ReVal: A Simple and Effective Machine Translation Evaluation Metric Based on Recurrent Neural Networks

Rohit Gupta[1], **Constantin Orăsan**[1], **Josef van Genabith**[2]
[1]Research Group in Computational Linguistics, University of Wolverhampton, UK
[2]Saarland University and German Research Center for Artificial Intelligence (DFKI), Germany
{r.gupta, c.orasan}@wlv.ac.uk
josef.van_genabith@dfki.de

Abstract

Many state-of-the-art Machine Translation (MT) evaluation metrics are complex, involve extensive external resources (e.g. for paraphrasing) and require tuning to achieve best results. We present a simple alternative approach based on dense vector spaces and recurrent neural networks (RNNs), in particular Long Short Term Memory (LSTM) networks. For WMT-14, our new metric scores best for two out of five language pairs, and overall best and second best on all language pairs, using Spearman and Pearson correlation, respectively. We also show how training data is computed automatically from WMT ranks data.

1 Introduction

Deep learning approaches have turned out to be successful in many NLP applications such as paraphrasing (Mikolov et al., 2013b; Socher et al., 2011), sentiment analysis (Socher et al., 2013b), parsing (Socher et al., 2013a) and machine translation (Mikolov et al., 2013a). While dense vector space representations such as those obtained through Deep Neural Networks (DNNs) or RNNs are able to capture semantic similarity for words (Mikolov et al., 2013b), segments (Socher et al., 2011) and documents (Le and Mikolov, 2014) naturally, traditional MT evaluation metrics can only achieve this using resources like WordNet and paraphrase databases. This paper presents a novel, efficient and compact MT evaluation measure based on RNNs. Our metric is simple in the sense that it does not require much machinery and resources apart from the dense word vectors. This cannot be said of most of the state-of-the-art MT evaluation metrics, which tend to be complex and require extensive feature engineering. Our metric is based on RNNs and particularly on Tree Long Short Term Memory (Tree-LSTM) networks (Tai et al., 2015). LSTM (Hochreiter and Schmidhuber, 1997) is a sequence learning technique which uses a memory cell to preserve a state over a long period of time. This enables distributed representations of sentences using distributed representations of words. Tree-LSTM is a recent approach, which is an extension of the simple LSTM framework (Zaremba and Sutskever, 2014). To provide the required training data, we also show how to automatically convert the WMT-13 (Bojar et al., 2013) human evaluation rankings into similarity scores between the reference and the translation. Our metric including training data is available at https://github.com/rohitguptacs/ReVal.

2 Related Work

Many metrics have been proposed for MT evaluation. Earlier popular metrics are based on n-gram counts (e.g. BLEU (Papineni et al., 2002) and NIST (Doddington, 2002)) or word error rate. Other popular metrics like METEOR (Denkowski and Lavie, 2014) and TERp (Snover et al., 2008) also use external resources like WordNet and paraphrase databases. However, system-level correlation with human judgements for these metrics remains below 0.90 Pearson correlation coefficient (as per WMT-14 results, BLEU-0.888, NIST-0.867, METEOR-0.829, TER-0.826, WER-0.821).

Recent best-performing metrics in the WMT-14 metric shared task (Machácek and Bojar, 2014) used a combination of different metrics. The top performing system DiskoTK-Party-Tuned (Joty et al., 2014) in the WMT-14 task uses five different discourse metrics and twelve different metrics from the ASIYA MT evaluation toolkit (Giménez and Màrquez, 2010). The metric computes the number of common sub-trees between a reference and a translation using a convolution

Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, pages 1066–1072,
Lisbon, Portugal, 17-21 September 2015.

tree kernel (Collins and Duffy, 2001). The basic version of the metric does not perform well but in combination with the other 12 metrics from the ASIYA toolkit obtained the best results for the WMT-14 metric shared task. Another top performing metric LAYERED (Gautam and Bhattacharyya, 2014), uses linear interpolation of different metrics. LAYERED uses BLEU and TER to capture lexical similarity, Hamming score and Kendall Tau Distance (Birch and Osborne, 2011) to identify syntactic similarity, and dependency parsing (De Marneffe et al., 2006) and the Universal Networking Language[1] for semantic similarity. Recently, Guzmán et al. (2015) presented a metric based on word embeddings and neural networks. However, this metric is limited to ranking the available systems and does not provide an absolute score.

In this paper we propose a compact MT evaluation metric. We hypothesize that our model learns different notions of similarity (which other metrics tend to capture using different metrics) using input, output and forget gates of an LSTM architecture.

3 LSTMs and Tree-LSTMs

Recurrent Neural Networks allow processing of arbitrary length sequences, but early RNNs had the problem of vanishing and exploding gradients (Bengio et al., 1994). RNNs with LSTM (Hochreiter and Schmidhuber, 1997) tackle this problem by introducing a memory cell composed of a unit called constant error carousel (CEC) with multiplicative input and output gate units. Input gates protect against irrelevant inputs and output gates against current irrelevant memory contents. This architecture is capable of capturing important pieces of information seen in a bigger context. Tree-LSTM is an extension of simple LSTM. A typical LSTM processes the information sequentially whereas Tree-LSTM architectures enable sentence representation through a syntactic structure. Equation (1) represents the composition of a hidden state vector for an LSTM architecture. For a simple LSTM, c_t represents the memory cell and o_t the output gate at time step t in a sequence. For Tree-LSTM, c_t represents the memory cell and o_t represents the output gate corresponding to node t in a tree. The structural processing of Tree-LSTM makes it better suited to representing sentences. For example, dependency tree structure captures syntactic features and model parameters the importance of words (content vs. function words).

$$h_t = o_t \odot \tanh c_t \tag{1}$$

Figure 1 shows simple LSTM and Tree-LSTM architectures.

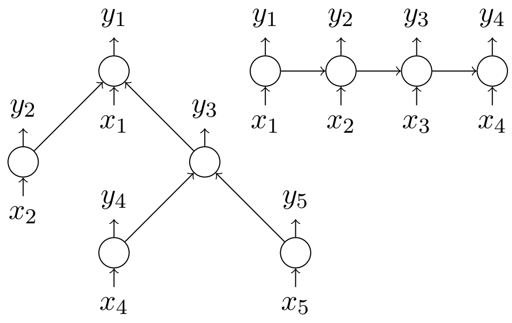

Figure 1: Tree-LSTM (left) and simple LSTM (right)

4 Evaluation Metric

We represent both the reference (h_{ref}) and the translation (h_{tra}) using an LSTM and predict the similarity score $\hat{y}$ based on a neural network which considers both distance and angle between h_{ref} and h_{tra}:

$$\begin{aligned} h_\times &= h_{ref} \odot h_{tra} \\ h_+ &= |h_{ref} - h_{tra}| \\ h_s &= \sigma\left(W^{(\times)} h_\times + W^{(+)} h_+ + b^{(h)}\right) \\ \hat{p}_\theta &= \text{softmax}\left(W^{(p)} h_s + b^{(p)}\right) \\ \hat{y} &= r^T \hat{p}_\theta \end{aligned} \tag{2}$$

where, σ is a sigmoid function, $\hat{p}_\theta$ is the estimated probability distribution vector and $r^T = [1\ 2 ... K]$. The cost function $J(\theta)$ is defined over probability distributions p and $\hat{p}_\theta$ using regularised Kullback-Leibler (KL) divergence.

$$J(\theta) = \frac{1}{n} \sum_{i=1}^{n} \text{KL}\left(p^{(i)} \middle\| \hat{p}_\theta^{(i)}\right) + \frac{\lambda}{2} ||\theta||_2^2 \tag{3}$$

In Equation 3, i represents the index of each training pair, n is the number of training pairs and p is the sparse target distribution such that $y = r^T p$ is defined as follows:

$$p_j = \begin{cases} y - \lfloor y \rfloor, & j = \lfloor y \rfloor + 1 \\ \lfloor y \rfloor - y + 1, & j = \lfloor y \rfloor \\ 0 & \text{otherwise} \end{cases}$$

[1] http://www.undl.org/unlsys/unl/unl2005/UW.htm

for $1 \leq j \leq K$, where, $y \in [1, K]$ is the similarity score of a training pair. For example, for $y = 2.7$, $p^T = [0\ 0.3\ 0.7\ 0\ 0]$. In our case, the similarity score y is a value between 1 and 5.

For our work, we use *glove* word vectors (Pennington et al., 2014) and the simple LSTM, the dependency Tree-LSTM and neural network implementations by Tai et al. (2015). [2] The system uses the scientific computing framework Torch[3]. Training is performed on the data computed in Section 5. The system uses a mini batch size of 25 with learning rate 0.05 and regularization strength 0.0001. The compositional parameters for our Tree-LSTM systems with memory dimensions 150 and 300 are 203,400 and 541,800, respectively. The training is performed for 10 epochs. System-level scores are computed by aggregating and normalising segment-level scores.

5 Computing Similarity Scores from WMT Rankings

As we do not have access to any dataset which provides scores to segments on the basis of translation quality, we used the WMT-13 ranks corpus to automatically derive training data. This corpus is a by-product of the manual systems evaluation carried out in the WMT-13 evaluation. In the evaluation, the annotators are presented with a source segment, the output of five systems and a reference translation. The annotators are given the following instructions: "*You are shown a source sentence followed by several candidate translations. Your task is to rank the translations from best to worst (ties are allowed)*". Using the WMT-13 ranked corpus, we derived a corpus where the reference and corresponding translations are assigned similarity scores. The fact that *ties are allowed* makes it more suitable to generate similarity scores. If all translations are bad, annotators can mark all as rank 5 and if all translations are accurate, annotators can mark all as rank 1. The selection of the WMT-13 corpus over other WMT workshops is motivated by the fact that it is the largest among them. It contains ten times more ranks than WMT-12 and three to four times more than WMT-14. This also makes it possible to obtain enough reference translation pairs which are evaluated several times.

Our hypothesis is that if a translation is given a certain rank many times, this reflects its similarity score with the reference. A better ranked translation among many systems will be close to the reference whereas a worse ranked translation among many systems will be dissimilar from the reference. To remove noisy pairs, we collect reference translation pairs below a certain variance only. We determined appropriate variance values using Algorithm 1 below for $n = 3, 4, 5, 6, 7$ and ≥ 8, separately. The computed variance values are given in Table 1.

n	3	4	5	6	7	≥ 8
Var	0.65	1.0	1.2	1.2	1.3	0.85

Table 1: Variances computed using Algorithm 1

Algorithm 1 Variance Computation

```
procedure GETVARIANCE($judgements$)
    $V, v \leftarrow -1, 0.25$                          ▷ Initialise $N$
    for $v \leq max$ do
        $prs \leftarrow$ pairs with variance below $v$
        $score \leftarrow kendall(prs, judgements)$
        if $score \geq 0.78$ then
            $V \leftarrow v$
            $v \leftarrow v + 0.05$
        else
            $break$
    Return $V$                                      ▷ Return variance
```

In Algorithm 1, the *kendall* function calculates Kendall tau correlation using the WMT-13 human judgements. We select a set for which the correlation coefficient is greater than 0.78.[4] The correlation is computed using the annotations for which scores are available in the corpus (prs). In other words, the corpus acts as a scoring function for the available reference translation pairs, which gives a similarity score between a reference and a translation. We selected pairs below the variance values obtained for $n = 4, 5, 6, 7$ and ≥ 8. Finally, all the pairs are merged to obtain a set (L). Apart from this set, we created three other sets for our experiments. The last two also use the SICK data (Marelli et al., 2014) which was developed for evaluating semantic similarity. All four sets are described below:

L: contains the set generated by selecting the pairs ranked *four or more times* and filtering the segments based on the variance

LNF: contains the set generated by selecting the pairs ranked *four or more times* without any filtering depending on the variance

[2] The adapted code for MT evaluation scenarios is available at https://github.com/rohitguptacs/ReVal.

[3] http://torch.ch

[4] The score was decided so that we obtain around 10K pairs which are annotated at least four times.

L+Sick: Added 4500 sentence pairs from the SICK training set to Set L in the training set and 500 pairs in the development set.

XL+Sick: Added also the pairs ranked *three times* to Set L+Sick.

	Train	Dev	Test
L	9559	1000	1000
LNF	17855	1000	1000
L+Sick	14059	1500	1000
XL+Sick	21356	1500	1000

Table 2: Derived Corpus statistics

Table 2 shows the number of pairs extracted for each set to train our LSTM based models.[5]

6 Results

We evaluate our approach trained on the four different datasets obtained from WMT-13 (as given in Table 2) on WMT-14. Table 3 shows system-level Pearson correlation obtained on different language pairs as well as average Pearson correlation (PAvg) over all language pairs. The last column of the table also shows average Spearman correlation (SAvg). The 95% confidence level scores are obtained using bootstrap resampling as used in the WMT-2014 metric task evaluation. The scores in bold show best scores overall and the scores in bold italic show best scores in our variants.

In Table 3 and Table 4, the first section (L+Sick(lstm)) shows the results obtained using simple LSTM (layer 1, hidden dimension 50, memory dimension 150, compositional parameters 203400). The second section shows the scores of our Tree-LSTM metric trained on different training sets and dimensions. Dimensions are shown in brackets, e.g L(50,150) shows the results on set 'L' with the hidden dimension 50 and the memory dimension 150. L+Sick(mix) shows results of combining the two systems: L+Sick(50,150) and L+Sick(100,150). For the sentences longer than 20 words, the system uses scores of L+Sick(100,150) and scores of L+Sick(50,150) for the rest. The third section shows the best three overall systems from the WMT-14 metric task. The fourth section in Table 3 shows the systems from the WMT-14 task which obtained best results for certain languages but do not preform well overall. The last section in Tables 3 and 4 shows systems implementing BLEU (or variants for the segment level) and METEOR in the WMT-14 metric task.

Tables 3 and 4 contain a deluge of evaluation data, mainly to explore the effect of different training data and model parameter settings for our models. The main messages can be summarised as follows: 1. Tree LSTM models significantly outperform the LSTM model (L+Sick(lstm) and L+Sick(50,150) have the same data and parameter settings). 2. For Tree-LSTM models different parameter settings have only a minor impact on performance (in fact only for a few language pairs (e.g. hi-en at system-level, L+Sick(100, 300) and L+Sick(100,150)) results are statistically significantly different). This is reassuring as it indicates that the metric is not overly sensitive to extensive and delicate parameter tuning. 3. For the system level evaluation Tree-LSTM models are fully competitive with the best of the current complex models that combine many different metrics, substantial external resources and may require a significant amount of feature engineering and tuning. 4. For the segment level evaluation our metric outperforms BLEU based approaches and the other three systems[6] but lags behind some other approaches. We investigate this further below.

Tables 3 and 4 show that set L is able to obtain similar results compared to set LNF even though we filter out almost half of the pairs. Table 3 shows that for L+Sick(50, 150) and L+Sick(mix), we obtained an average second best Pearson correlation and best Spearman correlation coefficient. We also obtained better results for the Russian-English and Czech-English language pairs compared to any other systems in the WMT-14 task.

We also evaluate our setting L-Sick(50,150) on the WMT-12 task dataset. Our metric performs best for two out of four language pairs and best overall at the system level with 0.950 and 0.926 Pearson and Spearman correlation coefficient, respectively. At the segment level, we obtained 0.222 Kendall tau correlation which was better than seven out of the total ten metrics in the WMT-12 task.

One of the reasons for the difference in segment-level and system-level correlations is that Kendall Tau segment-level correlation is calcu-

[5] For testing our approach we use WMT-12 and WMT-14 rankings instead of the test sets in this table.

[6] These three systems are not given in this paper. Please refer (Machácek and Bojar, 2014) for results of these systems.

Test	**cs-en**	**de-en**	**fr-en**	**hi-en**	**ru-en**	**PAvg**	**SAvg**
L+Sick(lstm)	.922 ± .051	.882 ± .028	.974 ± .009	.898 ± .011	.863 ± .023	.908 ± .024	.872 ± .060
LNF(50,150)	.972 ± .032	.900 ± .026	.974 ± .009	.900 ± .011	.882 ± .021	.925 ± .020	.913 ± .045
L(50,150)	.988 ± .022	.897 ± .027	.978 ± .008	.905 ± .010	.875 ± .022	.929 ± .018	.904 ± .042
L+Sick(50,150)	.993 ± .017	.904 ± .025	.978 ± .008	.908 ± .010	.881 ± .022	.933 ± .016	.915 ± .042
L+Sick(100,300)	.993 ± .018	.907 ± .025	.973 ± .009	.866 ± .012	**.890** ± .020	.926 ± .017	.902 ± .050
XL+Sick(100,300)	.913 ± .054	***.917*** ± .024	.978 ± .008	.904 ± .010	.884 ± .022	.919 ± .024	.889 ± .055
L+Sick(100,150)	***.994*** ± .016	.911 ± .025	.975 ± .009	***.923*** ± .010	.870 ± .022	***.935*** ± .016	.904 ± .049
L+Sick(mix)	***.994*** ± .017	.906 ± .025	***.979*** ± .008	.918 ± .010	.881 ± .022	***.935*** ± .016	***.919*** ± .045
DISCOTK-PARTY-TUNED	.975 ± .031	**.943** ± .020	.977 ± .009	.956 ± .007	.870 ± .022	**.944** ± .018	.912 ± .043
LAYERED	.941 ± .045	.893 ± .026	.973 ± .009	**.976** ± .006	.854 ± .023	.927 ± .022	.894 ± .047
DISCOTK-PARTY	.983 ± .025	.921 ± .024	.970 ± .010	.862 ± .015	.856 ± .023	.918 ± .019	.856 ± .046
REDSYS	.989 ± .021	.898 ± .026	**.981** ± .008	.676 ± .022	.814 ± .026	.872 ± .021	.786 ± .047
REDSYSSENT	.993 ± .018	.910 ± .024	.980 ± .008	.644 ± .023	.807 ± .027	.867 ± .020	.771 ± .043
BLEU	.909 ± 0.54	.832 ± .034	.952 ± .012	.956 ± .007	.789 ± .027	.888 ± .027	.833 ± .058
METEOR	.980 ± .029	.927 ± .022	.975 ± .009	.457 ± .027	.805 ± .026	.829 ± .023	.788 ± .046

Table 3: Results: System-Level Correlations on WMT-14

Test	**cs-en**	**de-en**	**fr-en**	**hi-en**	**ru-en**	**Average**	**Avg wmt12**
L+Sick(lstm)	.204 ± .015	.232 ± .014	.289 ± .013	.319 ± .013	.236 ± .012	.256 ± .013	.254 ± .013
NFL(50,150)	.228 ± .015	***.288*** ± .014	.318 ± .014	.341 ± .014	.271 ± .012	.289 ± .014	.287 ± .014
L(50,150)	.225 ± .015	.272 ± .014	.328 ± .013	.346 ± .013	.280 ± .011	.290 ± .013	.287 ± .013
L+Sick(50,150)	.243 ± .016	.274 ± .013	.333 ± .013	.360 ± .014	.278 ± .011	.298 ± .013	.295 ± .014
L+Sick(100,300)	.233 ± .014	.286 ± .014	.343 ± .014	.358 ± .013	***.281*** ± .011	.300 ± .013	.297 ± .013
XL+Sick(100,300)	***.252*** ± .014	.279 ± .014	***.347*** ± .013	.367 ± .013	.274 ± .011	***.304*** ± .013	***.301*** ± .013
L+Sick(100,150)	.243 ± .016	.274 ± .014	.329 ± .013	***.368*** ± .012	.276 ± .011	.298 ± .013	.295 ± .013
L+Sick(mix)	.243 ± .016	.276 ± .013	.338 ± .013	.358 ± .013	.273 ± .011	.298 ± .013	.295 ± .013
DISCOTK-PARTY-TUNED	**.328** ± .014	**.380** ± .014	**.433** ± .013	.434 ± .013	**.355** ± .010	**.386** ± .013	**.386** ± .013
BEER	.284 ± .015	.337 ± .014	.417 ± .013	**.438** ± .014	.333 ± .011	.362 ± .013	.358 ± .013
REDCOMBSENT	.284 ± .015	.338 ± .013	.406 ± .012	.417 ± .014	.336 ± .011	.356 ± .013	.346 ± .013
METEOR	.282 ± .015	.334 ± .014	.406 ± .012	..420 ± .013	.329 ± .010	.354 ± .013	.341 ± .013
BLEU_NRC	.226 ± .014	.272 ± .014	.382 ± .013	.322 ± .013	.269 ± .011	.294 ± .013	.267 ± .013
SENTBLEU	.213 ± .016	.271 ± .014	.378 ± .013	.300 ± .013	.263 ± .011	.285 ± .013	.258 ± .014

Table 4: Results: Segment-Level Correlations on WMT-14

lated based on rankings and does not consider the amount of difference between scores. Here is an example similar to that given in (Hopkins and May, 2013). Suppose four systems produce the translations T0, T1, T2 and T3. Suppose we have two metrics M1 and M2 and they produce scores and rankings as follows. GS represents the correct ranking and scores; Scores are in a scale [0, 1] with a higher score indicating a better translation:

M1: T0 (0.10), T3 (0.71), T1 (0.72), T2 (0.73)

M2: T1 (0.71), T0 (0.72), T2 (0.73), T3 (0.74)

GS: T0 (0.10), T1 (0.71), T2 (0.72), T3 (0.73)

Certainly, M1 produces better scores and ranking than M2. But, Kendall Tau segment-level correlation is higher for M2. (There are four concordant pairs in the M1 rank and five in the M2 rank.) Therefore, if a metric does not scale well as per the quality of translations, it may still obtain a good Kendall Tau segment-level correlation and a better metric may end up getting a low correlation. Another reason for the discrepancy between segment and system-level scores may be a low agreement on annotations. For the WMT-14 dataset, inter-annotator and intra-annotator agreement were 0.367 and 0.522. These problems should not occur with Pearson correlation at the system level because system-level scores are calculated using more sophisticated approaches (Koehn, 2012; Hopkins and May, 2013; Sakaguchi et al., 2014). For example, Hopkins and May (2013) model the differences among annotators by adding random Gaussian noise.

7 Conclusion

We conclude that our dense-vector-space-based ReVal metric is simple, elegant and effective with state-of-the-art results. ReVal is fully competitive with the best of the current complex alternative approaches that involve system combination, extensive external resources, feature engineering and tuning.

Acknowledgement

The research leading to these results has received funding from the People Programme (Marie Curie Actions) of the European Unions Seventh Framework Programme FP7/2007-2013/ under REA grant agreement no. 317471 and the EC- funded project QT21 under Horizon 2020, ICT 17, grant agreement no. 645452.

References

Yoshua Bengio, Patrice Simard, and Paolo Frasconi. 1994. Learning long-term dependencies with gradient descent is difficult. *IEEE Transactions on Neural Networks*, 5(2):157–166.

Alexandra Birch and Miles Osborne. 2011. Reordering metrics for MT. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies-Volume 1*, pages 1027–1035. Association for Computational Linguistics.

Ondřej Bojar, Christian Buck, Chris Callison-Burch, Christian Federmann, Barry Haddow, Philipp Koehn, Christof Monz, Matt Post, Radu Soricut, and Lucia Specia. 2013. Findings of the 2013 Workshop on Statistical Machine Translation. In *Proceedings of the Eighth Workshop on Statistical Machine Translation*, pages 1–44, Sofia, Bulgaria, August. Association for Computational Linguistics.

Michael Collins and Nigel Duffy. 2001. Convolution kernels for natural language. In *Advances in Neural Information Processing Systems*, pages 625–632.

Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of LREC*, volume 6, pages 449–454.

Michael Denkowski and Alon Lavie. 2014. Meteor universal: Language specific translation evaluation for any target language. In *Proceedings of the EACL 2014 Workshop on Statistical Machine Translation*.

George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the second international conference on Human Language Technology Research*, pages 138–145. Morgan Kaufmann Publishers Inc.

Shubham Gautam and Pushpak Bhattacharyya. 2014. Layered: Metric for machine translation evaluation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*.

Jesús Giménez and Lluís Màrquez. 2010. Linguistic measures for automatic machine translation evaluation. *Machine Translation*, 24(3-4):209–240.

Francisco Guzmán, Shafiq Joty, Lluís Màrquez, and Preslav Nakov. 2015. Pairwise neural machine translation evaluation. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 805–814, Beijing, China. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Mark Hopkins and Jonathan May. 2013. Models of translation competitions. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1416–1424, Sofia, Bulgaria.

Shafiq Joty, Francisco Guzmán, Lluís Màrquez, and Preslav Nakov. 2014. DiscoTK: Using Discourse Structure for Machine Translation Evaluation. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*.

Philipp Koehn. 2012. Simulating human judgment in machine translation evaluation campaigns. In *Proceedings of the Ninth International Workshop on Spoken Language Translation*, pages 179–184, Hong Kong.

Quoc Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *Proceedings of The 31st International Conference on Machine Learning*, pages 1188–1196.

Matouš Machácek and Ondrej Bojar. 2014. Results of the WMT-14 metrics shared task. In *Proceedings of the Ninth Workshop on Statistical Machine Translation*.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 2014. A sick cure for the evaluation of compositional distributional semantic models. In *Proceedings of LREC 2014*.

Tomas Mikolov, Quoc V Le, and Ilya Sutskever. 2013a. Exploiting Similarities among Languages for Machine Translation. *CoRR*, pages 1–10.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013b. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems*, pages 3111–3119.

Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the ACL*, pages 311–318.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the Empiricial Methods in Natural Language Processing (EMNLP 2014)*, pages 1532–1543.

Keisuke Sakaguchi, Matt Post, and Benjamin Van Durme. 2014. Efficient elicitation of annotations for human evaluation of machine translation. In *In Proceedings of the Ninth Workshop on Statistical Machine Translation*, Baltimore, Maryland.

Matthew Snover, Nitin Madnani, Bonnie Dorr, and Richard Schwartz. 2008. TERp system description. In *MetricsMATR workshop at AMTA*. Citeseer.

Richard Socher, Eh Huang, and Jeffrey Pennington. 2011. Dynamic Pooling and Unfolding Recursive Autoencoders for Paraphrase Detection. In *Advances in Neural Information Processing Systems*, pages 801–809.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. 2013a. Parsing With Compositional Vector Grammars. *In Proceedings of the ACL*, pages 455–465.

Richard Socher, Alex Perelygin, and Jy Wu. 2013b. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of EMNLP*, pages 1631–1642.

Kai Sheng Tai, Richard Socher, and Christopher D. Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 1556–1566, Beijing, China. Association for Computational Linguistics.

Wojciech Zaremba and Ilya Sutskever. 2014. Learning to execute. *arXiv preprint arXiv:1410.4615*.