# Character-level Convolutional Neural Network for Paraphrase Detection and other Experiments

Vladislav Maraev, Chakaveh Saedi, João Rodrigues, António Branco, and João Silva

Department of Informatics, Faculty of Sciences, University of Lisbon, Portugal
{vlad.maraev,chakaveh.saedi,joao.rodrigues,antonio.branco,jsilva}@di.fc.ul.pt

**Abstract** The central goal of this paper is to report on the results of an experimental study on the application of character-level embeddings and basic convolutional neural network to the shared task of sentence paraphrase detection in Russian. This approach was tested in the standard run of Task 2 of that shared task and revealed competitive results, namely 73.9% accuracy against the test set. It is compared against a word-level convolutional neural network for the same task, and varied other approaches, such as rule-based and classical machine learning.

**Keywords:** paraphrase detection, word embeddings, character embeddings, convolutional neural networks, distributional semantics

## 1 Introduction

The Russian language is a morphologically rich language with free word order and can be an interesting workbench for testing different models of paraphrase detection, which have been studied mostly against English datasets.

In this paper, we report on addressing this task by using a system that we developed and showed competitive results in the standard run Task 2 of Russian paraphrase detection shared task [11], where participating systems cannot resort to data other than the ones provided for the shared task. This system is based on a character-based convolutional neural network.

We report also on the results obtained with the application of other approaches that we developed and tested initially for the task of duplicate question detection [13, 15].

Paraphrase detection belongs to a family of semantic text similarity tasks, which have been addressed in SemEval challenges since 2012, and which in the last SemEval-2016, for instance, included also tasks like the degree of similarity between machine translation output and its post-edited version, among others.

Semantic textual similarity assesses the degree to which two textual segments are semantically equivalent to each other, which is typically scored on an ordinal scale ranging from semantic equivalence to complete semantic dissimilarity.

Paraphrase detection is a special case of semantic textual similarity, where the scoring scale is reduced to its two extremes and the outcome for an input pair of textual segments is yes/no.

The present paper is organized as follows. In the next Section 2, the conditions of and the results for the shared task are discussed. The character-level convolutional neural network and respective results are discussed in Section 3. Sections 4, 5 and 6 present the experimental results of a range of other approaches, respectively, rule-based, supervised classifiers and other deep neural networks. In Section 7, the results obtained are discussed. Sections 8 and 9 discuss the related work and present the conclusions.

## 2 Dataset and results of participation

For the experimental results reported in the present paper, we resorted to the shared task's ParaPhraser dataset [12], a freely available corpus of Russian sentence pairs manually annotated as precise paraphrases, near-paraphrases and non-paraphrases. Each pair was collected from news headlines and then manually annotated by three native speakers.

The size of the training set is 7,000 pairs and the test set contains 1,924 pairs. The number of tokens, the number of types and average sentence length in the training set are presented in the Table 1.

| | |
|---|---|
| Pairs | 7,000 |
| Total tokens | 126,303 |
| Lowercased types | 20,252 |
| Average sentence length (words) | 8.7 |

**Table 1.** Quantitative attributes of the training set.

The shared task consists of two subtasks: one for three-class classification, and another for binary classification. We have tackled the second one (Task 2) which is defined as follows:

> *Given a pair of sentences, to predict whether they are paraphrases (whether precise or near paraphrases) or non-paraphrases.*

There were two types of shared settings: the *standard run* where only the ParaPhraser corpus could be used for training, and the *non-standard run* where any other corpora could be also used. We participated in both types of submissions.

According to the results obtained by submitting the output to the shared task organisation: (i) our system *CNN-char*, which participated in standard run obtained a competitive accuracy score of 72.7%, which stands just 1.9 percentage points below the best system's score; (ii) our system *CNN-word*, which participated in the non-standard run obtained an accuracy score of 69.9%, which is quite lower than the best system's accuracy of 77.4%.

Below we will discuss also the results obtained a posteriori in our lab once the test sets were released, which are slightly different from the ones above reported

by the shared task organization, due to the random initialization of the weights of the neural network.

## 3 Convolutional neural network

The architecture of convolutional neural network (CNN) used to address the paraphrase detection task was introduced by Bogdanova et al. [3] for the task of detecting semantically equivalent questions in online question answering forums. It also takes advantage of the approach introduced by Kim [7] for sentence classification task using a set of convolutional filters of an arbitrary length.

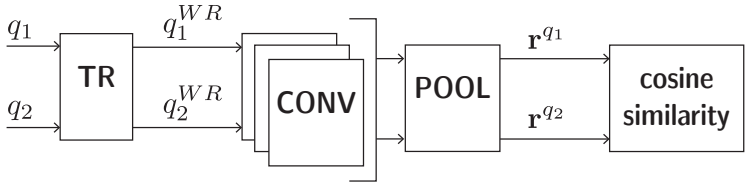


**Figure 1.** CNN architecture.

Figure 1 shows the layers of the CNN: token (word or character) representation layer (TR), convolution layer(s) (CONV), pooling layer (POOL) and cosine similarity measurement step.

To obtain the representation of a sentence, it is pipelined along these major steps:

1. Obtaining token representations;
2. Applying a set of convolutional filters;
3. Concatenating the results of convolution;
4. Pooling the product of convolution filters.

We resort to two variants[1] for paraphrase detection using a convolutional neural network.

The first one uses randomly initialized character representations on a token representation layer that are further passed as input to a set of convolutional filters.

The second one follows Bogdanova et al. [3] and relies on pre-trained word embeddings for the initial token representation.

[1] Source code is available as a part of Vladislav Maraev's MA dissertation at: https://github.com/vladmaraev/msrdsdl

### 3.1 Character embeddings

In the first variant, referred to as *CNN-char*, we split sentences into characters instead of tokenizing them into words. The main reason to have followed this route is that character-level embeddings are reported to be good in capturing morphological information [16, 8], which is important for a morphologically rich language like Russian.

In terms of preprocessing, a few basic procedures were applied, namely, lowercasing the input and removing non-word characters.

Table 2 summarizes the hyper-parameters that were used for this run.

| *Parameter* | *Value* | *Description* |
|---|---|---|
| $k$ | $\{2, 3, 5, 7, 9, 11\}$ | Sizes of $k$-grams |
| $l_u$ | 100 | Size of each convolutional filter |
| $d$ | 100 | Size of character representation |
| epochs | 20 | Number of training epochs |
| pooling | MAX | Pooling layer function |
| optimizer | SGD | Stochastic Gradient Descent |
| loss | MSE | Mean Squared Error |

**Table 2.** Hyper-parameters of CNN-char.

**Results** This approach leads to the highest accuracy of 73.9%, reported in this work regarding Russian paraphrase detection task.

### 3.2 Pre-trained word embeddings

In this other variant, referred to as *CNN-word*, the approach adopted by Bogdanova et al. [3] for the task of duplicate question detection was followed here for paraphrase detection, where word embeddings were pre-trained.

We employed word2vec word embeddings from Kutuzov and Andreev [9].[2]

In order to preprocess the input sentences, these were lowercased, lemmatised and PoS-tagged using MyStem [17], which is the same tool that was reported by the authors of RusVectores model [9].

The Table 3 summarizes the hyper-parameters that were used for this run.

**Results** This variant leads to an accuracy score of 70.6%, which is 3.3 pp. lower than the score obtained by the character-based model in spite of the usage of external resources.

[2] These word embeddings for Russian are available from: http://rusvectores.org/ru/models/, *ruscorpora_2015* model

| *Parameter* | *Value* | *Description* |
|---|---|---|
| $k$ | 3 | Size of $k$-gram |
| $l_u$ | 300 | Size of convolutional filter |
| $d$ | 300 | Size of word representation |
| epochs | 5 | Number of training epochs |
| pooling | MAX | Pooling layer function |
| optimizer | SGD | Stochastic Gradient Descent |
| loss | MSE | Mean Squared Error |

**Table 3.** Hyper-parameters of CNN-word.

## 4 Rule-based

A rule-based approach, referred to as *Jaccard*, was used to establish a baseline. We used the Jaccard Coefficient over $n$-grams ($n$ ranging from 1 to 4), inspired by the usage of this coefficient in [18].

Before applying this technique, the textual segments were preprocessed by submitting them to lowercasing, tokenization and lemmatisation using the MyStem tool [17].

To find the best threshold, the training set was used in a series of trials. This led to the thresholds of 0.13 for the English dataset, and 0.1 for the Russian dataset.

**Results** This system achieves the accuracy score of 67.0%. This result is lower than ones obtained by *CNN-char* and described above. It is in line tough with the scores obtained in other experiments that were carried out for another task, namely duplicate question detection [13, 15].

## 5 Classic machine learning approaches

### 5.1 SVM with basic features

To set up a paraphrase detection system based on a supervised machine learning classifier, we resorted to support vector machines (SVM), following its acknowledged good performance in many NLP tasks. We employed SVC (Support Vector Classification) implementation from the sklearn support vector machine toolkit [10].

For the first version of the classifier, a basic feature set ($FS$) was created. $N$-grams, with $n$ ranging from 1 to 4, were extracted from the training set. Afterwards, among those extracted n-grams, the ones with at least 10 occurrences were selected to support the $FS$. We tried thresholds ranging from 5 to 15 and the best result was achieved when the threshold was set to 10.

For each textual segment in a pair, a vector of size $k$ was generated, where $k$ is the number of $n$-grams included in the $FS$. Each vector encodes the occurrences of the $n$-grams in the corresponding segment, where vector position $i$ will be 1 if

the $i$-th $n$-gram occurs in the segment, and 0 otherwise. Then a feature vector of size $2k$ is created by concatenating the vectors of the two segments. This vector is further extended with the scores of the Jaccard coefficient determined over 1, 2, 3 and 4-grams. Hence, the final feature vector representing the pair to the classifier has the length $2k + 4$.

**Results** This system achieves 70.4% accuracy when trained over the Russian dataset, which suggests that the result is comparable with *CNN-word* that also uses external language resources.

### 5.2 SVM classifier with advanced features

In order to get an insight on how strong an SVM-based system for paraphrase detection resorting to a basic $FS$ like the one described above may be, we proceeded with further experiments, by adding more advanced features.

**Lexical features** The vector of each segment was extended with an extra feature, namely the number of negative words, e.g.: *ничего* ("nothing"), *никогда* ("never"), etc. occurring in it. And, to the concatenation of segment vectors, one further feature was added, the number of nouns that are common to both segments, provided they are not already included in the $FS$. Any pair was then represented by a vector of size $2(k + 1) + 4 + 1$.

**Semantic features** Eventually, any pair was represented by a vector of size $2(k+1)+4+2$, with its length being extended with yet an extra feature, namely the value of the cosine similarity between the embeddings of the segments in the pair.

For a given segment, its embedding, or distributional semantic vector, was obtained by summing up the embeddings of the nouns and verbs occurring in it, as these showed to support the best performance after experiments that have been undertaken with all parts-of-speech and their subsets. We employed word2vec word embeddings from Kutuzov and Andreev [9] the same ones that we used in the experiment discussed in Section 3.2.

**Results** The resulting system permitted an improvement of over 1 percentage points with respect to its previous version trained with basic features, scoring 71.7% accuracy, thus being slightly superior to our *CNN-word* system above, with pre-trained word embeddings.

## 6 Deep neural network architectures

In this section we discuss the experiments that were carried out in order to assess the performance, in the paraphrase detection task, of the deep neural network

architectures that were able to achieve very high performance in the duplicate question detection task [13, 15].

We begin by applying the architecture of MayoNLP, the system that was the top scoring system in SemEval-2016 Task 1 [2]. We will then proceed with discussing a hybrid approach that combines convolutional and fully-connected layers in a neural network.

The same preprocessing used on the convolutional neural networks (lower-casing, lemmatization, and PoS-tagged) was used in these models.

### 6.1 Deep Neural Network (MayoNLP)

We implemented a deep neural network (DNN) based on MayoNLP [1]. This system follows the architecture of Deep Structured Semantic Models, introduced by Huang et al. [5], which consists of a multi-layer neural architecture of feed-forward and fully connected layers. The neural network has as input a 30k neurons dense layer followed by two hidden multi-layers with 300 neurons each and finally a 128 neuron output layer.

MayoNLP also implemented a preprocessing dimension reduction with a word hashing method which creates trigrams for every word in the input sentence.

Given that we did not face the same dimension problem, we implemented a one-hot encoding process, which eventually ended up reducing even further, from an original 30k dimension in Mayo to 10k for the ParaPhrase dataset.

The MayoNLP system also differs from the Deep Structured Semantic Models by adopting a 1k neuron layer instead of two hidden layers in its architecture.

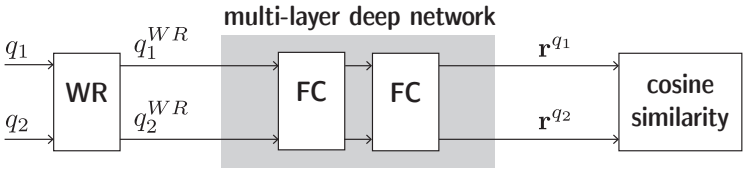


**Figure 2.** DNN architecture: word representation layer (WR), fully connected layers (FC) and cosine similarity measurement layer.

A diagram of the implemented neural network is presented in Figure 2. The Table 4 summarizes the hyper-parameters that were used.

**Results** The model obtained a 59.9% accuracy, scoring the worst result in comparison with the results of the models experimented and reported in this paper. This is mainly due to the lack of sufficient data and the overwhelming complexity of the neural network for the given dataset.

| *Parameter* | *Value* | *Description* |
|---|---|---|
| lr | 0.01 | Learning rate |
| hidden neurons | 728 | Hidden layer neurons |
| epochs | 20 | Training epochs |
| pooling | MAX | Pooling layer function |
| optimizer | SGD | Stochastic Gradient Descent |
| loss | MSE | Mean Squared Error |

**Table 4.** DNN-word approach hyper-parameters.

### 6.2 Deep convolutional neural network

Finally, we also experimented with a deep convolutional neural network (DCNN) model with which we obtained the best accuracies in a related semantic similarity task [13]. This model is a combination of the convoluted and dense models previously described. A lite version of the original model was deployed given the decrease in the available dataset when compared with the originally designed dataset. We resorted to Keras and Tensorflow for its implementation.

Both input sentences are fed to the neural network, both pass the same neural network layers in parallel and are compared before the output result, in a so-called Siamese architecture.

A vectorial representation for words is used at the beginning of the model with a layer that acts as a distributional semantic space and learns a vector for each word in the training dataset.

That vectorial representation is fed to a convolutional layer with 50 neurons and a window with size 15.

This convolutional layer is then combined with a pooling layer that resorts to a max filter.

With the resulting vector of the pooling layer the network connects to three dense layers of fully connected layers with 15 neurons each.

In a final step, the output of the layers is then computed by means of the cosine distance between the result of both inputs.

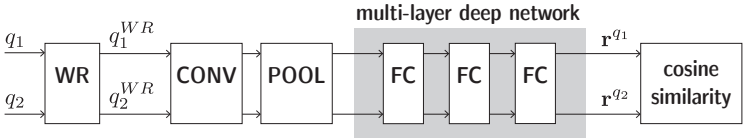


**Figure 3.** DCNN architecture.

A diagram of this hybrid neural network is presented in Figure 3. The Table 5 summarizes the hyper-parameters that were used.

| *Parameter* | *Value* | *Description* |
|---|---|---|
| lr | 0.01 | Learning rate |
| epochs | 20 | Training epochs |
| $d$ | 50 | Size of word representation |
| $l_u$ | 50 | Size of convolutional filter |
| $k$ | 5 | Size of convolutional kernel |
| hidden neurons | 45 | Hidden layer neurons |
| pooling | MAX | Pooling layer function |
| optimizer | SGD | Stochastic Gradient Descent |
| loss | MSE | Mean Squared Error |

**Table 5.** DCNN-word approach hyper-parameters.

**Results** The DCNN model obtained 70.0 % accuracy, which is in line with the results of other models such as SVM and Jaccard. This is mainly due to it being a lite version of the original neural network. As it is common with neural networks, the more data the better, which makes us believe higher accuracies can be obtained with a larger dataset.

## 7 Discussion

The experimental results reported in the previous sections are summarized in Table 6.

| System | Accuracy (%) |
|---|---|
| majority class | 49.7 |
| Jaccard* | 67.0 |
| SVM-bas* | 70.4 |
| CNN-word* | 70.6 |
| SVM-adv* | 71.7 |
| DNN | 59.9 |
| DCNN | 70.0 |
| CNN-char | 73.9 |
| Best system in shared task* | 77.4 |
| Best system in shared task | 74.6 |

**Table 6.** Accuracy of the 7 systems plus the majority class baseline over the Russian paraphrases dataset.

In this table, the star (*) superscript indicates systems that use resources other than just the ParaPhraser dataset distributed by the shared task organizers.

At the bottom of the table, the best results obtained by systems that participated in the shared task are displayed.

## 8 Related work

The best three systems in the SemEval-2016 Task 1 are the following: Rychalska et al. [14], which employs autoencoders, WordNet and SVM; Brychcín and Svoboda [4], which combines various meaning representation algorithms and different classifiers; and the MayoNLP system [1], whose architecture is adopted in one of our experiments and was presented in Section 6.1.

The competitor non-NN-based system [6] uses discriminative term-weighting (TF-KLD) and matrix factorisation.

The work on CNNs reported in this paper was inspired by the work of Bogdanova et al. [3] that employ Siamese CNN with shared weights for detecting semantically equivalent question. It also takes advantage of the approaches introduced by Kim [7] for concatenating convolutional filters of various lengths and [8] for employing character embedding for morphologically rich languages.

## 9 Conclusions

This paper has presented the results of a range of experiments to address the task of paraphrase detection for Russian under the conditions and with the datasets of the respective shared task organized in 2016.

The application of the convolutional neural network model to this task showed the best results. In particular, the character-based convolutional neural network model achieves competitive performance for the task of detecting if two sentences are paraphrases without using any external resources.

## Acknowledgements

The present research was also partly supported by the CLARIN and ANI/3279/2016 grants.

# Bibliography


[1] Afzal, N., Wang, Y., Liu, H.: MayoNLP at SemEval-2016 task 1: Semantic textual similarity based on lexical semantic net and deep learning semantic model. In: Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016). pp. 1258–1263 (2016)

[2] Agirre, E., Banea, C., Cer, D.M., Diab, M.T., Gonzalez-Agirre, A., Mihalcea, R., Rigau, G., Wiebe, J.: SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In: Bethard, S., Cer, D.M., Carpuat, M., Jurgens, D., Nakov, P., Zesch, T. (eds.) Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016, San Diego, CA, USA, June 16–17, 2016. pp. 497–511. The Association for Computer Linguistics (2016), http://aclweb.org/anthology/S/S16/S16-1081.pdf

[3] Bogdanova, D., dos Santos, C.N., Barbosa, L., Zadrozny, B.: Detecting semantically equivalent questions in online user forums. In: Alishahi, A., Moschitti, A. (eds.) Proceedings of the 19th Conference on Computational Natural Language Learning, CoNLL 2015, Beijing, China, July 30-31, 2015. pp. 123–131. ACL (2015), http://aclweb.org/anthology/K/K15/K15-1013.pdf

[4] Brychcín, T., Svoboda, L.: UWB at SemEval-2016 task 1: Semantic textual similarity using lexical, syntactic, and semantic information. In: Bethard, S., Cer, D.M., Carpuat, M., Jurgens, D., Nakov, P., Zesch, T. (eds.) Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016. pp. 588–594. The Association for Computer Linguistics (2016), http://aclweb.org/anthology/S/S16/S16-1089.pdf

[5] Huang, P.S., He, X., Gao, J., Deng, L., Acero, A., Heck, L.: Learning deep structured semantic models for web search using clickthrough data. In: Proceedings of the 22nd ACM International Conference on Information & Knowledge Management. pp. 2333–2338. ACM (2013)

[6] Ji, Y., Eisenstein, J.: Discriminative improvements to distributional sentence similarity. In: Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL. pp. 891–896. ACL (2013), http://aclweb.org/anthology/D/D13/D13-1090.pdf

[7] Kim, Y.: Convolutional neural networks for sentence classification. In: Moschitti, A., Pang, B., Daelemans, W. (eds.) Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL. pp. 1746–1751. ACL (2014), http://aclweb.org/anthology/D/D14/D14-1181.pdf

[8] Kim, Y., Jernite, Y., Sontag, D., Rush, A.M.: Character-aware neural language models. In: Schuurmans, D., Wellman, M.P. (eds.) Proceedings of

the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA. pp. 2741–2749. AAAI Press (2016), http://www.aaai.org/ocs/index.php/AAAI/AAAI16/paper/view/12489
[9] Kutuzov, A., Andreev, I.: Texts in, meaning out: Neural language models in semantic similarity tasks for russian. vol. 2, pp. 133–144 (2015)
[10] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research 12, 2825–2830 (2011)
[11] Pivovarova, L., Pronoza, E., Yagunova, E., Pronoza, A.: Russian paraphrase corpus and shared task. In: CCIS 789. pp. ??–?? (this volume)
[12] Pronoza, E., Yagunova, E., Pronoza, A.: Construction of a russian paraphrase corpus: unsupervised paraphrase extraction. In: Information Retrieval, pp. 146–157. Springer (2016)
[13] Rodrigues, J.A., Saedi, C., Maraev, V., Silva, J., Branco, A.: Ways of asking and replying in duplicate question detection. In: Ide, N., Herbelot, A., Màrquez, L. (eds.) Proceedings of the 6th Joint Conference on Lexical and Computational Semantics, *SEM @ACM 2017, Vancouver, Canada, August 3-4, 2017. pp. 262–270. Association for Computational Linguistics (2017), https://doi.org/10.18653/v1/S17-1030
[14] Rychalska, B., Pakulska, K., Chodorowska, K., Walczak, W., Andruszkiewicz, P.: Samsung poland NLP team at semeval-2016 task 1: Necessity for diversity; combining recursive autoencoders, wordnet and ensemble methods to measure semantic similarity. In: Bethard, S., Cer, D.M., Carpuat, M., Jurgens, D., Nakov, P., Zesch, T. (eds.) Proceedings of the 10th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2016, San Diego, CA, USA, June 16-17, 2016. pp. 602–608. The Association for Computer Linguistics (2016), http://aclweb.org/anthology/S/S16/S16-1091.pdf
[15] Saedi, C., Rodrigues, J., Silva, J., Branco, A., Maraev, V.: Learning profiles in duplicate question detection. In: Proceedings of IEEE IRI 2017 conference (in press)
[16] dos Santos, C.N., Gatti, M.: Deep convolutional neural networks for sentiment analysis of short texts. In: Hajic, J., Tsujii, J. (eds.) COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland. pp. 69–78. ACL (2014), http://aclweb.org/anthology/C/C14/C14-1008.pdf
[17] Segalovich, I.: A fast morphological algorithm with unknown word guessing induced by a dictionary for a web search engine. In: Arabnia, H.R., Kozerenko, E.B. (eds.) Proceedings of the International Conference on Machine Learning; Models, Technologies and Applications. MLMTA'03, June 23 - 26, 2003, Las Vegas, Nevada, USA. pp. 273–280. CSREA Press (2003)
[18] Wu, Y., Zhang, Q., Huang, X.: Efficient near-duplicate detection for Q&A forum. In: Fifth International Joint Conference on Natural Language Processing, IJCNLP 2011, Chiang Mai, Thailand, November 8–13,

2011. pp. 1001–1009. The Association for Computer Linguistics (2011), http://aclweb.org/anthology/I/I11/I11-1112.pdf