UMDeep at SemEval-2017 Task 1: End-to-End Shared Weight LSTM Model for Semantic Textual Similarity

Joe Barrow
University of Maryland
Computer Science
jdbarrow@cs.umd.edu

Denis Peskov
University of Maryland
Computer Science
dpeskov@cs.umd.edu

Abstract

We describe a modified shared-LSTM network for the Semantic Textual Similarity (STS) task at SemEval-2017. The network builds on previously explored Siamese network architectures. We treat max sentence length as an additional hyperparameter to be tuned (beyond learning rate, regularization, and dropout). Our results demonstrate that hand-tuning max sentence training length significantly improves final accuracy. After optimizing hyperparameters, we train the network on the multilingual semantic similarity task using pre-translated sentences. We achieved a correlation of 0.4792 for all the subtasks. We achieved the fourth highest team correlation for Task 4b, which was our best relative placement.

1 Introduction

Semantic Textual Similarity (STS) has been a staple of the SemEval competition and requires systems that automatically identify the semantic relatedness of two sentences. The resulting system could be used down-stream in many important NLP tasks, such as scoring the output of a machine translation system or finding related document/query pairs in web search.

The data available for this competition has been updated annually and contains gold-label, human-evaluated scores based on sentence pairs across multiple languages ((Agirre et al., 2012), (Agirre et al., 2013), (Agirre et al., 2014), (Agirre et al., 2015)). The gold label for each sentence pair is in the range $[0, 5]$, with 0 being *the sentences are completely dissimilar* to 5 being *the sentences are completely equivalent*. (Agirre et al., 2016)

The task is not restricted to English or monolingual similarity scoring. The 2017 SemEval task consists of seven different tracks, each with a different language pair: Arabic-Arabic, Arabic-English, Spanish-Spanish, Spanish-English, an additional Spanish-English track, English-English, and English-Turkish. We avoid language-specific feature engineering and take a representation learning approach to STS. This requires constructing directly-comprable sentence representations that can be induced from the limited amounts of annotated STS training data.

We present a modified version of the Siamese Long Short-Term Memory (LSTM) network to solve this problem. (Mueller and Thyagarajan, 2016) A Siamese network is one in which parameters between layers are shared, and are updated in parallel during the learning phase. For the semantic relatedness task, this allows two sentences to be encoded into the same space using a single shared recurrent neural network. The dual-encoding enables the use of end-to-end supervised deep learning, using only the surface forms of the sentences and the gold labels.

We extend the Siamese LSTM in two ways. First, we consider the semantic relatedness as a classification, rather than a regression problem. Initially, semantic relatedness appears to be a continuous one-dimensional measure suitable for regression. However, there are many subtleties within the bands of scores, as sentences can differ along more than a single dimension. Thus, rather than regressing over the label, our model generates a distribution over possible labels. Second, we use a different concatenative dense layer on top of the dual LSTMs to better model the classification problem (Tai et al., 2015), and train using KL-Divergence as the loss function for training.

Our results did not achieve the state-of-the-art performance possible with a Siamese LSTM architecture. Despite this set-back, we are able to demonstrate the effect of sentence training length on a LSTM. Additionally, all foreign languages were translated through Google Translate and the same model was used for the seven tracks. This standardization provides insight into the quality of Google Translate and the negative effect of machine translation on correlation.

The following section provides detail on our system and the training process. As our submission is focused on the use of end-to-end deep learning in semantic relatedness, we do not use hand-crafted features from external data, except for pre-trained word embeddings to speed up training. A visual overview of the shared LSTM model can be seen in Figure 1.

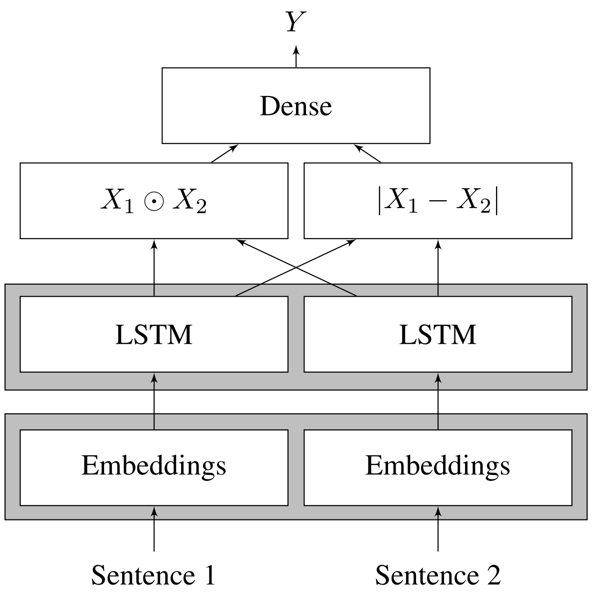

Figure 1: The end-to-end shared-LSTM model. Note that the shaded boxes represent shared parameters that are updated in parallel when the error is backpropagated. In this model, both the embeddings and the LSTM weights are shared, meaning the sentences are encoded into the same space. The model was implemented in Lasagne. (Dieleman et al., 2015)

2 End-to-End Shared LSTM

We use a shared-parameter LSTM model, also known as a Siamese LSTM model, as a completely end-to-end deep learning model. (Mueller and Thyagarajan, 2016; Tai et al., 2015)

Shared Parameters. In the siamese LSTM, the embedding layers share weights with each other, as do the LSTM layers. These weights are shared throughout the entire training process, so updates applied to one are applied to both. Each sentence was transformed into a sequence of embeddings and then encoded into a sentence vector by the LSTM, and since the embedding and LSTM layers were the same for both sentences, both sentences were encoded into the same space. The sentence embeddings were the final vector in the LSTM.

Word Embeddings. The model was initialized with GloVe word embeddings. (Pennington et al., 2014) Our experiments with both GloVe and the Paragram (Wieting et al., 2015) embeddings showed only a negligible difference in the final performance of the model. This difference disappeared when embeddings were made trainable.

Should one include all the embeddings or simply the subset seen in the training data? If all the embeddings are included, then the model should theoretically generalize better, as there are fewer UNKNOWN's in the validation and testing data. However, if all the embeddings are included and the embeddings are trainable, then only the seen portion of the embedding space is updated, which could hurt model generalization.

Our model uses the whole embedding space, but also updates the embeddings after each batch. Although updating only part of the space could risk damaging model generalizability, our experiments found that we actually saw an improvement in generalizability with both the whole embedding space and trainable embeddings.

Dense Concatenative Layer. The original equation for the dense layer is: $exp(-||X_1 - X_2||_1)$. (Mueller and Thyagarajan, 2016) However, as shown in Figure 1, our dense layer takes the concatenation of two different transformations: $|X_1 - X_2|$ and $X_1 \odot X_2$. This is used to capture both the difference in the angle and the absolute difference of the two sentences. (Tai et al., 2015)

Training Objective. In order to use Kullback-Leiber divergence (KL divergence) as the objective function, we had to convert the gold labels into probability distributions: (Tai et al., 2015)

$$p_i = \begin{cases} y - \lfloor y \rfloor, & i = \lfloor y \rfloor + 1 \\ \lfloor y \rfloor - y + 1, & i = \lfloor y \rfloor \\ 0 & otherwise \end{cases}$$

Thus, a label of 4.7 would distribute 70% of its probability mass to the category 5, and 30% of its probability mass to the category 4. To convert from a probability distribution to a prediction, simply take the dot product of the ordered vector $< 0, 1, 2, 3, 4, 5 >$ and the distribution.

Then, the loss for each example was computed using the standard KL divergence formula, with some minor smoothing to disallow zeros: $D_{KL}(P||Q) = \sum_i P(i) log \frac{P(i)}{Q(i)}$.

3 Experiments and Results

We opted for minimal preprocessing in our final model: merely tokenizing and lower-casing the input. Lemmatizing the words did not lead to a notable improvement, and hence was omitted. Additionally, experimenting with targeted Part of Speech exclusion (removing all articles, increasing weight of proper nouns, etc.) did not produce dramatically higher results. Therefore, we decided to let the LSTM learn for itself.

Our final results on the 2017 data are shown in Table 1. Retrospectively, we saw that the 2016 postediting data (65.13% accuracy when 2016 data was held out from training) would have served as a close proxy for 2017 En-En performance. Our three submissions to the 2017 Semeval task were trained treating maximum training sentence length as a hyperparameter. Our results show that this parameter can have a large impact on the final outcome of the model.

The cosine baseline provided by SemEval organizers achieved a 0.72 correlation for the English-English sentences, which was roughly 0.07 higher than our best performance on the same dataset. Although disheartening, a Siamese LSTM model is capable of performing dramatically better with curated training data, whereas the baseline approach cannot be significantly modified.

Network Architecture and Parameters. Our final model used length 300 GloVe embeddings, 100 LSTM cells, 50 neurons in the final dense layer, and 6 output neurons, one for each class. We used the Kullback-Leibler Divergence of the output distribution and the gold label distribution as the objective function.

Data. We used all available past STS Task 1 datasets and no external data. In order to participate in the non-English tracks, we used Google Translate to translate all the sentence pairs into English. We then used the model trained on the English-English pairs on the translated-English data.

4 Discussion

Length. As shown in Table 1, the best identified length was 20. Meanwhile, the median length for the labeled sentences was below 11. Training on the max length saw an improvement on the English-English dataset, but an overall decrease in performance on the other datasets, in particular the SP-EN-WMT dataset, which contained very long English-Spanish sentence pairs. This is likely due to the network capturing long-term dependencies present in the native English sentence pairs that weren't present in the translated sentence pairs.

Translations. Our results demonstrate that the introduction of machine translation into the pipeline damages performance. The drop for non-English monolingual tasks exceeds that for English cross-lingual tasks, as translation is only applied to one side in the latter.

- **Spanish** - On the translated Spanish-Spanish sentence pairs, our correlation went down from 0.62 to 0.52. However, the drop was only to 0.56 on the English-Spanish sentence pairs, likely because half of the data was the native English used in training.

- **Arabic** - We saw a larger drop in accuracy on the Arabic-Arabic sentence pairs, from 0.62 to 0.48. This likely demonstrates that the translation quality of Google Translate is higher for Spanish than for Arabic. As was the case with Spanish, the Arabic-English pairs did better than the Arabic-Arabic pairs, achieving a correlation of 0.49 with the length 20 model, and 0.52 with the max length model.

- **Turkish** - Although there was no Turkish-Turkish track this year, our system performed roughly as expected on the Turkish-English track, given its performance on the Spanish-English and Arabic-English tracks. Uniquely in Turkish, accuracy spikes between the length 20 and max length models: from .53 to over .57 respectively.

Overall, we found the superior translation of Spanish unsurprising given the similarity of the languages and the large corpora available for Spanish-English translations.

Investigating the Results. Table 2 shows a selection of sentence pairs, their gold labels, and our

2017 Language Pairs	**Number of Pairs**	**Length 11 (ρ)**	**Length 20 (ρ)**	**Max Length (ρ)**
AR-AR	250	0.3905	0.4753	0.4587
AR-EN	250	0.3713	0.4939	0.5199
SP-SP	250	0.4588	0.5165	0.5148
SP-EN	250	0.3482	0.5615	0.5232
SP-EN-WMT	250	0.0586	0.1609	0.1300
EN-EN	250	0.4727	0.6174	0.6222
EN-TR	250	0.3644	0.5293	0.5725
Weighted Mean	-	**0.3521**	**0.4792**	**0.4773**

Table 1: Results in the different tracks of SemEval-2017. The lengths refer to the maximum lengths of the sentences used for training the model.

Sentence 1	**Sentence 2**	**Gold**	**Pred.**
A man is performing labor.	A man is performing today.	2.8	1.5
A kid sits on a soccer ball outside.	A kid sitting on a soccer ball at the park.	4.2	4.2
The player shoots the winning points.	The basketball player is about to score points for his team.	2.8	2.7
The yard has a dog.	The dog is running after another dog.	1.6	4.1
the people are running a marathon	People are running a marathon	5.0	0.9

Table 2: A selection of results showing the successes and failures of our shared-LSTM architecture. These sentences were selected to show areas in which our system excels or under-performs.

system's predicted score. There were many cases in which our system achieved very precise scoring, as included in the table. The examples on which the end-to-end model failed prove more interesting.

There were many simple examples that fooled our system. The most notable one is the pair (”the people are running a marathon”, ”People are running a marathon”). In this case, the only difference is the inclusion of the determiner ”the” at the start of the sentence, as the capitalization of people would have been removed during preprocessing. Yet our system predicts the relatedness to be 0.9, rather than 5.0. This example shows that, although the sentences are theoretically encoded into the same space, the series of transformations that the sentence undergoes is complex and imperfect. Another such example is the pair (”The yard has a dog.”, ”The dog is running after another dog.”) The fact that a dog exists in both sentences should not merit such a high score alone. This trend of attributing similarity to sentences with similar subjects percolates throughout our results.

The sentence pair (”A man is performing labor.”, ”A man is performing today.) demonstrates the learning potential of our model's predictions. These sentences are identical in length and the surface forms are 80% similar as only the final word differs. However, the different sense of perform make these sentences mostly unrelated. The difference is subtle, and unlikely to be picked up by a more naive system. Some basic ability to disambiguate different word senses is suggested by the shared weight LSTM's 1.5 assignment.

5 Conclusion and Future Work

A Siamese LSTM architecture has the potential for generating sophisticated predictions, but relies heavily on selecting appropriate training data. Our results show that hand-tweaking the maximum length of training sentences can significantly affect model output. Additionally, we show that the LSTM model performs worse on machine-translated data than on native English sentences.

There are several possible extensions of the proposed shared LSTM framework, such as a tree-structured, rather than linear LSTM. This uses the sentence parse as a ”feature” for structuring the model, and can provide significant improvements over a purely linear LSTM for semantic relatedness. (Tai et al., 2015)

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2014. Semeval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th international workshop on semantic evaluation (SemEval 2014)*. Association for Computational Linguistics Dublin, Ireland, pages 81–91.

Eneko Agirre, Carmen Baneab, Claire Cardiec, Daniel Cerd, Mona Diabe, Aitor Gonzalez-Agirrea, Weiwei Guof, Inigo Lopez-Gazpioa, Montse Maritxalara, Rada Mihalceab, et al. 2015. Semeval-2015 task 2: Semantic textual similarity, english, spanish and pilot on interpretability. In *Proceedings of the 9th international workshop on semantic evaluation (SemEval 2015)*. pages 252–263.

Eneko Agirre, Carmen Baneab, Daniel Cerd, Mona Diabe, Aitor Gonzalez-Agirrea, Rada Mihalceab, German Rigaua, Janyce Wiebef, and Basque Country Donostia. 2016. Semeval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. *Proceedings of SemEval* pages 497–511.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. sem 2013 shared task: Semantic textual similarity, including a pilot on typed-similarity. In *In* SEM 2013: The Second Joint Conference on Lexical and Computational Semantics. Association for Computational Linguistics*. Citeseer.

Eneko Agirre, Mona Diab, Daniel Cer, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of 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*. Association for Computational Linguistics, pages 385–393.

Sander Dieleman, Jan Schlter, Colin Raffel, Eben Olson, Sren Kaae Snderby, Daniel Nouri, et al. 2015. Lasagne: First release. https://doi.org/10.5281/zenodo.27878.

Jonas Mueller and Aditya Thyagarajan. 2016. Siamese recurrent architectures for learning sentence similarity. In *AAAI*. pages 2786–2792.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*. volume 14, pages 1532–1543.

Kai Sheng Tai, Richard Socher, and Christopher D Manning. 2015. Improved semantic representations from tree-structured long short-term memory networks. *arXiv preprint arXiv:1503.00075* .

John Wieting, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. Towards universal paraphrastic sentence embeddings. *arXiv preprint arXiv:1511.08198* .