Learning to rank lexical substitutions

György Szarvas[1]
Amazon Inc.
szarvasg@amazon.com

Róbert Busa-Fekete[2] **Eyke Hüllermeier**
University of Marburg
Hans-Meerwein-Str., 35032 Marburg, Germany
{busarobi,eyke}@mathematik.uni-marburg.de

Abstract

The problem to replace a word with a synonym that fits well in its sentential context is known as the lexical substitution task. In this paper, we tackle this task as a supervised ranking problem. Given a dataset of target words, their sentential contexts and the potential substitutions for the target words, the goal is to train a model that accurately ranks the candidate substitutions based on their contextual fitness. As a key contribution, we customize and evaluate several learning-to-rank models to the lexical substitution task, including classification-based and regression-based approaches. On two datasets widely used for lexical substitution, our best models significantly advance the state-of-the-art.

1 Introduction

The task to generate lexical substitutions in context (McCarthy and Navigli, 2007), i.e., to replace words in a sentence without changing its meaning, has become an increasingly popular research topic. This task is used, e.g. to evaluate semantic models with regard to their accuracy in modeling word meaning in context (Erk and Padó, 2010). Moreover, it provides a basis of NLP applications in many fields, including linguistic steganography (Topkara et al., 2006; Chang and Clark, 2010), semantic text similarity (Agirre et al., 2012) and plagiarism detection (Gipp et al., 2011). While closely related to WSD, lexical substitution does not rely on explicitly defined sense inventories (Dagan et al., 2006): the possible substitutions reflect all conceivable senses of the word, and the correct sense has to be ascertained to provide an accurate substitution.

While a few lexical sample datasets (McCarthy and Navigli, 2007; Biemann, 2012) with human-provided substitutions exist and can be used to evaluate different lexical paraphrasing approaches, a practically useful system must also be able to rephrase unseen words, i.e., any word for which a list of synonyms is provided. Correspondingly, unsupervised and knowledge-based approaches that are not directly dependent on any training material, prevailed in the SemEval 2007 shared task on English Lexical Substitution and dominated follow-up work. The only supervised approach is limited to the combination of several knowledge-based lexical substitution models based on different underlying lexicons (Sinha and Mihalcea, 2009).[3]

A recent work by Szarvas et al. (2013) describes a tailor-made supervised system based on delexicalized features that – unlike earlier supervised approaches, and similar to unsupervised and knowledge-based methods proposed for this task – is able to generalize to an open vocabulary. For each target word to paraphrase, they first compute a set of substitution candidates using WordNet: all synonyms from all of the target word's WordNet synsets, together with the words from synsets in *similar to, entailment* and *also see* relation to these synsets are considered as potential substitutions. Each candidate then constitutes a training (or test)

[1] Work was done while working at RGAI of the Hungarian Acad. Sci. and University of Szeged.

[2] R. Busa-Fekete is on leave from the Research Group on Artificial Intelligence of the Hungarian Academy of Sciences and University of Szeged.

[3] Another notable example for supervised lexical substitution is Biemann (2012), but this is a lexical sample system applicable only to the target words of the training datasets.

example, and these instances are characterized using non-lexical features from heterogeneous evidence such as lexical-semantic resources and distributional similarity, n-gram counts and shallow syntactic features computed on large, unannotated background corpora. The goal is then i) to predict how well a particular candidate fits in the original context, and ii) given these predictions for each of the candidates, to correctly order the elements of the candidate set according to their contextual fitness. That is, a model is successful if it prioritizes plausible substitutions ahead of less likely synonyms (given the context). This model is able to generate paraphrases for target words not contained in the training material. This favorable property is achieved using only such features (e.g. local n-gram frequencies in context) that are meaningfully comparable across the different target words and candidate substitutions they are computed from. More importantly, their model also provides superior ranking results compared to state of the art unsupervised and knowledge based approaches and therefore it defines the current state of the art for open vocabulary lexical substitution.

Motivated by the findings of Szarvas et al. (2013), we address lexical substitution as a supervised learning problem, and go beyond their approach from a methodological point of view. Our experiments show that the performance on the lexical substitution task is strongly influenced by the way in which this task is formalized as a machine learning problem (i.e., as binary or multi-class classification or regression) and by the learning method used to solve this problem. As a result, we are able to report the best performances on this task for two standard datasets.

2 Related work

Previous approaches to lexical substitution often seek to automatically generate a set of candidate substitutions for each target word first, and to rank the elements of this set of candidates afterward (Hassan et al., 2007; Giuliano et al., 2007; Martinez et al., 2007; Yuret, 2007). Alternatively, the candidate set can be defined by all human-suggested substitutions for the given target word in all of its contexts; then, the focus is just on the ranking problem (Erk and Padó, 2010; Thater et al., 2010; Dinu and Lapata, 2010; Thater et al., 2011). While only the former approach qualifies as a full-fledged substitution system for arbitrary, previously unseen target words, the latter simplifies the comparison of semantic ranking models, as the ranking step is not burdened with the shortcomings of automatically generated substitution candidates.

As mentioned before, Szarvas et al. (2013) recently formalized the lexical substitution problem as a supervised learning task, using delexicalized features. This non-lexical feature representation makes different target word/substitution pairs in different contexts[4] directly comparable. Thus, it becomes possible to learn an all-words system that is applicable to unseen words, using supervised methods, which provides superior ranking accuracy to unsupervised and knowledge based models.

In this work, we build on the problem formulation and the features proposed by Szarvas et al. (2013) while largely extending their machine learning methodology. We customize and experiment with several different learning-to-rank models, which are better tailored for this task. As our experiments show, this contribution leads to further significant improvements in modeling the semantics of a text and in end-system accuracy.

3 Datasets and experimental setup

Here we introduce the datasets, experimental setup and evaluation measures used in our experiments. Since space restrictions prohibit a comprehensive exposition, we only provide the most essential information and refer to Szarvas et al. (2013), whose experimental setup we adopted, for further details.

Datasets. We use two prominent datasets for lexical substitution. The *LexSub dataset* introduced in the Lexical Substitution task at Semeval 2007 (McCarthy and Navigli, 2007)[5] contains 2002 sentences for a total of 201 target words (from *all parts of speech*), and lexical substitutions assigned (to each target word and sentence pair) by 5 native speaker annotators. The second dataset, *TWSI* (Biemann, 2012)[6], consists of 24,647 sentences for a total of 1,012 target *nouns*, and lexical substitu-

[4] E.g., *bright* substituted with *intelligent* in *"He was bright and independent and proud"* and *side* for *part* in *"Find someone who can compose the biblical side"*.

[5] http://nlp.cs.swarthmore.edu/semeval/tasks/task10/data.shtml

[6] http://www.ukp.tu-darmstadt.de/data/lexical-resources/twsi-lexical-substitutions/

tions for each target word in context resulting from a crowdsourced annotation process.

For each sentence in each dataset, the annotators provided as many substitutions for the target word as they found appropriate in the context. Each substitution is then labeled by the number of annotators who listed that word as a good lexical substitution.

Experimental setup and Evaluation. On both datasets, we conduct experiments using a 10-fold cross validation process, and evaluate all learning algorithms on the same train/test splits. The datasets are randomly split into 10 equal-sized folds on the *target word* level, such that all examples for a particular target word fall into *either the training or the test set, but never both*. This way, we make sure to evaluate the models on target words *not seen during training*, thereby mimicking an open vocabulary paraphrasing system: at testing time, paraphrases are ranked for unseen target words, similarly as the models would rank paraphrases for any words (not necessarily contained in the dataset). For algorithms with tunable parameters, we further divide the training sets into a training and a validation part to find the best parameter settings. For evaluation, we use Generalized Average Precision (GAP) (Kishida, 2005) and Precision at 1 (P@1), i.e., the percentage of correct paraphrases at rank 1.

Features. In all experiments, we used the features described in Szarvas et al. (2013), implemented precisely as proposed by the original work.

Each $(sentence, target\,word, substitution)$ triplet represents an instance, and the feature values are computed from the sentence context, the target word and the substitution word. The features used fall into four major categories.

The most important features describe the syntagmatic coherence of the substitution in context, measured as local n-gram frequencies obtained from web data. The frequency for a 1-5gram context with the substitution word is computed and normalized with respect to either 1) the frequency of the original context (with the target word) or 2) the sum of frequencies observed for all possible substitutions. A third feature computes similar frequencies for the substitution *and* the target word observed in the local context (as part of a conjunctive phrase).

A second group of features describe the (non-positional, i.e. non-local) distributional similarity of the target and its candidate substitution in terms of sentence level co-occurrence statistics collected from newspaper texts: 1) How many words from the sentence appear in the top 1000 salient words listed for the candidate substitution in a distributional thesaurus, 2) how similar the top K salient words lists are for the candidate and the target word, 3) how similar the 2nd order distributional profiles are for candidate and target, etc. All these features are carefully normalized so that values compare well accross different words and contexts.

Another set of features capture the properties of the target and candidate word in WordNet, such as their 1) number of senses, 2) how frequent senses are synonymous and 3) the lowest common ancestor (and all synsets up) for the candidate and target word in the WordNet hierarchy (represented as a nominal feature, by the ID of these synsets).

Lastly a group of features capture shallow syntactic patterns of the target word and its local context in the form of 1) part of speech patterns (trigrams) in a sliding window around the target word using main POS categories, i.e. only the first letter of the Penn Treebank codes, and 2) the detailed POS code of the candidate word assigned by a POS tagger.

We omit a mathematically precise description of these features for space reasons and refer the reader to Szarvas et al. (2013) for a more formal and detailed description of the feature functions. Importantly, these delexicalized features are numerically comparable across the different target words and candidate substitutions they are computed from. This property enables the models to generalize over the words in the datasets and thus enables a supervised, all-words lexical substitution system.

4 Learning-to-Rank methods

Machine learning methods for ranking are traditionally classified into three categories. In the *pointwise* approach, a model is trained that maps instances (in this case candidate substitutions in a context) to scores indicating their relevance or fitness; to this end, one typically applies standard regression techniques, which essentially look at individual instances in isolation (i.e., independent of any other instances in the training or test set). To predict a ranking of a set of query instances, these are simply sorted by their predicted scores (Li et al., 2007).

The *pairwise* approach trains models that are able to compare pairs of instances. By marking such a pair as positive if the first instance is preferred to the second one, and as negative otherwise, the problem can formally be reduced to a binary classification task (Freund et al., 2003). Finally, in the *listwise* approach, tailor-made learning methods are used that directly optimize the ranking performance with respect to a global evaluation metric, i.e., a measure that evaluates the ranking of a complete set of query instances (Valizadegan et al., 2009).

Below we give a brief overview of the methods included in our experiments. We used the implementations provided by the MultiBoost (Benbouzid et al., 2012), RankSVM and RankLib packages.[7] For a detailed description, we refer to the original literature.

4.1 MAXENT

The ranking model proposed by Szarvas et al. (2013) was used as a baseline. This is a pointwise approach based on a maximum entropy classifier, in which the ranking task is cast as a binary classification problem, namely to discriminate good ($label > 0$) from bad substitutions. The actual label values for good substitutions were used for weighting the training examples. The underlying MaxEnt model was trained until convergence, i.e., there was no hyperparameter to be tuned. For a new target/substitution pair, the classifier delivers an estimation of the posterior probability for being a good substitution. The ranking is then produced by sorting the candidates in decreasing order according to this probability.

4.2 EXPENS

EXPENS (Busa-Fekete et al., 2013) is a pointwise method with listwise meta-learning step that exploits an ensemble of multi-class classifiers. It consists of three steps. First, ADABOOST.MH (Schapire and Singer, 1999) classifiers with several different weak learners (Busa-Fekete et al., 2011; Kégl and Busa-Fekete, 2009) are trained to predict the level of relevance (quality) of a substitution (i.e., the number of annotators who proposed the candidate for that particular context). Second, the classifiers are calibrated to obtain an accurate posterior distribution; to this end, several calibration techniques, such as Platt scaling (Platt, 2000), are used to obtain a diverse pool of calibrated classifiers. Note that this step takes advantage of the ordinal structure of the underlying scale of relevance levels, which is an important difference to MAXENT. Third, the posteriors of these calibrated classifiers are additively combined, with the weight of each model being exponentially proportional to its GAP score (on the validation set). This method has two hyperparameters: the number of boosting iterations T and the scaling factor in the exponential weighting scheme c. We select T and c from the intervals $[100, 2000]$ and $[0, 100]$, with step sizes 100 and 10, respectively.

4.3 RANKBOOST

RANKBOOST (Freund et al., 2003) is a pairwise boosting approach. The objective function is the rank loss (as opposed to ADABOOST, which optimizes the exponential loss). In each boosting iteration, the weak classifier is chosen by maximizing the weighted rank loss. For the weak learner, we used the decision stump described in (Freund et al., 2003), which is able to optimize the rank loss in an efficient way. The only hyperparameter of RANKBOOST to be tuned is the number of iterations that we selected from the interval $[1, 1000]$.

4.4 RANKSVM

RANKSVM (Joachims, 2006) is a pairwise method based on support vector machines, which formulates the ranking task as binary classification of pairs of instances. We used a linear kernel, because the optimization using non-linear kernels cannot be done in a reasonable time. The tolerance level of the optimization was set to 0.001 and the regularization parameter was validated in the interval $[10^{-6}, 10^{4}]$ with a logarithmically increasing step size.

4.5 LAMBDAMART

LAMBDAMART (Wu et al., 2010) is a listwise method based on the gradient boosted regression trees by Friedman (1999). The ordinal labels are learned directly by the boosted regression trees whose parameters are tuned by using a gradient-based optimization method. The gradient of parameters is calculated based on the evaluation metric used (in this case GAP). We tuned the number of boosting

[7] RankLib is available at `http://people.cs.umass.edu/~vdang/ranklib.html`. We extended the implementation of the LAMBDAMART algorithm in this package to compute the gradients of and optimize for the GAP measure.

Database	LexSub		TWSI	
Candidates	WN	Gold	WN	Gold
	GAP			
MaxEnt	43.8	52.4	36.6	47.2
ExpEns	44.3	**53.5**	**37.8**	**49.7**
RankBoost	44.0	51.4	37.0	47.8
RankSVM	43.3	51.8	35.5	45.2
LambdaMART	**45.5**	**55.0**	**37.8**	**50.1**
	P@1			
MaxEnt	40.2	57.7	32.4	49.5
ExpEns	39.8	**58.5**	**33.8**	**53.2**
RankBoost	**40.7**	55.2	33.1	50.8
RankSVM	40.3	51.7	33.2	45.1
LambdaMART	**40.8**	**60.2**	**33.1**	**53.6**

Table 1: GAP and p@1 values, with significant improvements over the performance of MaxEnt marked in bold.

System	GAP
Erk and Padó (2010)	38.6
Dinu and Lapata (2010)	42.9
Thater et al. (2010)	46.0
Thater et al. (2011)	51.7
Szarvas et al. (2013)	52.4
EXPENS	53.5
LAMBDAMART	55.0

Table 2: Comparison to previous studies (dataset *LexSub*, candidates *Gold*).

iterations in the interval $[10, 1000]$ and the number of tree leaves in $\{8, 16, 32\}$.

5 Results and discussion

Our results using the above learning methods are summarized in Table 1. As can be seen, the two methods that exploit the cardinal structure of the label set (relevance degrees), namely EXPENS and LAMBDAMART, consistently outperform the baseline taken from Szarvas et al. (2013) – the only exception is the $p@1$ score for EXPENS on the Semeval Lexical Substitution dataset and the candidate substitutions extracted from WordNet. The improvements are significant (using paired t-test, $p < 0.01$) for 3 out of 4 settings for EXPENS and in all settings for LAMBDAMART. In particular, the results of LAMBDAMART are so far the best scores that have been reported for the best studied setting, i.e. the LexSub dataset using substitution candidates taken from the gold standard (see Table 2).

We suppose that the relatively good results achieved by the LAMBDAMART and EXPENS methods are due to that, first, it seems crucial to properly model and exploit the ordinal nature of the annotations (number of annotators who suggested a given word as a good paraphrase) provided by the datasets. Second, the RANKBOOST and RANKSVM are less complex methods than the EXPENS and LAMBDAMART. The RANKSVM is the least complex method from the pool of learning-to-rank methods we applied, since it is a simple linear model. The RANKBOOST is a boosted decision *stump* where, in each boosting iteration, the stump is found by maximizing the weighted exponential rank loss. On the other hand, both the EXPENS and LAMBDAMART make use of *tree* learners in the ensemble classifier they produce. We believe that overfitting is not an issue in a learning task like the LexSub task: most features are relatively weak predictors on their own, and we can learn from a large number of data points (2000 sentences with an average set size of 20, about 40K data points for the smallest dataset and setting). Rather, as our results show, less complex models tend to underfit the data. Therefore we believe that more complex models can achieve a better performance, of course with an increased computational cost.

6 Conclusion and future work

In this paper, we customized and applied some relatively novel algorithms from the field of learning-to-rank for ranking lexical substitutions in context. In turn, we achieved significant improvements on the two prominent datasets for lexical substitution.

Our results indicate that an exploitation of the ordinal structure of the labels in the datasets can lead to considerable gains in terms of both ranking quality (GAP) and precision at 1 (p@1). This observation is supported both for the theoretically simpler pointwise learning approach and for the most powerful listwise approach. On the other hand, the pairwise methods that cannot naturally exploit this property, did not provide a consistent improvement over the baseline. In the future, we plan to investigate this finding in the context of other, similar ranking problems in Natural Language Processing.

Acknowledgment

This work was supported by the German Research Foundation (DFG) as part of the Priority Programme 1527.

References

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*, pages 385–393, Montréal, Canada.

D. Benbouzid, R. Busa-Fekete, N. Casagrande, F.-D. Collin, and B. Kégl. 2012. MultiBoost: a multi-purpose boosting package. *Journal of Machine Learning Research*, 13:549–553.

Chris Biemann. 2012. Creating a System for Lexical Substitutions from Scratch using Crowdsourcing. *Language Resources and Evaluation: Special Issue on Collaboratively Constructed Language Resources*, 46(2).

R. Busa-Fekete, B. Kégl, T. Éltető, and Gy. Szarvas. 2011. Ranking by calibrated AdaBoost. In *(JMLR W&CP)*, volume 14, pages 37–48.

R. Busa-Fekete, B. Kégl, T. Éltető, and Gy. Szarvas. 2013. Tune and mix: learning to rank using ensembles of calibrated multi-class classifiers. *Machine Learning*, 93(2–3):261–292.

Ching-Yun Chang and Stephen Clark. 2010. Practical linguistic steganography using contextual synonym substitution and vertex colour coding. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1194–1203, Cambridge, MA.

Ido Dagan, Oren Glickman, Alfio Gliozzo, Efrat Marmorshtein, and Carlo Strapparava. 2006. Direct word sense matching for lexical substitution. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, ACL-44, pages 449–456, Sydney, Australia.

Georgiana Dinu and Mirella Lapata. 2010. Measuring distributional similarity in context. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 1162–1172, Cambridge, MA.

Katrin Erk and Sebastian Padó. 2010. Exemplar-based models for word meaning in context. In *Proceedings of the ACL 2010 Conference Short Papers*, pages 92–97, Uppsala, Sweden.

Y. Freund, R. Iyer, R. E. Schapire, and Y. Singer. 2003. An efficient boosting algorithm for combining preferences. *Journal of Machine Learning Research*, 4:933–969.

J. Friedman. 1999. Greedy function approximation: a gradient boosting machine. Technical report, Dept. of Statistics, Stanford University.

Bela Gipp, Norman Meuschke, and Joeran Beel. 2011. Comparative Evaluation of Text- and Citation-based Plagiarism Detection Approaches using GuttenPlag. In *Proceedings of 11th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL'11)*, pages 255–258, Ottawa, Canada. ACM New York, NY, USA. Available at http://sciplore.org/pub/.

Claudio Giuliano, Alfio Gliozzo, and Carlo Strapparava. 2007. FBK-irst: Lexical substitution task exploiting domain and syntagmatic coherence. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 145–148, Prague, Czech Republic.

Samer Hassan, Andras Csomai, Carmen Banea, Ravi Sinha, and Rada Mihalcea. 2007. UNT: SubFinder: Combining knowledge sources for automatic lexical substitution. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 410–413, Prague, Czech Republic.

T. Joachims. 2006. Training linear svms in linear time. In *Proceedings of the ACM Conference on Knowledge Discovery and Data Mining (KDD)*.

B. Kégl and R. Busa-Fekete. 2009. Boosting products of base classifiers. In *International Conference on Machine Learning*, volume 26, pages 497–504, Montreal, Canada.

Kazuaki Kishida. 2005. *Property of Average Precision and Its Generalization: An Examination of Evaluation Indicator for Information Retrieval Experiments*. NII technical report. National Institute of Informatics.

P. Li, C. Burges, and Q. Wu. 2007. McRank: Learning to rank using multiple classification and gradient boosting. In *Advances in Neural Information Processing Systems*, volume 19, pages 897–904. The MIT Press.

David Martinez, Su Nam Kim, and Timothy Baldwin. 2007. MELB-MKB: Lexical substitution system based on relatives in context. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 237–240, Prague, Czech Republic.

Diana McCarthy and Roberto Navigli. 2007. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 48–53, Prague, Czech Republic.

J. Platt. 2000. Probabilistic outputs for support vector machines and comparison to regularized likelihood methods. In A.J. Smola, P. Bartlett, B. Schoelkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press.

R. E. Schapire and Y. Singer. 1999. Improved boosting algorithms using confidence-rated predictions. *Machine Learning*, 37(3):297–336.

Ravi Sinha and Rada Mihalcea. 2009. Combining lexical resources for contextual synonym expansion. In *Proceedings of the International Conference RANLP-2009*, pages 404–410, Borovets, Bulgaria.

György Szarvas, Chris Biemann, and Iryna Gurevych. 2013. Supervised all-words lexical substitution using delexicalized features. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT 2013)*, June.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2010. Contextualizing semantic representations using syntactically enriched vector models. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 948–957, Uppsala, Sweden.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2011. Word meaning in context: A simple and effective vector model. In *Proceedings of the Fifth International Joint Conference on Natural Language Processing : IJCNLP 2011*, pages 1134–1143, Chiang Mai, Thailand. MP, ISSN 978-974-466-564-5.

Umut Topkara, Mercan Topkara, and Mikhail J. Atallah. 2006. The hiding virtues of ambiguity: quantifiably resilient watermarking of natural language text through synonym substitutions. In *Proceedings of the 8th workshop on Multimedia and security*, pages 164–174, New York, NY, USA. ACM.

H. Valizadegan, R. Jin, R. Zhang, and J. Mao. 2009. Learning to rank by optimizing NDCG measure. In *Advances in Neural Information Processing Systems 22*, pages 1883–1891.

Q. Wu, C. J. C. Burges, K. M. Svore, and J. Gao. 2010. Adapting boosting for information retrieval measures. *Inf. Retr.*, 13(3):254–270.

Deniz Yuret. 2007. Ku: Word sense disambiguation by substitution. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*, pages 207–214, Prague, Czech Republic, June. Association for Computational Linguistics.