# Space Decomposition for Sentence Embedding

**Wuttikorn Ponwitayarat**[*†], **Peerat Limkonchotiwat**[*†],
**Ekapol Chuangsuwanich**[‡], **Sarana Nutanong**[†]
[†]School of Information Science and Technology, VISTEC, Thailand
[‡]Department of Computer Engineering, Faculty of Engineering,
Chulalongkorn University, Thailand
{wuttikorn.p_s22,peerat.l_s19,snutanon}@vistec.ac.th
ekapolc@cp.eng.chula.ac.th

## Abstract

Determining sentence pair similarity is crucial for various NLP tasks. A common technique to address this is typically evaluated on a continuous *semantic textual similarity* scale from 0 to 5. However, based on a linguistic observation in STS annotation guidelines, we found that the score in the range [4,5] indicates an upper-range sample, while the rest are lower-range samples. This necessitates a new approach to treating the upper-range and lower-range classes separately. In this paper, we introduce a novel embedding space decomposition method called *MixSP* utilizing a *Mixture of Specialized Projectors*, designed to distinguish and rank upper-range and lower-range samples accurately. The experimental results demonstrate that MixSP decreased the overlap representation between upper-range and lower-range classes significantly while outperforming competitors on STS and zero-shot benchmarks. [1]

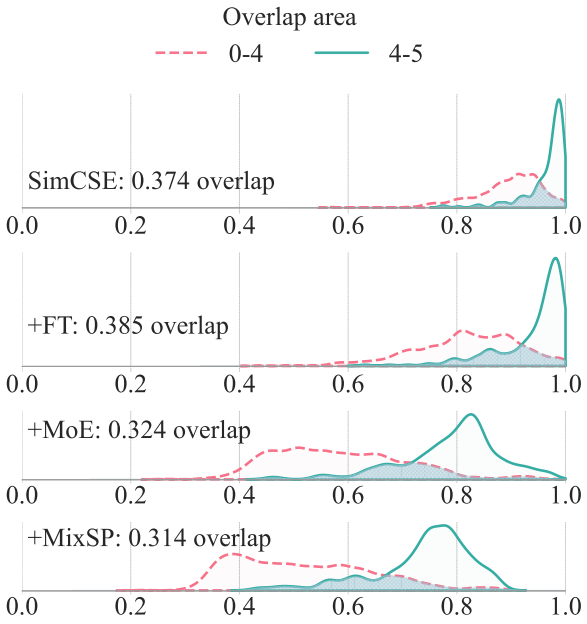



Figure 1: Cosine similarity distributions for BERT-Base formulated on Gaussian estimation. The overlap value refers to the intersection between the upper and lower ranges. We use the data from the CDSC-R test data.

## 1 Introduction

Determining the similarity between sentence pairs is fundamental to many downstream applications such as text classification, search, and ranking. Usually, sentence pair similarity is assessed via *Semantic Textual Similarity (STS)*, where each sample contains a pair of sentences, and their label denotes the degree of similarity, which uses scores from 0 to 5, where 5 represents the highest degree of similarity. Studies have shown that improving the ability to rank sentence pairs according to their similarities enhances text classification accuracy (Gao et al., 2021; Limkonchotiwat et al., 2022; Miao et al., 2024) and reranking mean-average precision (Wang et al., 2021).

A common approach to solving the STS problem is employing a pre-trained language model (Devlin et al., 2019; Liu et al., 2019) and finetuning it with a supervised learning objective. In particular, we aim to construct an embedding space in which the cosine similarity between sentence pairs reflects the degree of similarity; contrastive learning is a popular method to achieve such an embedding space (Yan et al., 2021; Gao et al., 2021; Jiang et al., 2022a; Wang and Lu, 2022). Regardless of the differences in their data augmentation strategies, all aforementioned methods treat the degree of similarity as a continuous range. In other words, all learning methods indifferently treat sentence pairs with different degrees of similarities within the STS score range [0,5].

In this paper, we challenge the common practice of treating STS scores as a continuous spectrum. Several studies observed that the score range [4,5] signifies semantically related (i.e., upper-range) samples, while the rest represents unrelated (i.e., lower-range) samples (Gao et al., 2021; Chuang et al., 2022). Consequently, the STS problem

[*]Equal contributions
[1]The code and models are available at https://github.com/KornWtp/MixSP.



*Findings of the Association for Computational Linguistics: ACL 2024*, pages 11227–11239
August 11-16, 2024 ©2024 Association for Computational Linguistics

should be considered a ranking within two distinct classes rather than one continuous spectrum.

We introduce a novel embedding space decomposition method called *MixSP* utilizing a *Mixture of Specialized Projectors*. The novelty of MixSP lies in a carefully designed learning pipeline with the following traits: (i) the ability to distinguish upper-range from lower-range samples and (ii) the ability to accurately rank sentence pairs within each class. In particular, our method uses a routing network and two specialized projectors to handle upper-range and lower-range representations, resulting in a better STS performance overall.

Figure 1 illustrates how our embeddings can better distinguish different sentence pairs as compared to our competitors: SimCSE (Gao et al., 2021), FT (Reimers and Gurevych, 2019), and MoE (Zhou et al., 2022b) compared to that of MixSP. We quantify the confusion between upper-range and lower-range classes as the cosine score overlaps between these two classes using Gaussian kernel density estimates. A smaller overlap indicates the ability to distinguish the upper-range and lower-range classes. We can see that MixSP obtains the smallest overlap between upper-range and lower-range classes of 31.4% while all competitors have an overlap ranging from 32.4% to 38.5%. Regarding the similarity ranking performance, our method also produces superior performance compared to these competitors (as shown in Figure 3, Section 5.3). These results demonstrate that our method improves the ability to distinguish upper-range and lower-range samples and rank sentence pairs according to their similarities.

Our contributions are as follows:

- We have recast the sentence embedding paradigm from one embedding space containing upper-range and lower-range to separate embedding space for each group.
- We propose a novel embedding space decomposition technique called *Mixture of Specialized Projectors* (*MixSP*). Our model has the ability to distinguish upper-range and lower-range samples while accurately ranking sentence pairs within each class.
- We demonstrate the efficiency of our method on STS and zero-shot benchmarks. In addition, we provide deep analyses of (i) performance efficiency and (ii) design choice in embedding space decomposition settings.

## 2 Related Work

### 2.1 Sentence Representation

Currently, researchers typically use pre-trained language models and supervised contrastive learning to train sentence representation models. The main goal of contrastive learning is to maximize the similarity between anchor and positive while minimizing the similarity between anchor and negative. The key component of contrastive learning is data augmentation for positive and negative pairs. Gao et al. (2021) proposed SimCSE, a contrastive learning for sentence embedding. SimCSE used dropout masks in two forward passes as the data augmentation method.

Jiang et al. (2022a) proposed PromptBERT, a prompt-based sentence embedding. PromptBERT used contrastive learning with template denoising to generate positive pairs, while negative pairs are sentences within the same mini-batch.

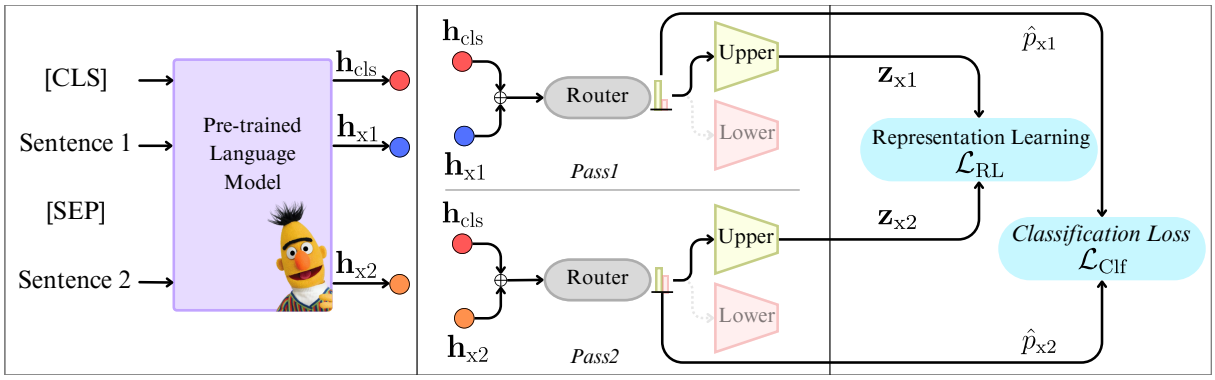
Wang and Lu (2022) proposed DiffAug, a two-stage training objective. The training objective of DiffAug is similar to SimCSE, but DiffAug used a contextual prompt to produce a hard positive, improving the generalizability of the embedding space.

Additionally, other works used various augmentation to obtain augmented texts. Notable techniques include back-translation (Fang et al., 2020; Limkonchotiwat et al., 2022, 2023), MLM (Yang et al., 2021; Chuang et al., 2022), and prompting (Zhou et al., 2022a; Wang et al., 2022; Jiang et al., 2022a,b).

### 2.2 Embedding Space Decomposition

Embedding space decomposition is the task of partitioning data into distinct subsets within the embedding space, enhancing model performance and understanding through focused representations. A common technique is to separate the space with semantic features. Wang et al. (2020) proposed semantic subspace analysis to break down the high-dimensional embedding space into semantic groups and examine their interrelationships. Opitz and Frank (2022) decomposed the embedding space to unveil interpretable semantic features within sentence embeddings, targeting semantic roles, negation, and quantification for a deeper understanding of the conveyed meaning.

Recently, researchers employed Mixture-of-Experts (MoE), which partitions the space into smaller subspaces managed by specialized experts,




(a) Sentence Pair Encoding (b) Embedding Space Decomposition (c) Loss Functions

Figure 2: The overview of *Mixture of Specialized Projectors (MixSP)*. (a) Given an upper-range sample, we encode the sample with a pre-trained language model. (b) We use a router to classify a class of sentences 1 and 2 (upper-range or lower-range). The final representation is formulated by projecting the representation with specialized projectors. (c) We improve the classification and representation with our training losses $\mathcal{L}_{\text{Clf}}$ and $\mathcal{L}_{\text{RL}}$, respectively.

to handle varied data aspects (Li et al., 2023; Chen et al., 2023; Chowdhury et al., 2023). The key components of MoE comprise *routing networks* that classify embedding space weights and aggregate embedding spaces from *expertise networks* to formulate the final representation.

While these works offer a decomposed learning paradigm, their approaches are inapplicable to the linguistic property of sentence relationship prediction. In particular, we must address the following issues to formulate a suitable space decomposition method. First, prior works lack an explicit control mechanism to differentiate between types of concern (upper-range and lower-range classes in this case). Second, the space embedding decomposition is only performed in the training step while aggregating the embedding space at the inference stage. In the following section, we propose a solution that addresses these limitations to reflect the problem requirements.

## 3 Mixture of Specialized Projectors

We design our method, *Mixture of Specialized Projectors (MixSP)*, based on our linguistic observation that when labeling sentence pairs for similarity, the score ranges of [0,4) and [4,5] are considered as two distinct classes: lower-range and upper-range, respectively. In particular, we address our space decomposition problem by designing a classify-and-rank pipeline with a routing mechanism and one specialized projector for each class. Consequently, we improve (i) the ability to differentiate between upper-range and lower-range sentence pairs and (ii) the ranking performance within each, thereby uplifting the overall sentence similarity prediction performance.

Figure 2 displays our classify-and-rank pipeline consisting of the following components:

- The cross-encoder setup that transforms input sentence pairs into vectors (Section 3.1).
- The space decomposition mechanism that differentiates and separately handles upper-range and lower-range classes (Section 3.2).
- The training objective that improves the ranking consistency within the embedding space of each class (Section 3.3).

### 3.1 How Do We Encode Sentence Pairs?

As shown in Figure 2, given a sentence-pair (`sent1`,`sent2`), we input them to a cross-encoder architecture from a pre-trained language model as `[CLS]sent1[SEP]sent2[SEP]` and obtained three embeddings:

- $\mathbf{h}_{\text{cls}}$ is the embedding of `[CLS]` from the last layer of the pre-trained language model.
- $\mathbf{h}_{\text{x1}}$ is the mean pooling of `sent1`'s representation.
- $\mathbf{h}_{\text{x2}}$ is the mean pooling of `sent2`'s representation.

### 3.2 How Do We Decompose the Embedding Space?

The embedding space decomposition mechanism consists of the routing network and specialized projectors. These two parts are explained in the following subsections.

#### 3.2.1 The Routing Network

As discussed in the related work section, the routing network can be used as a technique to decompose an embedding space. Unlike existing methods, however, we introduce the `[CLS]` label into the routing network in addition to the contextual representation $\mathbf{h}_{\text{x}}$. This additional information allows the routing mechanism to understand the relation between the sentence pair (`sent1`,`sent2`). In particular, by incorporating the global representation, our sentence-pair input is the element-wise addictions: $\mathbf{h}_{\text{cls}} \oplus \mathbf{h}_{\text{x}}$. We formulate the routing network as a group classification from a linear layer $\text{G}_1(\cdot)$, whether $\mathbf{h}_{\text{cls}} \oplus \mathbf{h}_{\text{x1}}$ and $\mathbf{h}_{\text{cls}} \oplus \mathbf{h}_{\text{x2}}$ are the representation of upper-range or lower-range groups based on the softmax probability $\hat{p}_{\text{x}j}$:

$$\hat{p}_{\text{x}j} = \text{SoftMax}(\text{G}_1(\mathbf{h}_{\text{cls}} \oplus \mathbf{h}_{\text{x}j})), \quad (1)$$

where $j$ is `sent1` ($j$=1) or `sent2` ($j$=2). We calculate the softmax probability $\hat{p}_{\text{x1}}$ and $\hat{p}_{\text{x2}}$ from `sent1` $\mathbf{h}_{\text{x1}}$ and `sent2` $\mathbf{h}_{\text{x2}}$, respectively.

To assist the routing network in classifying the input, we employ a binary cross-entropy (BCE) as follows.

$$\mathcal{L}_{\text{Clf}} = \frac{1}{2}\text{BCE}(\hat{y}, \hat{p}_{\text{x1}}) + \frac{1}{2}\text{BCE}(\hat{y}, \hat{p}_{\text{x2}}), \quad (2)$$

where $y$ is a gold label indicating whether the sentence pair is upper-range or lower-range. The classification loss $\mathcal{L}_{\text{Clf}}$ is used as part of the overall learning objective explained in Section 3.3.

#### 3.2.2 Specialized Projectors

The main goal of MixSP is to decompose the representation $\mathbf{h}_{\text{x}j}$ into upper-range or lower-range subspaces. Note that previous works in embedding space decomposition produce a composite representation, i.e., obtaining the final representation by computing the vector summation from multiple projectors' outputs. We found that such a soft-selection approach results in overlaps between subspaces, which is detrimental to the model's performance. Consequently, we derive a hard-selection process in which our specialized projectors have a separate projection for each class and use only one head per representation. In particular, our two specialized projectors, (i) an upper-range projector $\text{Upper}(\cdot)$ and (ii) a lower-range projector $\text{Lower}(\cdot)$, map representations $\mathbf{h}_{\text{x}j}$ to upper-range or lower-range subspaces as follows:

$$\mathbf{z}_{\text{x}j} = \begin{cases} \text{Upper}(\mathbf{h}_{\text{x}j}) * \beta_{\text{j}}, & \text{if argmax}(\hat{p}_{\text{x}j}) = 0 \\ \text{Lower}(\mathbf{h}_{\text{x}j}) * \beta_{\text{j}}, & \text{otherwise} \end{cases} \quad (3)$$

where $\beta_j$ is the highest probability of $\hat{p}$ obtained from $\max(\hat{p}_{\text{x}j})$ and $\mathbf{z}_{\text{x}j}$ is the representation that mapped to upper-range or lower-range subspaces.

With the output from different projectors, we obtain the representation pair of upper-range and lower-range samples separately, $\mathbf{z}_{\text{x1}}$ and $\mathbf{z}_{\text{x2}}$. However, $\mathbf{z}_{\text{x}j}$ is produced from a random weight of the specialized projectors. We require a method to improve the representation of the projectors.

### 3.3 How Do We Improve The Contextual Embedding Space?

One of the key components in this work is improving the semantic understanding of representation $\mathbf{z}_{\text{x}j}$ produced from specialized projectors $\text{Upper}(\cdot)$ or $\text{Lower}(\cdot)$. A common practice is applying supervised contrastive learning to a pair-wise representation. However, we found that in-batch contrastive learning harms the projectors' performance because it is required to compose the representations from difference projectors ($\text{Upper}(\cdot)$ and $\text{Lower}(\cdot)$) in the same mini-batch. (See Section 5.4 for experimental analysis.) Therefore, we design a more suitable learning objective for the classify-and-rank mechanism, which is linear similarity prediction for each projector separately. In particular, we concatenate $\mathbf{z}_{\text{x1}}$ with $\mathbf{z}_{\text{x2}}$ using a linear layer $\text{G}_2$ where the linear's output is regression number from zero (dissimilar) to one (similar). We then minimize the discrepancy between the output and gold label $y_{\text{sim}}$ (STS score) with the BCE loss as follows:

$$\mathcal{L}_{\text{RL}} = \text{BCE}(y_{\text{sim}}, \text{G}_2(\text{concat}(\mathbf{z}_{\text{x1}}, \mathbf{z}_{\text{x2}}))) \quad (4)$$

The final training loss $\mathcal{L}$ is an end-to-end paradigm of representation learning and classification losses:

$$\mathcal{L} = \underbrace{\alpha_1 \mathcal{L}_{\text{RL}}}_{\text{representation learning}} + \underbrace{\alpha_2 \mathcal{L}_{\text{Clf}}}_{\text{classification}} \quad (5)$$

The parameters $\alpha_1$ and $\alpha_2$ are the loss weights obtained from tuning on the development set.

## 4 Experimental Setup

The purpose of our experimental studies is to understand how MixSP performs compared to the traditional fine-tuning method (Reimers and Gurevych,

2019) and Mixture-of-Expert (Zhou et al., 2022b) as competitive methods. Since we present MixSP as a generic STS enhancement method, we assess MixSP against its competitive methods by varying critical factors like the pre-trained sentence encoder, base model, and evaluation tasks and observe how results generalize.

### 4.1 Competitive Methods

To assess the effectiveness of MixSP as an STS enhancement method, we compare it against two competitors. Note that for the full implementation of competitive methods, please refer to Appendix A.1.

- **+FT** (Reimers and Gurevych, 2019) [No Space Decomposition]. We fine-tune the base model through the cosine similarity function. This method serves as our fine-tuning baseline in which the base model is directly adapted to the STS task.
- **+MoE** (Zhou et al., 2022b) [Soft Selection]. As our comparator for embedding space decomposition, we employ the Mixture-of-Expert method. Its relative performance against MixSP will provide insight into the merit of the hard selection approach adopted by MixSP, i.e., supervised classification loss $\mathcal{L}_{\text{Clf}}$ and the Argmax selection as opposed to weighted-average pooling in Section 3.2.2.

To ensure fair and transparent assessment, we apply the same STS dataset, STS-B (Cer et al., 2017), to all methods. We chose STS-B due to its well-documented sources, which can help us avoid data leakage when selecting evaluation datasets. For the full data leakage discussion, please refer to Appendix A.2.

### 4.2 Training Setup

We use STS-B training data following prior works (Cer et al., 2017; Reimers and Gurevych, 2019). For the lower-range and upper-range samples, we separate the lower-range and upper-range samples according to the STS score in the range of [0,4) and [4,5], respectively. We use AamW as the optimizer, a learning rate of $5e^{-5}$, and a batch size of 16 for 10 epochs. We use $\alpha_1$ and $\alpha_2$ equal to $7e^{-4}$ and $1e^{-4}$, respectively (tuned on the STS-B development set). For the base encoder, we use SBERT, SimCSE, and DiffAug to observe the improvement of changing from a single embedding to a separate embedding space. All experiments were done on a single V100 with three random seeds per model.

### 4.3 Sentence Encoders and Base Models

We employ off-the-shelf text encoder models as follows:

- **SBERT** (Reimers and Gurevych, 2019). A supervised baseline. The model was trained on the STS-B training set with cosine similarity as the training objective (similar to our method).
- **SimCSE** (Gao et al., 2021). A simple contrastive learning method using dropout as the data augmentation.
- **DiffAug** (Wang and Lu, 2022). A two-stage contrastive learning framework. Contrastive learning is applied to minimize the discrepancy between two differentiable augmentation schemes.

Note that SimCSE and DiffAug were trained on NLI-supervised datasets, MNLI (Williams et al., 2018) and SNLI (Bowman et al., 2015) datasets. In addition to varying the sentence encoders, we test our method with two different architectures, BERT-Base and RoBERTa-Base.

### 4.4 Evaluation Tasks

We evaluate the effectiveness of our method compared to competitive methods on two tasks: STS and zero-shot tasks. For the STS task, we select the STS benchmarks with low to non-word overlapping between our training data and benchmarks to prevent data leakage, as discussed in Appendix A.2. In particular, we evaluate our model with three STS benchmarks: CDSC-R (validation set), CDSC-R (test set) (Wróblewska and Krasnowska-Kieraś, 2017), and BIOSSES (Soğancıoğlu et al., 2017). In addition, we also evaluate our model on standard seven STS datasets in Appendix A.3. We use Spearman's rank correlation as the main metric to be consistent with prior works.

For the zero-shot task, we assess the generalizability of our model across unseen tasks/domains, namely reranking and binary text classification. In the reranking task, we adopt the settings and datasets from MTEB (Muennighoff et al., 2023), where the Mean Average Precision (MAP) is the main evaluation metric. We also test our model on sentence-pair binary classification tasks where the model has to decide if a sentence-pair has certain relations. The ground truth labels of this task are either 0 or 1. We calculate Area Under Curve (AUC) scores with the binary labels and the relevance scores predicted by models following previous works (Li et al., 2020; Liu et al., 2022; Limkonchotiwat et al., 2023).

| Method | BERT-Base | | | | RoBERTa-Base | | | |
|---|---|---|---|---|---|---|---|---|
| | BIOSSES | CDSC-R (Val) | CDSC-R (Test) | Avg. | BIOSSES | CDSC-R (Val) | CDSC-R (Test) | Avg. |
| *SBERT as the base encoder* | | | | | | | | |
| SBERT | 63.88 | 59.48 | 63.53 | 62.30 | 72.03 | 68.37 | 70.57 | 70.32 |
| +MoE | 78.52 | 84.70 | 84.02 | 82.41 | 73.75 | 85.09 | 79.88 | 79.57 |
| +MixSP | **80.58**$_{\pm 0.64}$ | **85.08**$_{\pm 0.65}$ | **84.15**$_{\pm 0.59}$ | **83.27**$_{\pm 0.58}$ | **76.01**$_{\pm 1.13}$ | **85.60**$_{\pm 0.44}$ | **81.21**$_{\pm 0.30}$ | **80.94**$_{\pm 0.47}$ |
| *SimCSE as the base encoder* | | | | | | | | |
| SimCSE | 68.38 | 70.21 | 70.63 | 69.74 | 67.75 | 68.38 | 70.64 | 68.92 |
| +FT | 76.62 | 69.98 | 69.53 | 72.04 | 73.35 | 69.01 | 71.69 | 71.35 |
| +MoE | 77.07 | 82.87 | 83.34 | 81.09 | 72.65 | 84.01 | 80.33 | 79.00 |
| +MixSP | **82.61**$_{\pm 0.80}$ | **88.27**$_{\pm 0.48}$ | **85.28**$_{\pm 0.06}$ | **85.39**$_{\pm 0.35}$ | **80.74**$_{\pm 0.85}$ | **84.48**$_{\pm 0.31}$ | **80.41**$_{\pm 0.19}$ | **81.88**$_{\pm 0.34}$ |
| *DiffAug as the base encoder* | | | | | | | | |
| DiffAug | 40.12 | 61.42 | 62.61 | 54.72 | 39.15 | 62.47 | 64.65 | 55.42 |
| +FT | 71.26 | 67.91 | 70.25 | 69.81 | 71.02 | 64.14 | 70.66 | 68.61 |
| +MoE | 79.95 | 86.29 | 84.71 | 83.65 | 72.65 | 85.01 | 81.33 | 79.66 |
| +MixSP | **81.23**$_{\pm 0.84}$ | **88.28**$_{\pm 0.42}$ | **85.45**$_{\pm 0.46}$ | **84.99**$_{\pm 1.03}$ | **80.35**$_{\pm 0.65}$ | **86.16**$_{\pm 0.58}$ | **81.79**$_{\pm 0.83}$ | **82.77**$_{\pm 0.69}$ |

Table 1: Spearman's rank correlation on the STS benchmarks in a fair environment for assessment.

# 5 Experimental Results

## 5.1 STS Benchmarks

To assess the generalizability of MixSP, we tested it on two architectures, three base encoders, and three datasets, bringing the total number of combinations to 18. As shown in Table 1, MixSP outperforms competitive methods in all cases. Moreover, as a generic improvement method, we can observe reasonable improvement compared to fine-tuning techniques for all the average scores. For instance, when changing from a single to separate embedding spaces, we improved the performance of SOTA (SimCSE-BERT-Base) from 69.74 to 85.39 (15.65 improvement points). In particular, compared to fine-tuning methods that use the same dataset as our method, FT and MoE, we outperform them by 13.35 points and 4.30 points on the average score, respectively. Furthermore, we evaluate the effectiveness of each method on RoBERTa-Base. The experimental results demonstrate consistency improvement similar to BERT-Base, e.g., MixSP outperforms FT and MoE on DiffAug by 14.16 and 3.11 points on the average score, respectively. Note that we experimented on the accuracy of our router network in Appendix 5.5.

## 5.2 Zero-shot Downstream Tasks

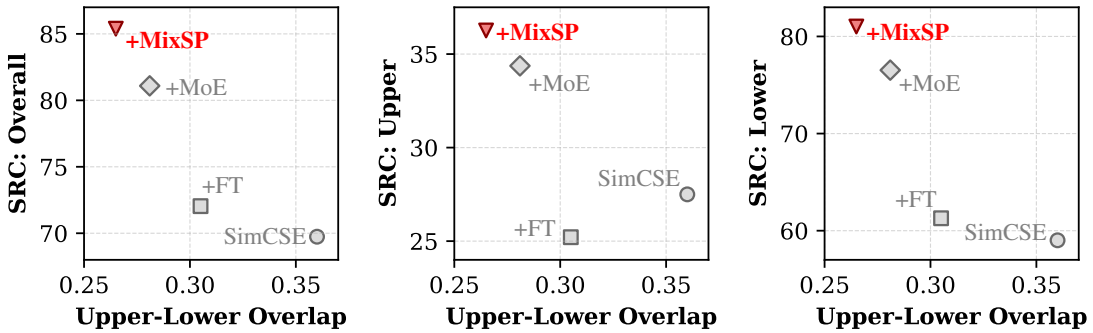
In our prior experiment, we demonstrated the effectiveness of our method in the seen task (STS). However, a crucial question emerges: *does sentence embedding performance in the STS task accurately represent a model's capabilities?* To explore this, we assess our method in zero-shot settings, evaluating its performance in two unseen tasks/domains: *reranking* and *binary text classification*. This study aims to unveil the versatility of our model, offering insights into its potential application across a diverse range of tasks and domains.

### 5.2.1 Reranking

In this experiment, we study the effectiveness of our model on unseen datasets and tasks from the MTEB reranking benchmark. As demonstrated in Table 2, MixSP outperforms competitive methods in all cases. Our method improves the performance of SOTA (SimCSE) from 47.54 to 51.01 points. Moreover, we achieve a new SOTA on DiffAug by improving the performance from 47.38 to 52.94 points. In contrast, we found performance decreases in the traditional fine-tuning technique, FT. For example, when we applied FT to DiffAug, the performance of DiffAug decreased from 47.38 to 46.66. *This finding demonstrates that MixSP has benefits beyond the seen task of STS.*

| Method | AU | MM | SD | SO | Avg. |
|---|---|---|---|---|---|
| *SBERT-BERT-Base* | | | | | |
| SBERT | 51.09 | 30.24 | 69.40 | 36.54 | 46.82 |
| +MoE | 53.72 | 30.00 | 75.14 | 41.72 | 50.15 |
| +MixSP | **54.56** | **30.59** | **75.78** | **42.39** | **50.83** |
| *SimCSE-BERT-Base* | | | | | |
| SimCSE | 51.80 | 29.30 | 70.14 | 38.90 | 47.54 |
| +FT | 51.81 | 28.91 | 70.07 | 38.19 | 47.25 |
| +MoE | 53.94 | 27.98 | 74.25 | 41.51 | 49.42 |
| +MixSP | **54.50** | **30.68** | **75.35** | **43.51** | **51.01** |
| *DiffAug-BERT-Base* | | | | | |
| DiffAug | 51.10 | 29.52 | 71.10 | 37.81 | 47.38 |
| +FT | 51.04 | 29.44 | 69.82 | 36.34 | 46.66 |
| +MoE | 52.68 | 29.83 | 74.76 | 40.52 | 49.45 |
| +MixSP | **53.92** | **30.07** | **75.68** | **42.08** | **52.94** |

Table 2: The MAP score on the reranking task from the MTEB benchmark where AU = AskUbuntuDupQuestions, MM = MindSmallReranking, SD = SciDocsRR, and SO = StackOverflowDupQuestions.




(a) Overall: STS $\in [0, 5]$ (b) Upper-Range: STS $\in [4, 5]$ (c) Lower-Range: STS $\in [0, 4)$

Figure 3: Comparison between MixSP and competitors on three STS datasets. The x-axis represents class overlap, quantifying the confusion between upper-range and lower-range classes. The y-axis measures similarity ranking performance using Spearman's rank correlation (SRC) coefficients. The figures include SRC Coefficients for Overall Results (3a), Upper-Range (3b), and Lower-Range (3c). These visuals illustrate the reduced class overlap and effective sentence pair ranking compared to competitors.

#### 5.2.2 Binary Text Classification

In this experiment, we study the generalization of sentence embedding methods on standard binary text classification datasets. As shown in Table 3, our method outperforms competitive methods in the average score. In particular, we improved the average AUC score of DiffAug-FT and DiffAug-MoE by 4.18 and 0.58 points, respectively. Moreover, a consistent pattern emerges, similar to observations in the STS and reranking benchmarks. Our method consistently outperforms competitive methods on SBERT and SimCSE; e.g., the gap between our work and MoE on SBERT is 0.68 points.

| Method | QQP | QNLI | MRPC | Avg. |
|---|---|---|---|---|
| *SBERT-BERT-Base* | | | | |
| SBERT | 80.44 | 68.51 | 80.13 | 76.36 |
| +MoE | 81.60 | 76.91 | 83.08 | 80.53 |
| +MixSP | **82.29** | **78.21** | **83.21** | **81.24** |
| *SimCSE-BERT-Base* | | | | |
| SimCSE | 81.37 | 74.53 | 77.83 | 77.91 |
| +FT | **82.55** | 72.40 | 81.48 | 78.81 |
| +MoE | 81.66 | 79.36 | **82.54** | 81.19 |
| +MixSP | 82.08 | **80.14** | 82.32 | **81.51** |
| *DiffAug-BERT-Base* | | | | |
| DiffAug | 81.90 | 74.17 | 78.08 | 78.05 |
| +FT | 81.39 | 69.11 | 81.30 | 77.27 |
| +MoE | 81.33 | 78.64 | **82.63** | 80.87 |
| +MixSP | **81.85** | **80.11** | 82.40 | **81.45** |

Table 3: The AUC score of binary text classification on three standard datasets.

### 5.3 Why Does MixSP Work?

In this experiment, we explain why our method outperforms previous works based on two metrics: (i) the overlap area in sentence embedding and (ii) the upper-range and lower-range alignment scores.

**Reduced Upper-Lower Ranges Overlap**. As discussed in the introduction (Figure 1), MixSP produces the smallest upper-lower ranges overlap compared to FT and MoE. In this subsection, let us examine how this quantity relates to the overall, upper-range, and lower-range raking performances shown in Figure 3. Results are reported as the average over the three STS datasets used in the main experiment. For the full results, please refer to Table 9 in the appendix section. As can be seen, MixSP exhibits the smallest overlap compared to competitive methods. Figure 3a shows that we decrease the overlap from 0.360 (SimCSE) and 0.281 (MoE) to 0.265. Regarding the overall ranking performance, MixSP provides the highest SRC.

**Ranking improvement in upper-range and lower-range classes**. Figures 3b and 3c provide an insight into the ranking performance within each class. We can see that MixSP is also the best performer regarding the upper-range and lower-range ranking. Interestingly, Figures 3b and 3c show contrasting results for FT regarding upper-range and lower-range classes ranking performances. Compared to the SimCSE baseline, FT provides a performance drop for the upper-range class and a performance increase for the lower-range class. One possible explanation comes from our hypothesis that the upper-range and lower-range samples differ linguistically. Since the lower-range class dominates the datasets, familiarizing the model with an STS dataset helps improve the lower-range ranking *but* detriments the upper-range ranking performance. Another valuable insight obtained from this analysis is the performance gap between upper-range and lower-range classes. All methods exhibit an upper-lower range performance difference of at

least 31.51 points. This insight suggests that more research attention should be dedicated to improving upper-range raking performance.

**Summary**. The space decomposition mechanism in MixSP produces the smallest upper-lower ranges overlap and obtains the best raking performance overall, as well as the individual cases of upper-range and lower-range classes. These results highlight the benefits of dividing the samples into upper-range and lower-range classes with the assistance of the routing network and specialized projectors. Our analysis also shows the ranking performance gap between the upper-range and lower-range classes. This insight suggests where the research attention should be dedicated to improving the STS ranking performance.

### 5.4 Ablation Study

In this study, we analyze our framework's performance and design choice, including the routing network, specialized projectors, and training objectives. In addition, we use SimCSE+MixSP as our baseline. The analyses of each component are presented as follows.

| Method | BERT-Base |
|---|---|
| MixSP | 85.39 |
| *Routing network* | |
| Using only $\mathbf{h}_{xj}$ for the routing network | ↓1.39 |
| Using $\mathbf{h}_{x1}$ and $\mathbf{h}_{x2}$ for the routing network | ↓1.30 |
| Removing $\mathcal{L}_{\text{clf}}$ | ↓1.10 |
| *Specialized projectors* | |
| [0,4), [4,5]→[0,1), [1,2), [2,3), [3,4), [4,5] | ↓0.73 |
| [0,4), [4,5]→[0,3), [3,5] | ↓0.92 |
| [0,4), [4,5]→[0,2), [2,5] | ↓1.09 |
| Argmax→Weighted-average pooling | ↓1.31 |
| *Training objectives* | |
| BCE→Contrastive learning | ↓3.43 |
| BCE→Cosine similarity | ↓1.79 |

Table 4: The design choice of our framework. We evaluate the average STS score across three STS datasets. → is replacing the left method with the right method.

**Routing network**. As presented in Table 4, the best setting of the routing network is the default setting (our decomposition embedding space setting). For example, we found performance decreases by 1.10 points when removing the classification loss ($\mathcal{L}_{\text{Clf}}$). This outcome underscores the importance of having a supervised signal for the router in contrast to the MoE, which lets the attention module automatically decide how to route. Altering the default setting, which is designed based on the desired property and linguistic observation, harms the model's performance in all cases.

**Specialized projectors**. Table 4 shows a consistent decline in the STS score when we change from the default range, [0,4) and [4,5], to other ranges. Specifically, when changing from the default range to five ranges, there is a noticeable drop in performance by 0.73 points. In addition, when projector representations are combined through weighted-average pooling similar to MoE, the results decrease by 1.31 points. This trend highlights the effective representation achieved using two projectors separately to sufficiently capture upper-range and lower-range samples.

**Training objective**. One of the key successes of MixSP is the training objective. In this study, we changed from binary cross-entropy to well-established supervised training objectives, such as contrastive learning and cosine similarity. As shown in Table 4, the experimental results demonstrate that using the default training objective yields the best STS score. While contrastive learning demonstrated a reasonable performance in competitive methods, it adversely impacted the model's performance more than any other setting. This is because contrastive learning combines negative representations produced from different specialized projectors through a mini-batch negative sample. Mixing both representations from upper-range and lower-range projectors in the training step only creates confusion between the two classes, deteriorating the model's performance. The experimental result from Argmax→Weighted-average pooling conforms with this analysis.

### 5.5 Vanilla Vs. End-to-End Routing Networks

A common technique to separate two classes (i.e., upper-range and lower-range classes) is training a sequence text classification with a PLM (Devlin et al., 2019), while our work trains the routing network simultaneously with the representation learning (the end-to-end manner). In this study, we evaluate the effectiveness of our routing network compared to the vanilla text classification model on three STS benchmarks. Table 5 demonstrates that training a text classification in an end-to-end manner outperforms the vanilla model in all metrics. This is because our routing network's training objective $\mathcal{L}_{\text{Clf}}$ receives the benefit from the $\mathcal{L}_{\text{RL}}$ loss by dynamically adjusting gradients alongside representation learning, thereby enhancing classification performance. This result confirms that training the group classification in an end-to-end manner surpasses the performance of the two-stage paradigm.

| Model | BIOSSES | CDSC-R(Val) | CDSC-R(Test) | Avg. |
|---|---|---|---|---|
| End–to–end | **95.50** | **88.80** | **86.95** | **90.42** |
| Vanilla | 94.00 | 82.60 | 82.40 | 86.33 |

Table 5: The accuracy score of our routing network (end-to-end) compared to a vanilla text classification model on three STS benchmarks.

### 5.6 Tuning Time and Memory Requirements

Table 6 compares the original SimCSE and three competitive tuning methods in terms of the number of parameters, tuning time, GPU memory consumption (during tuning), and the SRC score. Since FT *does not* introduce any new component, it has the shortest tuning time and consumes the smallest GPU memory. MoE and MixSP incur substantially higher costs than FT, which are comparable to each other, while MixSP provides the highest performance uplift among the three tuning methods.

| Method | #Params | Tuning time | GPU memory | SRC Avg. |
|---|---|---|---|---|
| SimCSE | **110M** | - | - | 69.73 |
| +FT | **110M** | **83 sec.** | **5.606 MBs** | 72.04 |
| +MoE | 133M | 182 sec. | 8,352 MBs | 81.09 |
| +MixSP | 145M | 198 sec. | 8,390 MBs | **85.39** |

Table 6: The number of parameters, training time, GPU memory usage, and the average Spearman's rank correlation (SRC) score from Table 1. We use the same training data, learning rate, epoch, and batch size for the competitive methods and ours.

## 6 Conclusion

In this paper, we proposed a novel embedding space decomposition method called MixSP, a mixture of specialized projectors. We challenge the common practice in treating STS scores from a continuous paradigm [0,5] to embedding space decomposition for lower-range [0,4) and upper-range [4,5] classes. Our experiments highlight that MixSP outperforms competitive methods in the average cases of STS and zero-shot benchmarks. We also discuss the improvement of our method, design choice, and inference speed to emphasize the effectiveness of our framework. The embedding space decomposition is the promising paradigm for sentence embedding.

## 7 Limitation

Since our method incorporates a learnable module for enhancing sentence embeddings, MixSP's parameter increases from 110 million to 145 million parameters, as shown in Table 6. In addition, the training time and memory usage of our method are also higher than competitor methods. However, regarding performance, our method outperforms competitors in all cases.

## References


Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Iñigo Lopez-Gazpio, Montse Maritxalar, Rada Mihalcea, German Rigau, Larraitz Uria, and Janyce Wiebe. 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, Denver, Colorado. Association for Computational Linguistics.

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)*, pages 81–91, Dublin, Ireland. Association for Computational Linguistics.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. 2016. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 497–511, San Diego, California. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 2012. SemEval-2012 task 6: A pilot on semantic textual similarity. In **SEM 2012: 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 (SemEval 2012)*, pages 385–393, Montréal, Canada. Association for Computational Linguistics.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *SEM 2013 shared task: Semantic textual similarity. In *Second Joint Conference on Lexical and Computational Semantics (*SEM), Volume 1: Proceedings of the Main Conference and the Shared Task: Semantic Textual Similarity*, pages 32–43, Atlanta, Georgia, USA. Association for Computational Linguistics.

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.

Daniel Cer, Mona Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. 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. Association for Computational Linguistics.

Zitian Chen, Yikang Shen, Mingyu Ding, Zhenfang Chen, Hengshuang Zhao, Erik G. Learned-Miller, and Chuang Gan. 2023. Mod-squad: Designing mixtures of experts as modular multi-task learners. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2023, Vancouver, BC, Canada, June 17-24, 2023*, pages 11828–11837. IEEE.

Mohammed Nowaz Rabbani Chowdhury, Shuai Zhang, Meng Wang, Sijia Liu, and Pin-Yu Chen. 2023. Patch-level routing in mixture-of-experts is provably sample-efficient for convolutional neural networks. In *International Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA*, volume 202 of *Proceedings of Machine Learning Research*, pages 6074–6114. PMLR.

Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljacic, Shang-Wen Li, Wen-tau Yih, Yoon Kim, and James Glass. 2022. DiffCSE: Difference-based contrastive learning for sentence embeddings. In *Annual Conference of the North American Chapter of the Association for Computational Linguistics (NAACL)*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *NAACL-HLT 2019*.

Hongchao Fang, Sicheng Wang, Meng Zhou, Jiayuan Ding, and Pengtao Xie. 2020. Cert: Contrastive self-supervised learning for language understanding.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *EMNLP 2021*.

Ting Jiang, Jian Jiao, Shaohan Huang, Zihan Zhang, Deqing Wang, Fuzhen Zhuang, Furu Wei, Haizhen Huang, Denvy Deng, and Qi Zhang. 2022a. Promptbert: Improving BERT sentence embeddings with prompts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022, Abu Dhabi, United Arab Emirates, December 7-11, 2022*, pages 8826–8837. Association for Computational Linguistics.

Yuxin Jiang, Linhan Zhang, and Wei Wang. 2022b. Improved universal sentence embeddings with prompt-based contrastive learning and energy-based learning. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3021–3035, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Bo Li, Yifei Shen, Jingkang Yang, Yezhen Wang, Jiawei Ren, Tong Che, Jun Zhang, and Ziwei Liu. 2023. Sparse mixture-of-experts are domain generalizable learners. In *The Eleventh International Conference on Learning Representations, ICLR 2023, Kigali, Rwanda, May 1-5, 2023*. OpenReview.net.

Bohan Li, Hao Zhou, Junxian He, Mingxuan Wang, Yiming Yang, and Lei Li. 2020. On the sentence embeddings from pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 9119–9130. Association for Computational Linguistics.

Peerat Limkonchotiwat, Wuttikorn Ponwitayarat, Lalita Lowphansirikul, Can Udomcharoenchaikit, Ekapol Chuangsuwanich, and Sarana Nutanong. 2022. ConGen: Unsupervised control and generalization distillation for sentence representation. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 6467–6480, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Peerat Limkonchotiwat, Wuttikorn Ponwitayarat, Lalita Lowphansirikul, Can Udomcharoenchaikit, Ekapol Chuangsuwanich, and Sarana Nutanong. 2023. An efficient self-supervised cross-view training for sentence embedding. *Transactions of the Association for Computational Linguistics*.

Fangyu Liu, Yunlong Jiao, Jordan Massiah, Emine Yilmaz, and Serhii Havrylov. 2022. Trans-encoder: Unsupervised sentence-pair modelling through self- and mutual-distillations. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*.

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 the Ninth International Conference on Language Resources and Evaluation (LREC'14)*, pages 216–223, Reykjavik, Iceland. European Language Resources Association (ELRA).

Zhongtao Miao, Qiyu Wu, Kaiyan Zhao, Zilong Wu, and Yoshimasa Tsuruoka. 2024. Enhancing cross-lingual sentence embedding for low-resource languages with word alignment.

Niklas Muennighoff, Nouamane Tazi, Loic Magne, and Nils Reimers. 2023. MTEB: Massive text embedding benchmark. In *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*, pages 2014–2037, Dubrovnik, Croatia. Association for Computational Linguistics.

Juri Opitz and Anette Frank. 2022. SBERT studies meaning representations: Decomposing sentence embeddings into explainable semantic features. In *Proceedings of the 2nd Conference of the Asia-Pacific*

*Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2022 - Volume 1: Long Papers, Online Only, November 20-23, 2022*, pages 625–638. Association for Computational Linguistics.

Nils Reimers and Iryna Gurevych. 2019. 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.

Gizem Soğancıoğlu, Hakime Öztürk, and Arzucan Özgür. 2017. BIOSSES: a semantic sentence similarity estimation system for the biomedical domain. *Bioinformatics*, 33(14):i49–i58.

Bin Wang, Fenxiao Chen, Yuncheng Wang, and C.-C. Jay Kuo. 2020. Efficient sentence embedding via semantic subspace analysis. In *25th International Conference on Pattern Recognition, ICPR 2020, Virtual Event / Milan, Italy, January 10-15, 2021*, pages 119–125. IEEE.

Kexin Wang, Nils Reimers, and Iryna Gurevych. 2021. TSDAE: Using transformer-based sequential denoising auto-encoderfor unsupervised sentence embedding learning. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 671–688, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Tianduo Wang and Wei Lu. 2022. Differentiable data augmentation for contrastive sentence representation learning. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Yufei Wang, Can Xu, Qingfeng Sun, Huang Hu, Chongyang Tao, Xiubo Geng, and Daxin Jiang. 2022. PromDA: Prompt-based data augmentation for low-resource NLU tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4242–4255, Dublin, Ireland. Association for Computational Linguistics.

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.

Alina Wróblewska and Katarzyna Krasnowska-Kieraś. 2017. Polish evaluation dataset for compositional distributional semantics models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 784–792, Vancouver, Canada. Association for Computational Linguistics.

Yuanmeng Yan, Rumei Li, Sirui Wang, Fuzheng Zhang, Wei Wu, and Weiran Xu. 2021. ConSERT: A contrastive framework for self-supervised sentence representation transfer. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 5065–5075, Online. Association for Computational Linguistics.

Ziyi Yang, Yinfei Yang, Daniel Cer, Jax Law, and Eric Darve. 2021. Universal sentence representation learning with conditional masked language model. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6216–6228, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Jing Zhou, Yanan Zheng, Jie Tang, Li Jian, and Zhilin Yang. 2022a. FlipDA: Effective and robust data augmentation for few-shot learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 8646–8665, Dublin, Ireland. Association for Computational Linguistics.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, zhifeng Chen, Quoc V Le, and James Laudon. 2022b. Mixture-of-experts with expert choice routing. In *Advances in Neural Information Processing Systems*, volume 35, pages 7103–7114. Curran Associates, Inc.

# A Appendix

## A.1 Competitive Method Implementations

As stated in Section 4.1, we compare our method with two competitive methods, such as +FT and +MoE. We explain in more detail about the implementation and differences between our work and competitive methods as follows:

- **+FT** [No Space Decomposition]: First, we modify the model architecture to conform with the learning process described in Sentence Transformers (Reimers and Gurevych, 2019). Then, for fine-tuning, we utilize the cosine similarity loss function to train the model, comparing predicted and ground truth similarity scores. The final result is a model that takes pairs of sentences and outputs similarity scores for STS. FT treats the entire STS score range [0,5], providing a contrast to our method, which separates the ranges [0,4) and [4,5].
- **+MoE** [Soft Selection]: First, we utilize the standard cross-encoder architecture (Reimers and Gurevych, 2019). Second, we modify the model to incorporate the MoE technique (Zhou et al., 2022b) and train the model using the BCE

| Dataset | 4-gram similarity | 5-gram similarity | 6-gram similarity |
|---|---|---|---|
| STS12 | 0.1168 | 0.1059 | 0.0968 |
| STS13 | 0.0147 | 0.0086 | 0.0048 |
| STS14 | 0.1492 | 0.1416 | 0.1876 |
| STS15 | 0.0675 | 0.0611 | 0.0546 |
| STS16 | 0.0066 | 0.0040 | 0.0024 |
| SICK–R | 0.0242 | 0.0160 | 0.0100 |
| STS–B | 0.0258 | 0.0188 | 0.0148 |
| BIOSSES | 0.0001 | – | – |
| CDSC–R (validation set) | – | – | – |
| CDSC–R (test set) | – | – | – |

Table 7: The comparison of N-gram similarity between the STS-B training set and other STS corpora.

loss function to compare predicted and ground truth similarity scores. Similar to FT, we obtain a model that accepts a sentence pair and outputs a similarity score as the final result. While MoE decomposes the embedding space using multiple experts, they combine outputs using weighted average or soft-selection. In contrast, MixSP performs a hard selection on the output, utilizing only one output representation at a time.

### A.2 Data Leakage

In this study, we demonstrate the data leakage in STS-B and standard STS benchmarks. We found that STS-B has a high n-gram overlap with the test data (STS 12-16, SICK-R, and STS-B test set). This is because STS-B comprises samples from STS datasets between 2012 to 2016, as stated on the STS-B's website [2]. In addition, we also conducted an n-gram overlap analysis and found a high Jaccard similarity compared to other datasets, as shown in Table 7. Therefore, we need to omit the high word-overlap dataset from our main experimental results. Our experimental results only consisted of the CDSR-R and BIOSSES datasets.

### A.3 Seven Standard STS Datasets

In this study, we evaluate the effectiveness of our method and competitive methods on the traditional seven benchmarks. Note that all models were trained on STS-B training data. However, we notice that using the STS-B as the training data might cause data leakage, as discussed in Appendix A.2. Thus, we did not include these results in the main paper.

As shown in Table 8, our method outperforms competitive methods on the average score. We observe performance improvements compared to base encodes in all cases for all methods. Moreover, we also notice the performance gap in SBERT on the main table (Table 1) and seven standard STS datasets. This is because the data leakage setting and SBERT seem to find a shortcut in this setting.

[2] https://ixa2.si.ehu.eus/stswiki/stswiki/index.php/Special:Random.html

| Method | STS12 | STS13 | STS14 | STS15 | STS16 | STS-B | SICK-R | Avg. |
|---|---|---|---|---|---|---|---|---|
| *SBERT as the base encoder* | | | | | | | | |
| SBERT | 72.51 | 87.16 | 84.32 | 85.26 | 79.65 | 82.67 | 73.37 | 80.71 |
| +MoE | **80.71** | 87.89 | 89.88 | 90.83 | 83.75 | 86.14 | 75.01 | 84.89 |
| +MixSP | 79.99 | **89.47** | **89.99** | **90.89** | **84.59** | **87.27** | **76.23** | **85.49** |
| *SimCSE as the base encoder* | | | | | | | | |
| SimCSE | 75.30 | 84.67 | 80.19 | 85.40 | 80.82 | 84.25 | 80.39 | 81.57 |
| +FT | 79.15 | 89.44 | 89.33 | 90.46 | 83.50 | 87.04 | **80.40** | 85.39 |
| +MoE | 79.38 | 89.02 | 89.80 | **91.35** | 83.38 | 86.34 | 76.43 | 85.10 |
| +MixSP | **81.08** | **89.74** | **90.41** | 91.27 | **84.16** | **87.18** | 75.86 | **85.67** |
| *DiffAug as the base encoder* | | | | | | | | |
| DiffAug | 76.92 | 85.17 | 80.81 | 86.91 | 82.52 | 84.32 | **80.27** | 82.42 |
| +FT | 77.30 | 89.56 | 87.77 | 88.54 | 82.02 | 85.36 | 78.94 | 84.21 |
| +MoE | 80.18 | 88.74 | **90.25** | **91.48** | 83.63 | 86.29 | 76.19 | 85.25 |
| +MixSP | **80.88** | **88.89** | 89.64 | 91.46 | **84.53** | **87.00** | 75.88 | **85.47** |

Table 8: Spearman's rank correlation on the seven STS benchmarks (Cer et al., 2017; Marelli et al., 2014; Agirre et al., 2012, 2013, 2014, 2015, 2016). All models were implemented on BERT-Base.

| Model | BIOSSES | | | CDSC-R(Val) | | | CDSC-R(Test) | | | Avg. | | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | Upper | Lower | Overlap | Upper | Lower | Overlap | Upper | Lower | Overlap | Upper | Lower | Overlap |
| SimCSE | 66.34 | 27.51 | 0.414 | 56.92 | 30.47 | 0.292 | 53.78 | 24.53 | 0.374 | 59.01 | 27.50 | 0.360 |
| +FT | 74.12 | 25.22 | 0.234 | 55.79 | 29.89 | 0.295 | 53.75 | 20.52 | 0.385 | 61.28 | 25.21 | 0.305 |
| +MoE | 74.21 | 34.36 | 0.238 | 75.97 | 45.08 | 0.281 | 79.42 | 23.65 | 0.324 | 76.53 | 34.37 | 0.281 |
| +MixSP | **80.06** | **36.27** | **0.201** | **79.57** | 44.69 | **0.280** | **83.44** | **27.83** | **0.314** | **81.02** | **36.26** | **0.265** |

Table 9: Spearman's rank correlation on the STS benchmarks where Upper is upper-range samples: STS $\in [4, 5]$, Lower is lower-range samples: STS $\in [0, 4)$, and Overlap is the overlap area between $[0, 4)$ and $[4, 5]$.