Hyperboloid embeddings and Hypersphere similarity metrics for Sentence-BERT

Garvit Goyal ggoyal9@gatech.edu
Gunjan Gupta ggupta68@gatech.edu
Tusheet Goli tgoli3@gatech.edu

Abstract

The architecture of the SBERT model was explained in the paper, *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks* (Reimers and Gurevych, 2019). SBERT model has a siamese network architecture and uses similarity measures such as cosine similarity/Euclidean/Manhattan distance to derive fixed-size vector embeddings for sentence models. It has been shown that SBERT outperformed other state-of-art sentence transformer models on common STS tasks. For this project, we intend to alter the SBERT model by changing the model architecture and similarity metrics.

1 Introduction

Semantic textual similarity is the task of measuring the closeness of the semantic meaning of two text snippets. We train models to get vector embeddings such that semantically similar sentences are closer in the vector space, thereby deriving meaningful sentence embeddings. This has further applications in other tasks such as semantic search, clustering, paraphrase mining, question answering (retrieve and re-rank) and image search. Having a reliable and computationally efficient sentence embedding model can help to improve the above tasks as well.

The SBERT model is the state-of-art sentence embedding model that is a computationally efficient way to get sentence embeddings. We expect that such a model is able to cluster semantically similar sentences together in vector space. Below is an example of the output from the all-miniLM-L6-v2 (SBERT model). We compare the cosine similarity of three sentences with the reference sentence in Table 1.

Reference Sentence: That is a happy person.

The model does not significantly differentiate between a happy girl and a happy dog. Our goal is to improve the performance of the SBERT model by 1) changing the architecture of the model while going from BERT word embeddings to sentence embeddings and 2) exploring other similarity metrics than the distance that might be more suited for the task.

Sentence	Similarity Score
That is a very happy person	0.943
That is a happy dog	0.665
That is a happy girl	0.685

Table 1: Example of semantic similarity between sentences from SBERT model

2 Related Work

BERT for sentence classification and pair regression tasks uses sentence pairs to predict the target value. However, this technique requires $\mathbb{O}(n^2)$ computations and is unable to compute independent sentence embeddings. To overcome this, Reimers and Gurevych, 2019. proposed new models in their paper *Sentence-BERT Sentence Embeddings using Siamese BERT-Networks* that used the average of the BERT output layer or [CLS] token embedding to represent sentences as fixed-size vectors. In general, this leads to poor performance, worse than averaging GloVe embeddings. Compared to the other neural sentence embedding models such as SkipThought, InferSent, and Universal Sentence Encoder, the training time of SBERT is lower as it uses a pretrained BERT network and fine-tunes the model to get sentence embeddings.

The model pools the output of word embeddings from the BERT model to derive sentence embeddings. It uses a siamese or triplet network to finetune the BERT model such that semantically similar sentences are embedded close to each other in the vector space. The input can be informed of sentence pairs or triplets. The objective depends on the dataset available and can be of the following types - classification objective, regression objective, and triplet objective. In this model, Cosine/Euclidean/Manhattan distances can be used as a measure of similarity between vector embeddings. The contrastive loss function introduced by Hadsell et al., 2006 is used here, works to reduce the distance between similar pairs and increase the distance between dissimilar ones. The authors evaluate the model performance on supervised, and unsupervised STS tasks, and SentEval tasks and show that it is able to outperform other sentence embedding models. It is also computationally efficient than InferSent and Universal Sentence Encoder.

The paper presents a computationally efficient way of

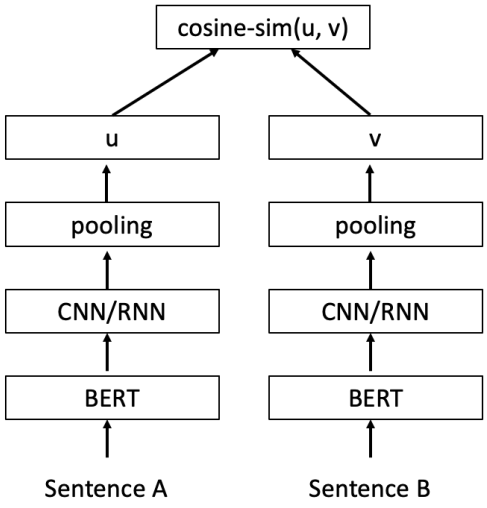

Figure 1: Proposed architecture for modified SBERT model

learning sentence embeddings through contrastive learning. The model in general outperforms other sentence embedding models, however, it appears to depend significantly on the training corpus. For instance, SBERT does not perform as well on the TREC dataset compared to Universal Sentence Encoder which was trained on question-answer type data. The SBERT model is also not suited for transfer learning for other STS tasks, which makes it restrictive in the application. While there are some limitations to this paper, it describes the detailed implementation of the SBERT model and provides a fundamental understanding of the model we want to implement. Thus, by gaining a deeper understanding of the model described in this paper, we can make our own siamese network to improve sentence embedding models.

3 Methods

3.1 Modified SBERT Architecture

The SBERT model architecture as described in (Reimers and Gurevych, 2019) generates word embedding using BERT model and uses attention mask based average pooling to derive sentence embeddings. We hypothesize that using a neural network with BERT word embeddings as input and then pooling them to get sentence embeddings might help to improve their quality. Figure 1 describes our proposed changes to the SBERT model.

3.2 Loss Functions

3.2.1 Uniformity and Alignment

Wang and Isola, 2020 introduce two metrics associated with alignment (minimizing distance between embeddings associated with similar samples, i.e., positive pair) and uniformity (feature vectors being uniformly distributed on the unit hypersphere, preserving maximal information). They show, empirically and theoretically, that the contrastive loss optimizes alignment and uniformity asymptotically in the limit of infinite negative samples.

The alignment metric computes the expected distance between positive pairs. On the other hand, the uniformity metric computes the logarithm of the average pairwise Gaussian potential using the Gaussian potential kernel. The latter has been further demonstrated to be similar to the uniform distribution on the hypersphere and universally optimal point configurations. The two metrics thus introduced are as follows:

Let $p_{data}(\cdot)$ be the data distribution over $\mathbb{R}^n$ and $p_{pos}(\cdot,\cdot)$ the distribution of positive pairs over $\mathbb{R}^n \times \mathbb{R}^n$.

$$\mathcal{L}_{align}(f;\alpha) \triangleq \mathbb{E}_{(x,y)\sim p_{pos}}[\|f(x)-f(y)\|_2^{\alpha}] \quad (1)$$

$$\mathcal{L}_{uniform}(f;t) \triangleq \log \mathbb{E}_{(x,y)\overset{\text{i.i.d.}}{\sim} p_{data}}[\exp(-t\|f(x)-f(y)\|_2^2)] \quad (2)$$

where $\alpha > 0, t > 0$, the latter being the parameter of the Gaussian kernel.

3.2.2 Hyperboloid Embeddings

Hyperbolic geometry is non-Euclidean geometry dealing with spaces with constant negative curvature. Nickel and Kiela, 2017b show that Poincare embeddings in hyperbolic space for WordNET outperform Euclidean and translation embeddings in representation capacity and generalization performance. Further, to avoid numerical instability in Poincare distance computations, as also noted in Chami et al., 2019, Nickel and Kiela, 2018 extend their approach to the unit hyperboloid model in hyperbolic space and compute the distance between two embeddings as given below.

Let $\mathbf{x}, \mathbf{y} \in \mathbb{R}^{n+1}$ and let

$$\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}} = -x_0 y_0 + \sum_{i=1}^{n} x_i y_i \quad (3)$$

Thus, the associated distance function in the unit hyperboloid space is given by

$$d_l(\mathbf{x}, \mathbf{y}) = \operatorname{arcosh}(-\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}}) \quad (4)$$

Further, Law et al., 2019 extend this approach to the general hyperboloid model with the following formulation for the hyperboloid models $\mathcal{H}^{d,\beta} \subseteq \mathbb{R}^{d+1}$.

$$\mathcal{H}^{d,\beta} := \{\mathbf{a} = (a_0, .., a_d) \in \mathbb{R}^{d+1} : \|\mathbf{a}\|_{\mathcal{L}}^2 = -\beta, a_0 > 0\} \quad (5)$$

where $\beta > 0$ and $\|\mathbf{a}\|_{\mathcal{L}}^2 = \langle \mathbf{a}, \mathbf{a} \rangle_{\mathcal{L}}$.

The squared Lorentzian distance defined for all pairs $\mathbf{a}, \mathbf{b} \in \mathcal{H}^{d,\beta}$ as:

$$d_{\mathcal{L}}^2(\mathbf{a}, \mathbf{b}) = \|\mathbf{a} - \mathbf{b}\|_{\mathcal{L}}^2 = -2\beta - 2\langle \mathbf{a}, \mathbf{b} \rangle_{\mathcal{L}} \quad (6)$$

This approach circumvents the need for specialized stochastic gradient descent in hyperboloid space as domain of (4) cannot be considered a vector space. Indeed,

	train	dev	test	total
news	3299	500	500	4299
caption	2000	625	625	3250
forum	450	375	254	1079
total	**5749**	**1500**	**1379**	**8628**

Table 2: Breakdown of train-dev-test splits across genres

$d_l(\mathbf{x}, \mathbf{y})$ is undefined for pairs satisfying $\langle \mathbf{x}, \mathbf{y} \rangle_{\mathcal{L}} > -1$. Instead, squared Lorentzian distance (6) is dependent only on well-defined Lorentzian inner product (3) for which directional derivatives exist, and thus standard SGD can be used in practice to compare two embeddings mapped to the hyperbolic space. We plan to use the distance given in (6) in the loss functions described in Section 3.2 to evaluate the effectiveness of hyperboloid embeddings in this task. Specifically, we consider the invertible mapping $g_\beta : \mathbb{R}^d \to \mathcal{H}^{d,\beta} \; \forall \, \mathbf{x} = (x_1, ..., x_d) \in \mathbb{R}^d$ as:

$$g_\beta(\mathbf{x}) \coloneqq (\sqrt{\|\mathbf{x}\|^2 + \beta}, x_1, ..., x_d) \in \mathcal{H}^{d,\beta} \quad (7)$$

We can then compare two examples $\mathbf{x}, \mathbf{y} \in \mathcal{R}^d$ using (6) and (7) as:

$$\begin{aligned} d^2_{\mathcal{L}}(g_\beta(\mathbf{x}), g_\beta(\mathbf{y})) &= -2\beta - 2\langle g_\beta(\mathbf{x}), g_\beta(\mathbf{y}) \rangle_{\mathcal{L}} \\ &= -2[\beta + \langle \mathbf{x}, \mathbf{y} \rangle - \sqrt{\|\mathbf{x}\|^2 + \beta}\sqrt{\|\mathbf{y}\|^2 + \beta}] \end{aligned} \quad (8)$$

4 Data

We use the STSb (Semantic Textual Similarity Benchmark) dataset for training and evaluation for the following two methods - (1) changing the architecture by introducing CNN layer after the BERT word embeddings, (2) changing the loss function using hyperboloid embeddings. The STSb dataset is a selection of English datasets that have been primarily used as a benchmark dataset for SemEval between 2012 and 2017. It includes text from image captions, news headlines, and user forums. The dataset consists of 8628 sentence pairs from multiple genres which are divided into a train, dev, and test splits.

The distribution of genres between the train-dev-test splits can be seen in Table 2.

An example of the dataset can be seen in Table 3.

For the method of using uniformity and alignment as similarity metrics in our loss function, we need binary labeled dataset of sentence pairs. For this task we use the SNLI dataset that contains 570,000 sentence pairs labeled as entailment, neural and contradiction. This is a publicly available dataset https://nlp.stanford.edu/projects/snli/ that offers the corpus as both JSON lines (jsonl) and a tab-separated text file. It is about 100MB and can be downloaded from https://nlp.stanford.edu/projects/snli/snli_1.0.zip. From the corpus we will filter out sentences labeled entailment (or positive pairs) and contradiction (or negative pairs) and use these to calculate the uniformity and alignment metrics. An example of this dataset can be seen in Table 4

sentence1	sentence2	similarity score
A plane is taking off.	An air plane is taking off.	5
Three men are playing chess.	Two men are playing chess.	2.6
A man is smoking.	A man is skating.	0.5
A man pouts oil into a pot.	A man pours wine into a pot.	3.2
A man is playing a guitar.	A girl is playing a guitar.	2.8

Table 3: Example of the STSb dataset

Premise	Hypothesis	Label
A soccer game with multiple males playing.	Some men are playing a sport.	entailment
An older and younger man smiling.	Two men are smiling and laughing at the cats playing on the floor.	neutral
A man inspects the uniform of a figure in some East Asian country.	The man is sleeping.	contradiction

Table 4: Example of the SNLI dataset

5 Evaluation

To evaluate the performance of our model, we used Spearman's rank correlation metric. Spearman's rank correlation is an effective way to measure the strength and direction of association between two ranked variables. It utilizes a monotonic function to effectively evaluate the relationship between two variables. Since our goal is to measure the performance for the estimated similarity between a pair of sentences in the STSb dataset, this would be a great and applicable metric for us to use. It can also be used for variables that are not normal-distributed and have a non-linear relationships.

The formula for Spearman's rank correlation is given as follow:

$$\rho = 1 - \frac{6 \sum d_i^2}{n(n^2 - 1)}$$

where ρ is the Spearman's rank correlation coefficient, d_i is the difference between the two ranks of each observation, and n is the number of observations.

In the paper *Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks* by Reimers and Gurevych, 2019, it was noted that the Pearson corre-

lation does not work well for STS (Semantic Textual Similarity) tasks. Since this is our benchmark paper that we want to improve the evaluation on, we plan on using the same evaluation metric, i.e. Spearman's rank correlation, to evaluate our novel model to obtain a more comparative analysis between the two models. Specific to our task, we use Spearman's rank correlation between the cosine similarity of the sentence embeddings and the gold labels, similar to what is mentioned by Reimers and Gurevych, 2019.

On the other hand, the average F1 score metric is used to evaluate the SNLI dataset since this is a binary classification. The cosine similarity of the sentence pairs is calculated and based on the f1 scores for the different thresholds, the average F1 score is calculated to indicate the performance on the SNLI dataset.

6 Experiments

For our midterm report, we tested models with modified architecture. The architecture with BERT followed by mean pooling is our baseline. We used pretrained bert-based-uncased model for obtaining word embeddings. The maximum sequence length for the BERT input is 512 and the dimension of word embeddings is 768. Then we make the following changes after the word embedding from BERT are obtained -

1. *CNN followed by mean pooling*: The number of output channels was set equal to 256 (the embedding dimension), and kernel size = [1, 3, 5]. The output embeddings for the sentence were then passed to the mean pooling layer to get the sentence embedding.

2. *CNN followed by max pooling*: The number of output channels was set equal to 256 (the embedding dimension), and kernel size = [1, 3, 5]. The output embeddings for the sentence were then passed to the max pooling layer to get the sentence embedding.

We used the above models to build a siamese network and finetune the model using contrastive loss function. The contrastive function uses cosine distance as a similarity measure between vector embeddings of sentences. We used Adam optimizer with learning rate = $2e^{-05}$ and trained the models for 10 epochs.

7 Preliminary Results

The inclusion of CNN before pooling the words' embeddings to estimate the sentence's embeddings results achieves similar cosine Spearman as the baseline, seen in Table 5. Thus, the preliminary experiments indicate that inclusion of CNN before pooling does not lead to a significant improvement in SBERT's performance, and thus we won't be modifying the pooling layer in this work.

We can also see that just using CNN (BERT-STSB-CNN + Max Pooling) to get sentence embedding by convolving the word embeddings yields inferior results. This might be because the mean pooling in the model makes use of attention mask which is better able to capture the contribution of individual words to the overall semantics of the sentence rather than just the convolutional layer.

Model	Spearman
BERT-STSB-Baseline	0.8394
BERT-STSB-CNN + Mean Pooling	0.8396
BERT-STSB-CNN + Max Pooling	0.8243

Table 5: Evaluation results on the STS benchmark test set. All models were trained for 10 epochs, fine-tuned on the STSb dataset

8 Future Work

So far we have experimented with changing the model architecture and adding a CNN layer in between the BERT model and pooling layer. However, our results indicate that there is no significant change in the model performance. We now intend to study the effect of changing loss function using different similarity metric (uniformity and alignment) and using hyperboloid embeddings.

9 Source/Version Control

We created a private repository on Gatech GitHub to store our data, code, analysis, and other documents. We have added all our team member as collaborators to the repository and will add the TAs and Professor if required.

This is the link to our repository - `https://github.gatech.edu/tgoli3/gt-nlp-f22`

References

Samuel R. Bowman, Gabor Angeli, Christopher Potts, and Christopher D. Manning. 2015. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 632–642, Lisbon, Portugal. Association for Computational Linguistics.

Ines Chami, Zhitao Ying, Christopher Ré, and Jure Leskovec. 2019. Hyperbolic graph convolutional neural networks. *Advances in neural information processing systems*, 32.

Qian Chen, Zhen-Hua Ling, and Xiaodan Zhu. 2018. Enhancing sentence embedding with generalized pooling. *CoRR*, abs/1806.09828.

Liat Ein Dor, Yosi Mass, Alon Halfon, Elad Venezian, Ilya Shnayderman, Ranit Aharonov, and Noam Slonim. 2018. Learning thematic similarity metric

from article sections using triplet networks. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 49–54, Melbourne, Australia. Association for Computational Linguistics.

R. Hadsell, S. Chopra, and Y. LeCun. 2006. Dimensionality reduction by learning an invariant mapping. In *2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06)*, volume 2, pages 1735–1742.

Marc Law, Renjie Liao, Jake Snell, and Richard Zemel. 2019. Lorentzian distance learning for hyperbolic representations. In *International Conference on Machine Learning*, pages 3672–3681. PMLR.

Maximilian Nickel and Douwe Kiela. 2017a. Poincaré embeddings for learning hierarchical representations. *CoRR*, abs/1705.08039.

Maximillian Nickel and Douwe Kiela. 2017b. Poincaré embeddings for learning hierarchical representations. *Advances in neural information processing systems*, 30.

Maximillian Nickel and Douwe Kiela. 2018. Learning continuous hierarchies in the lorentz model of hyperbolic geometry. In *International Conference on Machine Learning*, pages 3779–3788. PMLR.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *CoRR*, abs/1908.10084.

Tongzhou Wang and Phillip Isola. 2020. Understanding contrastive representation learning through alignment and uniformity on the hypersphere. In *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 9929–9939. PMLR.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.