Lasts4Ever: Language-Agnostic Semantic Text Similarity for "Every" Language

Stanford CS224N Custom Project

Miles McCain
Department of Computer Science
Stanford University
mccain@stanford.edu

Rhythm Garg
Department of Computer Science
Stanford University
rhythmg@stanford.edu

Igor Barakaiev
Department of Computer Science
Stanford University
igorb@stanford.edu

Abstract

In Semantic Textual Similarity (STS), a model must score how similar the meanings of two sentences are. The model's performance is measured against the STSb Multi MT benchmark dataset, which contains Amazon Mechanical Turk human annotations. There has been significant interest in this task recently due to its broad applicability across various use-cases, from machine translation to question answering. We propose a method for multilingual Semantic Text Similarity (STS) that leverages both parallel corpora and available human STS annotations, but does not require this data for every language pair. Our goal is to transfer knowledge learned from multilingual STS training data on some language pairs to other language pairs, thereby unlocking new value for low-resource languages that do not have human STS annotations. Our model achieves this goal through a two-stage training pipeline. Our Stage 1 model, which trains on unannotated parallel corpora, is a complete reimplementation of Tiyajamorn et al. (2021) that extracts aligned language-agnostic "meaning embeddings" from large pre-trained multilingual sentence encoding models. Our Stage 2 model, which is trained on human-annotated STS data for a subset of language pairs, augments these meaning embeddings to achieve even higher multilingual STS performance. Our best models improve the average Pearson STS score by an average of 0.06 atop LaBSE and 0.013 atop XLM-R. We find that these models also improve STS performance for language pairs on which neither the Stage 1 or Stage 2 models were trained.

1 Key Information to Include

- TA mentor: Elaine Sui
- External collaborators (if you have any): No
- Sharing project: No

2 Introduction

Semantic Textual Similarity (STS) is a task in which models must score how similar the meanings of two sentences are. There has been significant interest in this task recently due to its broad applicability across various use-cases, from machine translation to question answering. In SemEval 2017 Task 1[1], Cer et al. provide a formal outline of STS. The performance of a model on the STS task is defined straightforwardly to be the Pearson correlation of the model's similarity scores and human annotators' similarity scores across pairs of sentences.

The models typically used for these tasks are sentence encoders. These models represent sentences as high-dimensional vectors. Reimers et al. introduce Siamese BERT networks, in which the

semantic similarity between two sentences is defined as the cosine similarity between those sentences' embeddings [2]. Multilingual sentence encoders can represent sentences from many different languages. In a perfectly aligned multilingual sentence encoder, the same sentence in two different languages would produce the same sentence embedding.

Alignment is difficult. Pre-trained multilingual sentence encoders require parallel corpora to achieve alignment, and often even require human-annotated training data to achieve high STS performance. Tiyajamorn et al. identify this data requirement as an important bottleneck to high multilingual STS performance, especially for lower-resourced languages.

3 Related Work

In the current literature, many multilingual sentence encoders, including mBERT[3], are single self-attention networks. They are pre-trained on monolingual corpora in over 100 languages and optimize for masked language modelling, in which the model has to predict a random sample of tokens in the original text that have been replaced by a placeholder. In slight contrast, LaBSE[4] has been trained in both masked language modelling and translation language modelling, in which the model has to predict the translation for a token.

However, these pre-trained multilingual sentence encoders are often not sufficiently sensitive to cross-language semantic similarity. To address this insensitivity, Reimers and Gurevych [5] used human STS annotations to improve a pre-trained multilingual sentence encoder (BERT). Their approach only supports sentence pairs corresponding to language pairs that have human-annotated data (e.g. the model would only be able to score the similarity of an English sentence and a French sentence if their dataset included English-French human STS annotations).

Tiyajamorn et al. propose a method for extracting aligned language-agnostic *meaning embeddings* from sentence embeddings, using contrastive learning to train only on parallel corpora. Their novel approach forms the basis of our own models, which we describe in more detail below. However, the approach presented by Tiyajamorn et al. is not the only method for improving STS performance using parallel corpora. For example, Libovický et al. [6] take embeddings from pre-trained multilingual sentence encoders and, for each embedding $\mathbf{e}$ of a sentence that is of a language L, they subtract the mean embedding of L from $\mathbf{e}$ — this is known as the centered method. Libovický et al. also employ another approach called the projection method, where they use a parallel corpus and map embeddings in other languages into corresponding embeddings in English. Both the centered method and the projection method attempt to remove the parts of an embedding vector that are characteristic to the language corresponding to the embedding, and Tiyajamorn et al. use both methods as baselines.

However, Tiyajamorn et al. do not rely on any human-annotated data; using only parallel corpora, they build sentence encoders that perform well on multilingual STS. Their approach therefore does *not* take advantage of human-annotated sentence pairs, even though they are available for many higher-resourced languages. Therefore, there exists additional information that could be used to improve their sentence encoder, which could transfer to improve STS performance on lower-resourced languages without human-annotated sentence pairs. This insight forms the basis of our two-stage training pipeline.

4 Approach

Main Approach. We began by fully reimplementing Tiyajamorn et al.'s [7] architecture. We called this "Stage 1" of our training pipeline, and we do not rely on any human STS annotations. Tiyajamorn et al.'s approach works by passing sentence embeddings from an arbitrary multilingual sentence encoder (e.g., LaBSE [8] or XLM-R [9]) through two multi-layer perceptrons (MLPs), one of which extracts "meaning" embeddings (MLP_M) while the other extracts "language" embeddings (MLP_L). The sum of the language and meaning embedding should reconstruct the original embedding. The meaning embedding is intended to capture the semantics of the sentence, independent of language-specific details.

Figure 4 illustrates our Stage 1 architecture, which is a reimplementation of Tiyajamorn et al.'s architecture. It starts with a sentence being fed into a sentence encoding transformer, which is then transformed to a sentence embedding of size `embed_size` (for both LaBSE and XLM-R,`embed_size` is 768). Then, there are two separate fully-connected layers to a language embedding and a meaning embedding, and both embeddings are of size `embed_size`. The language embedding is then processed further to detect a specific language. A key part of this architecture are the loss functions, which

help connect the meaning and language embeddings, as can be seen in Figure 2. In particular, the loss functions ensure that the meaning embedding is capturing some notion of similarity between sentences agnostic of language, that the language embedding is sufficient for classifying the language of a sentence, and that the meaning and language embeddings sum to the original sentence embedding. In other words, the output of LaBSE is "disentangled" through the architecture above to produce a meaning embedding which can then be used for multilingual STS.

After reimplementing Tiyajamorn et al.'s architecture on top of a pretrained transformer, we apply our "Stage 2" model, as shown in Figure 4, which augments meaning embeddings from Tiyajamorn et al.'s architecture for high STS performance. We train our Stage 2 model using human STS annotations on a subset of language pairs. We used a *subset* to preserve language pairs with unseen human STS annotations for testing. Stage 2 starts with a meaning embedding produced by Tiyajamorn et al.'s architecture, which is of size `embed_size`. This input embedding is fully connected to a hidden layer of size $2 \cdot$`embed_size` with ReLU Activation and Dropout. Finally, this hidden layer is fully connected to a final output of size `embed_size` which represents our meaning embedding improved for STS. Our Stage 2 loss function is the mean-squared error between our model's similarity scores (cosine similarity of sentence embeddings) and the human STS annotations. This simple architecture allowed us to improve meaning embeddings produced by Tiyajamorn et al.'s architecture.

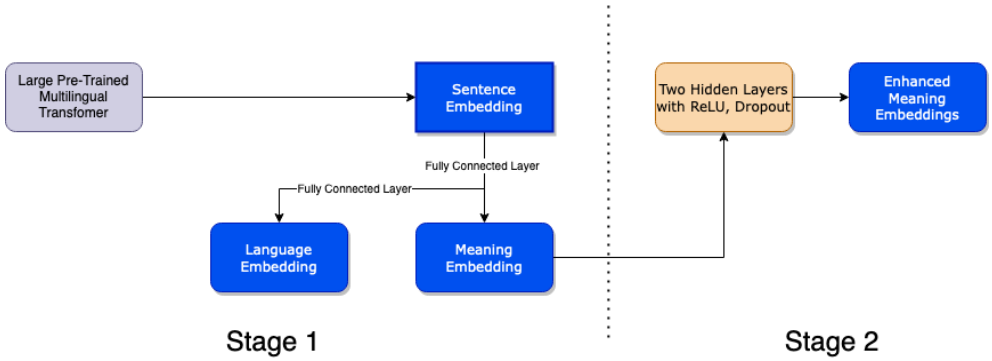


Figure 1: Stage 1 is a reimplementation of Tiyajamorn et al.'s [7] training method. Stage 2 is our addition to the architecture.

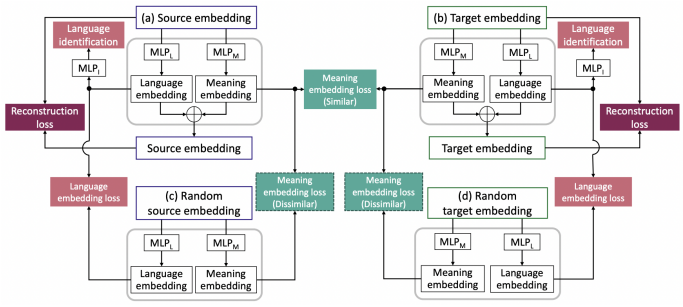


Figure 2: Tiyajamorn et al.'s [7] training method. Here, "source embedding" is the embedding of a sentence in the source language, and "target embedding" is the embedding of that same sentence in the target language. This is Figure 3 in Tiyajamorn et al. 2021.

Baselines. LaBSE[8], a multi-lingual pre-trained model, is our first baseline. Notably, LaBSE embeddings are intended to be language agnostic; two semantically-identical sentences, even if they are in different languages, should have similar LaBSE embeddings. However, because LaBSE is not specifically trained to perform cross-lingual STS, we expected it to have the poorest performance in our experiments. We used LaBSE from `sentence-transformers` via HuggingFace.[1]

XLM-R [9], also a multi-lingual pre-trained model, is our second baseline. Similar to LaBSE, embeddings produced by XLM-R are intended to be language agnostic. XLM-R is trained on

[1]From https://huggingface.co/sentence-transformers/LaBSE.

more languages, especially low-resource languages, than other models. The version of XLM-R we used is also fine-tuned to achieve high STS performance. Since the XLM-R architecture is similar to that of our own in the sense that it uses available labeled data to improve performance, we expected it to perform better than LaBSE. Furthermore, we did not expect that our multi-stage training pipeline would be able to significantly improve over XLM-R's performance. We used XLM-R from `sentence-transformers` via HuggingFace.[2]

We wrote the codebase entirely ourselves, leveraging PyTorch[10], SciPy[11], and Sentence Transformers[2].[3] Much of the time developing this codebase was spent optimizing performance. For example, we precomputed embeddings for all our parallel sentences and annotated sentence pairs using both LaBSE and XLM-R to enable fast training. We also implemented evaluation on the downstream STS task mostly ourselves, only using SciPy[11] to calculate the Pearson correlations.

5 Experiments

5.1 Data

Our approach leveraged two datasets. Following Tiyajamorn et al., we used the Tatoeba dataset [12] for parallel sentences across 15 different language pairs. We used this data to train our Stage 1 model. Figure 3 shows the number of parallel sentences per language pair in our dataset.

Second, we used the STSb Multi MT (SemEval 2017 Task 1) STS benchmark [1] for multilingual STS annotations to train our Stage 2 model. In particular, the SemEval 2017 Task 1 STS benchmark provides human-annotated sentence pairs for ten languages, consisting of 5,749 training samples, 1,500 dev samples, and 1,379 test samples for each language. We accessed this dataset through HuggingFace.[13][4] We did not have to perform any data cleaning.

Language Pair	Parallel Sentences
EN-AR	31,926
EN-DE	395,616
EN-ES	246,850
EN-FR	314,244
EN-IT	558,793
EN-NL	122,305
EN-PL	65,965
EN-RU	619,449
EN-TR	698,829
ES-ZH	10,814
FR-ES	57,097
RU-DE	140,108
TR-PL	935
UA-ES	25,731
ZH-RU	9,657

Figure 3: The number of unannotated parallel sentences per language pair in our training dataset for our Stage 1 model.

5.2 Evaluation Method

Following Tiyajamorn et al., we evaluated our models using the Pearson correlation coefficient between the cosine similarity of embeddings produced by our models and human annotators' similarity scores across 1,379 pairs of sentences in the test split of the SemEval 2017 STS Benchmark [13]. We calculated Pearson correlation coefficients using SciPy[11].

5.3 Experimental Details

We trained using the Adam optimizer with a learning rate of 1e-6. We trained our Stage 1 models for a maximum of 500 epochs, where each epoch included 5,000 steps (each containing a batch of 512 parallel sentences). For both Stage 1 and Stage 2, every epoch contained many training pairs; every language pair was weighted equally, irrespective of the number of training samples available for that language pair. To prevent overfitting, we used a patience of 50 epochs on the validation loss and, in our Stage 2 model, applied dropout during training between the first and second fully-connected layers.

We ran all our experiments atop both LaBSE and XLM-R. Notably, the version of XLM-R we used is a state-of-the-art model for multilingual STS and outperforms LaBSE; improving performance atop XLM-R is therefore significantly more challenging than atop LaBSE, and therefore tests the generality of our approach.

[2]From https://huggingface.co/sentence-transformers/stsb-xlm-r-multilingual.

[3]We used Tiyajamorn et al.'s codebase as a reference. Their code is available at https://github.com/nattaptiy/qe_disentangled.

[4]Note that the multilingual version of this dataset was created using machine translation.

We also experimented with adding a fully-connected "prelayer" in between the output of the pre-trained multilingual sentence encoder model and the first stage of the training pipeline. We also experimented with adding a 1D-convolutional layer between Stage 1 and Stage 2. Finally, to aid in calibration, we experimented with applying various non-linearities to the transformation from cosine similarity between embeddings to STS score for our mean-squared error loss in Stage 2. In particular, these non-linearities map cosine similarities in the range $[-1, 1]$ to STS scores which are in the range $[0, 5]$. In total, we trained and evaluated 16 models.

$$f(x) = 2.5(x+1) \qquad g(x) = \tanh^{-1}(x) + 2.5 \qquad h(x) = \text{sigmoid}^{-1}\left(\frac{x+1}{2}\right) + 2.5$$

Linear Map | Hyperbolic Tangent Non-Linear Map | Sigmoid Non-Linear Map

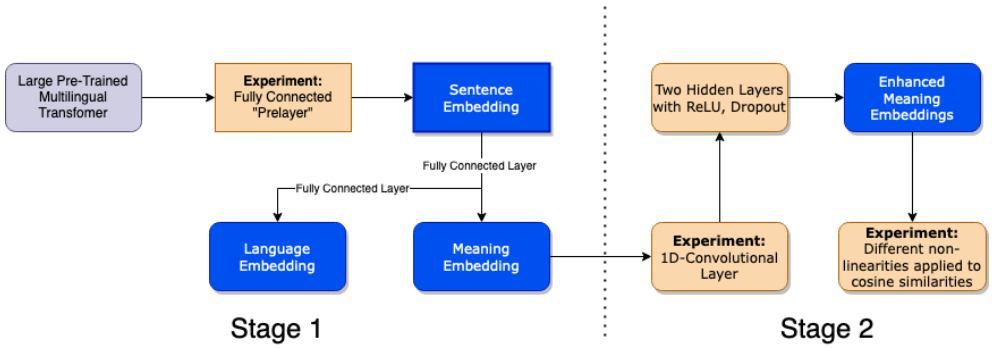


Figure 4: Stage 1 is a reimplementation of Tiyajamorn et al.'s [7] training method. Stage 2 is our addition to the architecture, though we also experimented with augmenting Stage 1.

5.4 Results

We observed strong results. Our best Stage 1 models improve the performance (as measured by the Pearson correlation on the STSb test set) over the LaBSE baseline across all language pairs by an average of 0.028, and over the strong XLM-R baseline by an average of 0.013. Our best Stage 2 models improve performance over the LaBSE baseline by an average of 0.06, and over the XLM-R baseline by an average of 0.012. Our Stage 1 model's performance improvement over the LaBSE baseline (0.028) exceeds the performance improvement reported by Tiyajamorn et al. over LaBSE (0.017).[5]

Notably, both our Stage 1 and Stage 2 models improve performance on EN-PT and PT-PL atop both LaBSE and XLM-R, despite not being trained on any PT (Portuguese) data.[6] Moreover, our models improve monolingual STS performance (e.g. EN-EN) despite not directly being trained on any monolingual STS data.

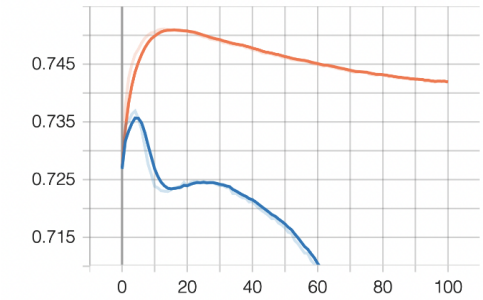
Table 1 shows the results of our models across several language pairs. We attribute these strong performance improvements — even on language pairs that our Stage 2 model was not directly trained on — in part to LaBSE's and XLM-R's already well-aligned representations. In particular, note that LaBSE and XLM-R were trained on every language with which we evaluate. Because LaBSE and XLM-R preserve sentence representations across translations, learning how to enhance meaning embeddings for multilingual STS in one language pair transfers to other language pairs as well.

6 Analysis

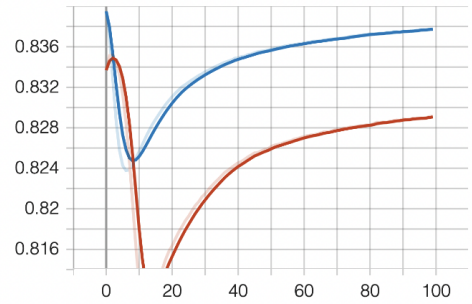
We found that the best-performing models typically had relatively lower levels of complexity. In particular, the model with the best STS performance on the test set was the simplest Stage 1 model atop on XLM-R. Our Stage 2 model (which augments embeddings using human-annotated STS data) improved STS performance on some language pairs, but additional layers of complexity did

[5] Tiyajamorn et al. did not run their architecture atop the version of XLM-R we used.

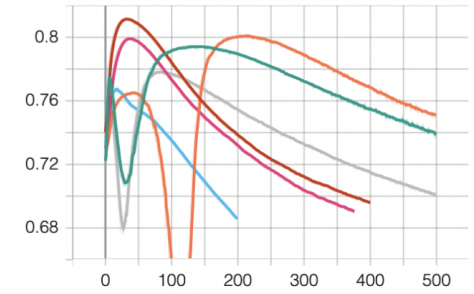
[6] Of course, XLM-R and LaBSE were trained on Portuguese data, making these transferable performance improvements possible.



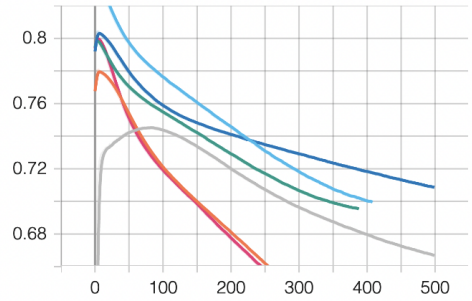
Stage 1 on XLM-R: Average Pearson STS performance over all evaluation language pairs on the dev set (y-axis) while training (x-axis) **with a prelayer** and **without a prelayer**.



Stage 1 on LaBSE: Average Pearson STS performance over all evaluation language pairs on the dev set (y-axis) while training (x-axis) **with a prelayer** and **without a prelayer**.



Stage 1+2 on XLM-R: Average Pearson STS performance over all evaluation language pairs on the dev set (y-axis) while training (x-axis) **without applying a non-linearity on cosine similarity and without a convolutional layer**, **with a hyperbolic tangent non-linearity and without a convolutional layer**, **with a sigmoid non-linearity and without a convolutional layer**, **without applying a non-linearity and with a convolutional layer**, with a hyperbolic tangent non-linearity and with a convolutional layer, **with a sigmoid non-linearity and with a convolutional layer**.



Stage 1+2 on LaBSE: Average Pearson STS performance over all evaluation language pairs on the dev set (y-axis) while training (x-axis) **without applying a non-linearity on cosine similarity and without a convolutional layer**, **with a hyperbolic tangent non-linearity and without a convolutional layer**, **with a sigmoid non-linearity and without a convolutional layer**, without applying a non-linearity and with a convolutional layer, **with a hyperbolic tangent non-linearity and with a convolutional layer**, **with a sigmoid non-linearity and with a convolutional layer**.

Figure 5: Learning curves for each of our experiments atop both XLM-R and LaBSE.

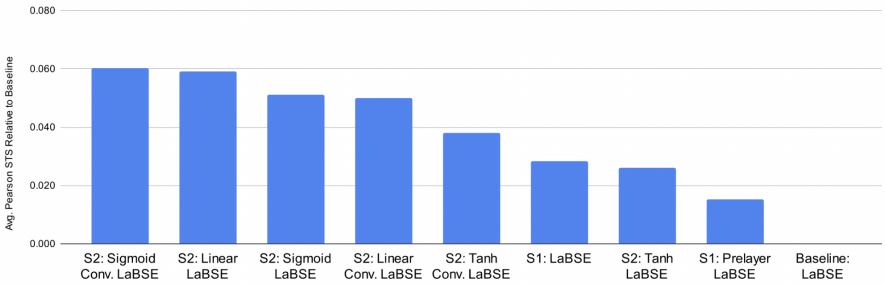


Figure 6: XLM-R: Relative Average Pearson STS scores on the STSb test set across experiments (n = 1,379). Zero represents XLM-R's baseline performance. S2 (Stage 2) models run atop the S1 (Stage 1) model without a prelayer. Note that EN-EN and EN-PT are held-out (we train our model on no monolingual or PT data).

not, including a fully connected "prelayer" in Stage 1, a 1-D convolutional prelayer in Stage 2, and non-linearities for cosine similarities in Stage 2. These changes did, however, improve performance atop LaBSE across all language pairs.

Model	Over Baseline	Average	EN-ES	ZH-RU	EN-EN	EN-PT	PT-PL
S1: XLM-R	**0.013**	**0.817**	**0.827**	**0.787**	0.848	**0.821**	**0.814**
S2: Linear XLM-R	0.012	0.816	**0.827**	0.784	**0.849**	0.820	0.811
S1: Prelayer XLM-R	0.008	0.812	0.823	0.784	0.845	0.816	0.808
Baseline: XLM-R	0.000	0.804	0.814	0.770	0.838	0.809	0.797
S2: Sigmoid Conv. XLM-R	-0.024	0.780	0.795	0.755	0.801	0.789	0.774
S2: Tanh XLM-R	-0.029	0.775	0.791	0.749	0.795	0.786	0.771
S2: Sigmoid XLM-R	-0.030	0.775	0.790	0.747	0.796	0.786	0.768
S2: Tanh Conv. XLM-R	-0.037	0.768	0.785	0.742	0.788	0.783	0.759
S2: Linear Conv. XLM-R	-0.102	0.702	0.724	0.651	0.770	0.715	0.678
S2: Sigmoid Conv. LaBSE	**0.060**	**0.762**	**0.774**	**0.730**	0.773	**0.778**	**0.750**
S2: Linear LaBSE	0.059	0.761	**0.774**	0.728	**0.788**	0.776	0.744
S2: Sigmoid LaBSE	0.051	0.753	0.769	0.711	0.766	0.772	0.737
S2: Linear Conv. LaBSE	0.050	0.752	0.762	0.723	0.774	0.760	0.743
S2: Tanh Conv. LaBSE	0.038	0.740	0.755	0.695	0.761	0.758	0.725
S1: LaBSE	0.028	0.730	0.739	0.707	0.738	0.733	0.735
S2: Tanh LaBSE	0.026	0.728	0.748	0.679	0.760	0.747	0.711
S1: Prelayer LaBSE	0.015	0.717	0.727	0.690	0.722	0.721	0.726
Baseline: LaBSE	0.000	0.702	0.720	0.649	0.727	0.710	0.696

Table 1: Overall Pearson STS results on the STSb test set for both Stage 1 and Stage 2 models atop LaBSE and XLM-R. Note that neither model was trained on any Portuguese (PT) data, nor were any models trained on monolingual data. The best performance per base model is shown in bold. "Over Baseline" is the difference between the model's average Pearson STS performance and that of the baseline. The average STS performance was computed across the following language pairs: EN-EN, EN-DE, EN-ES, EN-FR, EN-IT, EN-NL, EN-PL, EN-PT, EN-RU, RU-DE, FR-ES, ES-ZH, ZH-RU, and PT-PL.

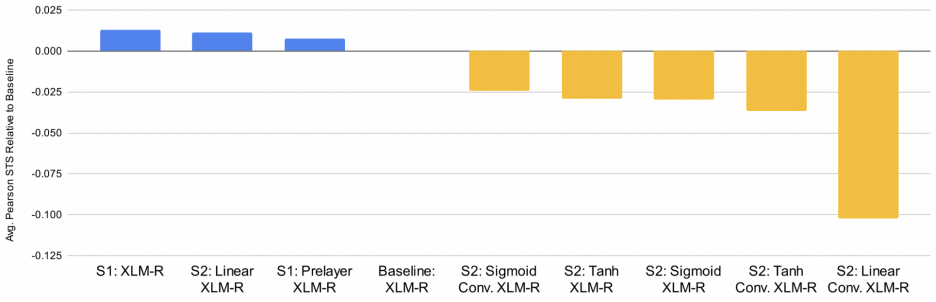


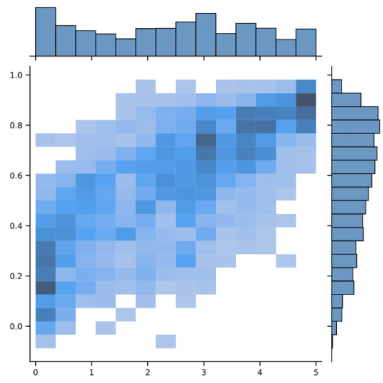
Figure 7: LaBSE: Relative Average Pearson STS scores on the STSb test set across experiments (n = 1,379). Zero represents LaBSE's baseline performance. S2 (Stage 2) models run atop the S1 (Stage 1) model without a prelayer. Note that EN-EN and EN-PT are held-out (we train our model on no monolingual or PT data).

These results aligned with our intuition. The version of XLM-R we used was fine-tuned to perform well specifically on multilingual STS, so it made sense that LaBSE's STS performance benefitted from extra complexity across all language pairs while XLM-R's performance only improved for some. In other words, adding extra layers of complexity causes XLM-R to quickly overfit since it already encodes so much useful context for STS.

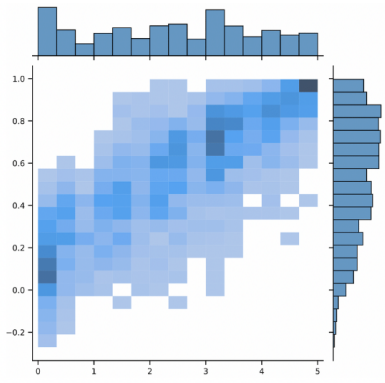
Our multi-stage training pipeline improved performance even on held-out language pairs.[7] As shown in Table 1, nearly all our models outperformed the baselines on EN-EN, EN-PT, and PT-PL, indicating that the model was able to learn more general structural patterns in sentence embeddings that were transferable across languages. Moreover, the model's STS performance improved roughly equally between language pairs including English (e.g. EN-PT) and language pairs not including English (e.g. PT-PL). This uniform improvement suggests the model was able to successfully produce aligned language-agnostic meaning embeddings in Stage 1 of the multi-stage training pipeline, thereby making the specific languages of the training data largely irrelevant in Stage 2.

[7]We use Portuguese as a stand-in for a low-resource language because we can only evaluate language pairs included in the STSb benchmark dataset, and none of the languages in that benchmark are low-resource.

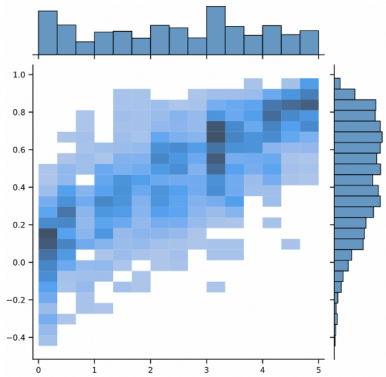
As Figure 8 illustrates, the model's computed similarity scores cluster around 0.75, while the human scores are uniform between 0 and 5. This opportunity for further calibration suggests further STS performance is possible with additional research.



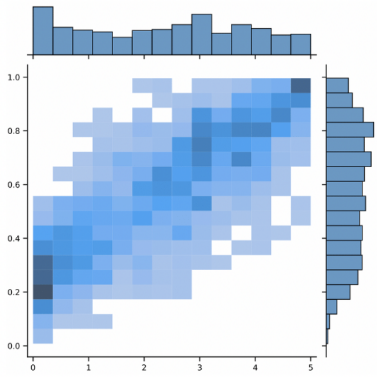
Stage 1 on LaBSE (no prelayer): Joint histogram of human annotations (x-axis) and our model's predicted STS scores across all evaluation language pairs. Average dev Pearson corr. across language pairs: **0.751** (same as Tiyajamorn et al.).



Stage 1 on XLM-R (no prelayer): Joint histogram of human annotations (x-axis) and our model's predicted semantic similarity scores across all evaluation language pairs. Average dev Pearson corr. across language pairs: **0.84** (+0.003 over XLM-R).



Stage 1+2 on LaBSE (no prelayer, linear): Joint histogram of human annotations (x-axis) and our model's predicted semantic similarity scores across all evaluation language pairs. Average Pearson corr. across language pairs: **0.803** (+0.09 over Stage 1).



Stage 1+2 on XLM-R (no prelayer, linear): Joint histogram of human annotations (x-axis) and our model's predicted semantic similarity scores across all evaluation language pairs. Average Pearson corr. across language pairs: **0.843** (+0.003 over Stage 1).

Figure 8: Joint plots comparing the human-generated STS scores to our models' computed similarity scores across the STSb dev set across the evaluation language pairs (listed beneath Table 1).

7 Conclusion

We have shown that making use of available human-annotated STS data through a multi-stage training pipeline can lead to improvements in STS performance atop state-of-the-art models. The effects of Stage 2 were more pronounced atop LaBSE, where Pearson STS scores on the STSb test set were higher across all language pairs. However, Stage 2 still yielded improvements for certain language pairs when using XLM-R as a base, which was notable given XLM-R was already directly trained to achieve high STS performance.

Overall, we present a novel approach for STS that enables use of all available data—both parallel corpora and human-annotated STS data—that improves performance even on held-out language pairs. Such a system is particularly useful for real-world downstream applications, where there are low-resource languages that have parallel corpora but few human STS annotations. As described earlier, much of the existing research in this area is "purist" in that it either only makes use of parallel corpora data or requires all supported languages to have human STS annotations. We hope our more nuanced and flexible approach will unlock new value, especially for low-resourced languages.

References

[1] Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. SemEval-2017 task 1: Semantic textual similarity multilingual and crosslingual focused evaluation. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 1–14, Vancouver, Canada, August 2017. Association for Computational Linguistics.

[2] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 11 2019.

[3] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[4] Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. Language-agnostic bert sentence embedding, 2020.

[5] Nils Reimers and Iryna Gurevych. Making monolingual sentence embeddings multilingual using knowledge distillation. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4512–4525, Online, November 2020. Association for Computational Linguistics.

[6] Jindřich Libovický, Rudolf Rosa, and Alexander Fraser. On the language neutrality of pre-trained multilingual representations. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1663–1674, Online, November 2020. Association for Computational Linguistics.

[7] Nattapong Tiyajamorn, Tomoyuki Kajiwara, Yuki Arase, and Makoto Onizuka. Language-agnostic representation from multilingual sentence encoders for cross-lingual similarity estimation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7764–7774, Online and Punta Cana, Dominican Republic, November 2021. Association for Computational Linguistics.

[8] Fangxiaoyu Feng, Yinfei Yang, Daniel Cer, Naveen Arivazhagan, and Wei Wang. Language-agnostic bert sentence embedding, 2020.

[9] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale, 2020.

[10] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

[11] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020.

[12] Tatoeba: Collection of sentences and translations. https://tatoeba.org.

[13] Philip May. Machine translated multilingual sts benchmark dataset. 2021.