, and each of their respective complementary targets. To compute compositionality of a target, we calculated the cosine similarity between the target vector and the target's complementary vector. So, for each syntactic relation, and for each target, we have a value of its similarity to the complementary target. These similarity values are considered features, from which to learn the compositionality of targets.

For results comparability, we used the PukWaC corpus[1] as dataset. PukWaC adds to UkWaC a layer of syntactic dependency annotation. The corpus has been POS-tagged and lemmatized with the TreeTagger[2]. The dependency parse was done with MaltParser (Nivre and Scholz, 2004).

2.2 Implementation details

We defined a set of 19 syntactic patterns that define interesting relations in which an $\langle a, n\rangle$ pair might take part, trying to exploit the dependencies produced by the MaltParser (Nivre and Scholz, 2004), including:

- Relations to a verb, other than the auxiliary to be and to have: subject; object; indirect object; subject of a passive construction; logical subject of a passive construction.
- The relations defined in the previous point, enriched with a noun that acts as the other element of a [subject-verb-object] or [subject-passive verb-logical subject] construction.
- Collapsed prepositional complexes.
- Noun complexes.
- As subject or object of the verb to be.
- Modified by a second adjective.
- As modifier of a possessive.

The paths were defined manually to match our intuitions of which are the paths that best describe the context of an $\langle a, n\rangle$pair, similarly to (Pado and Lapata, 2007). For each of the patterns, the set of words that are related through it to the target $\langle a, n\rangle$ define the target's *context*.

For most of our processing, we used simple programs implemented in Prolog and Python. We implemented Prolog programs to model the dependency parsed sentences of the full PUkWaC corpus, and to match and extract these patterns from them. After an aggregating step, where proper nouns, numbers and dates are substituted by place-holder vari-

[1] Available at `http://wacky.sslmit.unibo.it`

[2] `http://www.ims.uni-stuttgart.de/projekte/corplex/TreeTagger/DecisionTreeTagger.html`

ables, they amount to over 16 million instances, representing the syntactic relations in which every $\langle a, n\rangle$ pair in the corpus takes part. In further processing, only those that affect the target pairs, or the nouns in them, have to be taken into account.

As described above, each pattern we have defined yields a vector space, where each target and its complementary are represented as a vector. The base vectors of the vector space model for a pattern are the words that are syntactic contexts, with that syntactic pattern, of any target in the target set[3].

The value of the coordinate for a target and a base vector is the frequency of the context word as related to the target by the pattern. All frequencies were locally scaled using logarithms[4].

For each syntactic pattern, and for each target and complementary, we have two vectors, representing their meanings in the vector space distributional model. The complementary vector, in particular, represents the centroid (average) of the meanings of all $\langle b, n\rangle$ pairs, that share the noun with the target but have a different adjective, b

We propose that a target will be more compositional if its meaning is more similar to the meaning of the centroid of its complementary, that codifies the general meaning of that noun (whenever it appears with a different adjective).

For each syntactic pattern and target, we can compute the cosine similarity to the complementary target, and obtain a value to use as a feature of the compositionality of the target. Those features will be used to train a classifier, being the compositionality score of each sample the label to be learnt.

We used RapidMiner[5] (Mierswa et al., 2006) as our Machine Learning framework. The classifiers we have used, that are described below, are the implementations available in RapidMiner.

[3]It would have been possible to consider a common vector space, using all patterns as base vectors. We decided not to do so after realising that a single similarity value for a target and its complementary was not by itself a signal strong enough to predict the compositionality score. A second objective was to assess the relative importance of different syntactic contexts for the task.

[4]We did not attempt any global weighting. We leave this for future work.

[5]http://rapid-i.com

2.3 Feature selection

From the 19 original features, inspection of the correlation to the compositionality score label showed that some of them were not to be expected to have much predictive power, while some of them were too sparse in the collection.

We decided to perform feature selection previous to all subsequent learning steps. We used RapidMiner genetic algorithm for feature selection[6]. Among the patterns which features were not selected were those where the $\langle a, n\rangle$ pair appears in prepositional complexes, in noun complexes, as indirect object, as subject or object of the verb to be, and as subject of a possessive. Among those selected were subject and objects of both active and passive constructions, and the object of possessives.

2.4 Runs description

Numeric scores For the numeric evaluation task, we built a regression model by means of a SVM classifier. We used RapidMiner's implementation of mySVMLearner (Rüping, 2000), that is based on the optimization algorithm of SVM-light (Joachims, 1998). We used the default parameters for the classifier. A simple dot product kernel seemed to obtain the best results in 10-fold cross validation over the union of the provided train and validation results. For the three runs, we used identical settings, optimizing different quality measures in each run: absolute error (RUN SCORE-1), Pearson's correlation coefficient (RUN SCORE-2), and Spearman's rho (RUN SCORE-3). The choice of a SVM classifier was motivated by the objective of learning a good parametric classifier model. In initial experiments, SVM showed to perform better than other possible choices, like logistic regression. In hindsight, the relatively small size of the dataset might be a reason for the relatively poor results. Experimenting with other approaches is left for future work.

Coarse scores For the coarse scoring, we decided to build a different set of classifiers, that would learn the nominal 3-valued compositionality label. The classifiers built in our initial experiments turned out

[6]The mutation step switches features on and off, while the crossover step interchanges used features. Selection is done randomly. The algorithm used to evaluate each of the feature subsets was a SVM identical as the one described below.

Run	$avg_{\triangle}$	r	ρ
RUN-SCORE-1	16.395	0.483	0.487
RUN-SCORE-2	15.874	0.475	0.463
RUN-SCORE-3	16.318	0.494	0.486
baseline	17.857	–	–

Table 1: TRAINING. Numeric score runs results on 10-fold cross-validation for the training set. $avg_{\triangle}$: average absolute error; r: Pearson's correlation;ρ: Spearman's rho.

Run	$avg_{\triangle}$	r	ρ
RUN-SCORE-1	17.016	0.237	0.267
RUN-SCORE-2	17.180	0.217	0.219
RUN-SCORE-3	17.289	0.180	0.189
baseline	17.370	–	–

Table 2: TEST. Numeric score runs for the test set. Only for the en-ADJ-NN samples. $avg_{\triangle}$: average absolute error; r: Pearson's correlation;ρ: Spearman's rho.

to lazily choose the most frequent class ("high") for most of the test samples. In an attempt to overcome this situation and possibly learn non linearly separable classes, we tried neural network classifiers[7]. In hindsight, from seeing the very poor performance of this classifiers on the test set, it is clear that any performance gains were due to over-fitting on the training set.

For RUN COARSE-2, we binned the numeric scores obtained in RUN-SCORE-1, dividing the score space in three equal sized parts; we decided not to assume the same distribution of the three labels for the training and test sets. The results were worse than the numeric scores, due to the fact that the 3 classes are not equally sized.

2.5 Results

Results in the training phase For all our training, we performed 10-fold cross validation. For reference, we report the results as evaluated by averaging over the 10 splits of the union of the provided training and validation set in Table 1. We compared against a dummy baseline: return as constant score the average of the scores in the training and validation sample sets.

Disappointingly, the resulting classifiers seemed to be quite *lazy*, yielding values significatively close to the average of the compositionality label in the training and validation set.

The AutoMNLP and neural network seemed to perform reasonably, and better than other classifiers we tried (e.g., SVM based). We were wary, though, of the risk of having learnt an over-fitted model; unfortunately, the results on the test set confirmed that: for instance, the accuracy of RUN-SCORE-3 for the training set was 0.548, but for the test set it was only 0.327.

Results in the test phase After the task results were distributed, we verified that our numeric score runs, for the subtask en-ADJ-NN performed quite well: fifth among the 17 valid submissions for the subtask, using the average point difference as quality measure. Nevertheless, in terms of ranking correlation scores, our system performs presumably worse, although separate correlation results for the en-ADJ-NN subtask were not available to us at the time of writing this report.

Our naive baseline turns out to be strong in terms of average point score. Of course, the ranking correlation of such a baseline is none; using ranking correlation as quality measure would be more sensible, given that it discards such a baseline.

3 Conclusions

We obtained modest results in the task. Our three numeric runs obtained results very similar to each other. Only taking part in the en-ADJ-NN subtask, we obtained the 5th best of a total of 17 valid systems in average point difference. Nevertheless, in terms ranking correlation scores, our systems seem to perform worse. The modifications we tried to specialize for coarse scoring were unsuccessful, yielding poor results.

A few conclusions we can draw at this moment are: our system could benefit from global frequency weighting schemes that we did not try but that have shown to be successful in the past; the relatively small size of the dataset has not allowed us to learn a better classifier; finally, we believe the ranking correlation quality measures are more sensible than the point difference for this particular task.

[7]For RUN COARSE-1, we used AutoMLP (Breuel and Shafait, 2010), an algorithm that learns a neural network, optimizing both the learning rate and number of hidden nodes of the network. For RUN COARSE-3, we learnt a simple neural network model, by means of a feed-forward neural network trained by a backpropagation algorithm (multi-layer perceptron), with a hidden layer with sigmoid type and size 8.

References

Timothy Baldwin, Colin Bannard, Takaaki Tanaka, and Dominic Widdows. 2003. An empirical model of multiword expression decomposability. In *Proceedings of the ACL 2003 workshop on Multiword expressions: analysis, acquisition and treatment - Volume 18*, MWE '03, pages 89–96, Stroudsburg, PA, USA. Association for Computational Linguistics.

Thomas Breuel and Faisal Shafait. 2010. Automlp: Simple, effective, fully automated learning rate and size adjustment. In *The Learning Workshop*. Online, 4.

Zellig S. Harris. 1954. Distributional structure. *Word*, pages 146–162.

Thorsten Joachims. 1998. Making large-scale svm learning practical. LS8-Report 24, Universität Dortmund, LS VIII-Report.

Graham Katz and Eugenie Giesbrecht. 2006. Automatic identification of non-compositional multi-word expressions using latent semantic analysis. In *Proceedings of the Workshop on Multiword Expressions: Identifying and Exploiting Underlying Properties*, MWE '06, pages 12–19, Stroudsburg, PA, USA. Association for Computational Linguistics.

Ingo Mierswa, Michael Wurst, Ralf Klinkenberg, Martin Scholz, and Timm Euler. 2006. Yale: rapid prototyping for complex data mining tasks. In *KDD'06*, pages 935–940.

Jeff Mitchell and Mirella Lapata. 2010. Composition in distributional models of semantics. *Cognitive Science*, 34(8):1388–1429.

Joakim Nivre and Mario Scholz. 2004. Deterministic dependency parsing of english text. COLING '04.

Sebastian Pado and Mirella Lapata. 2007. Dependency-Based Construction of Semantic Space Models. *Computational Linguistics*, 33(2):161–199, jun.

Anselmo Peñas and Eduard Hovy. 2010. Semantic enrichment of text with background knowledge. pages 15–23, jun.

Stefan Rüping. 2000. mySVM-Manual. `http://www-ai.cs.uni-dortmund.de /SOFTWARE/MYSVM/`.

Peter D. Turney and Patrick Pantel. 2010. From frequency to meaning: Vector space models of semantics. *J. Artif. Intell. Res. (JAIR)*, 37:141–188.