Prompt Tuning Can Simply Adapt Large Language Models to Text Encoders

Kaiyan Zhao, Qiyu Wu, Zhongtao Miao, Yoshimasa Tsuruoka
The University of Tokyo, Tokyo, Japan
{zhaokaiyan1006, qiyuw, mzt, yoshimasa-tsuruoka}@g.ecc.u-tokyo.ac.jp

Abstract

Recently, many works have been attempting to adapt Large Language Models (LLMs) for sentence embedding, with most of them fine-tuning LLMs towards the contrastive objective and enabling bi-directional attention for better performance, using LoRA to address the large model scale. In this work, we suggest that this adaptation can also be simply and effectively achieved using causal attention and with even fewer trainable parameters through soft prompt tuning, as an alternative to fine-tuning with LoRA and other methods with extra post-training tasks. Our method only optimizes a few learnable tokens while keeping the rest of the model frozen. Through experiments on a diverse set of evaluation tasks, we find that simply tuning only a few tokens can achieve a competitive performance with that of fine-tuning with LoRA. The percentage of trainable parameters can be reduced to less than 0.001%. Moreover, we also demonstrate that turning causal attention to bi-directional attention with or without extra post-training tasks does not provide additional benefit when soft prompt tuning is applied, suggesting that causal attention can be naturally used in decoder-only LLMs for sentence embedding adaptation.

1 Introduction

Sentence embedding compresses the semantic meaning of sentences into fixed-size vectors in a shared space (Conneau et al., 2017; Wu et al., 2018; Reimers and Gurevych, 2019). Conventional sentence embedding models are typically built on an encoder-only architecture trained with Contrastive Learning (CL) (van den Oord et al., 2018), where the distance between semantically similar sentences are pulled closer and dissimilar ones are pushed farther (Gao et al., 2021; Wu et al., 2022; Chuang et al., 2022; Jiang et al., 2022a). On the other hand, scaled-up Large Language Models (LLMs) in the decoder-only architecture have dominated various downstream tasks with very large-scale parameters and training data (OpenAI, 2022; Touvron et al., 2023a,b; OpenAI, 2023). However, the use of LLMs on sentence embedding still remains challenging, given the fact that decoder-only LLMs are pre-trained to generate continuous texts instead of semantically meaningful vectors (Jiang et al., 2023).

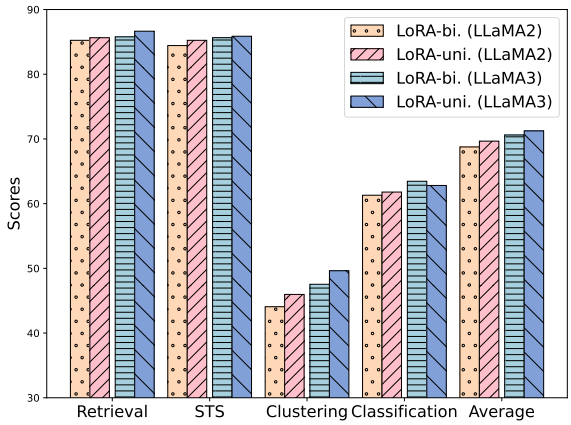

Figure 1: Comparison of LoRA fine-tuning using bi-directional attention or uni-directional attention. Extra post-training task is solely applied to LoRA-bi. Simply fine-tuning with LoRA-uni. shows strong performances. Refer to Table 1 for detailed results.

To this end, numerous recent methods attempt to adapt LLMs for sentence embedding, e.g., CL-based fine-tuning (Jiang et al., 2023; Li and Li, 2023), attention mechanism manipulation (Li and Li, 2024), instruction tuning (Muennighoff et al., 2024), with some approaches employing the combinations thereof. Among these efforts, LLM2Vec (BehnamGhader et al., 2024) stands out as a promising method, employing a three-step approach: (1) enabling bi-directional attention, (2) using Masked Next Token Prediction (MNTP) (Lv et al., 2023) to effectively adapt LLMs to bi-directional attention, and (3) fine-tuning with CL, as shown in the upper part of Figure 2.

Proceedings of the 10th Workshop on Representation Learning for NLP (RepL4NLP-2025), pages 38–50

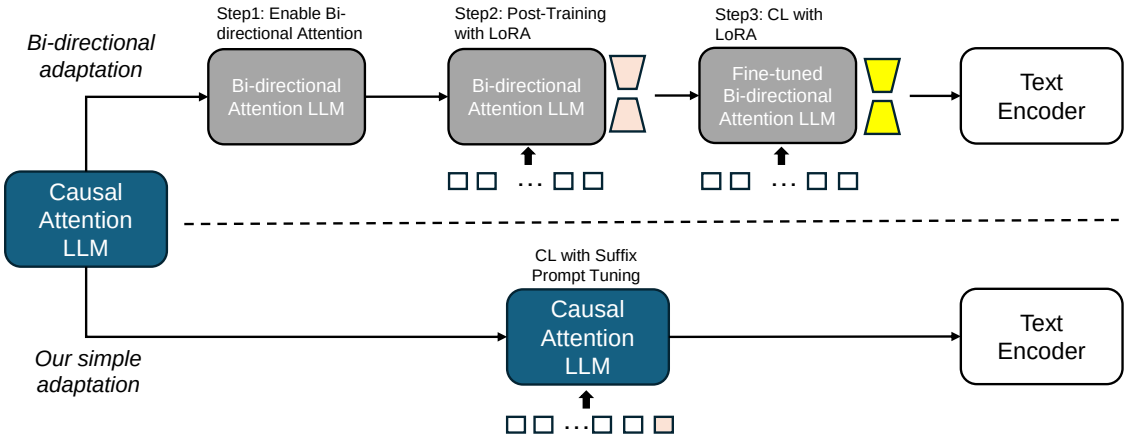

Figure 2: Upper: Conventional three-step methods of turning LLMs into text encoders. We refer to tasks performed before fine-tuning under the CL objective as post-training tasks. Lower: Our simple method which naturally maintains causal attention by appending trainable soft prompts into the input.

However, given the large-scale parameters of LLMs, performing these three steps, especially the latter two, can be computationally inefficient. To address this, Low-Rank Adaptation (LoRA) (Hu et al., 2022) is commonly employed in the aforementioned works to enable more efficient fine-tuning by reducing the number of trainable parameters while maintaining performance.

Given the additional post-training efforts required by LLM2Vec, we begin questioning whether it is possible to naturally maintain causal attention in LLMs for sentence embedding. To explore this, we compare LoRA under bi-directional attention with post-training to directly LoRA under uni-directional attention without post-training on the same dataset, and the results are shown in Figure 1. Interestingly, our results reveal that simply fine-tuning LLMs with LoRA consistently yields strong performances across the four evaluated tasks. Based on these findings, we seek answers to the following two questions: (1) Is bi-directional attention with additional post-training necessary for the adaptation? (2) Is there a simpler adaptation method with minimum modification of the original LLM?

We first investigate the adaptation of LLMs for sentence embedding with even fewer trainable parameters by employing soft prompt tuning (Lester et al., 2021; Li and Liang, 2021; Liu et al., 2022). We introduce SPT (**S**uffix **P**rompt **T**uning based Adaptation of LLMs for Sentence Embedding), a straightforward yet effective alternative to adapt LLMs for sentence embedding. The use of soft prompt tuning in this scenario is non-trivial. Specifically, we append trainable tokens to the inputs, allowing them to attend to all the input tokens due to the causal attention in the decoder-only LLMs, as illustrated in the lower part of Figure 2. Notably, as our approach only optimizes the parameters within the additional soft prompt tokens, it is flexible enough to reduce the amount of trainable parameters to just a few tokens[1]. The percentage of trainable parameters with our approach is less than 0.001%, which is a percentage unreachable by LoRA, even when setting the rank r to 1. Experimental results on retrieval, Semantic Textual Similarities (STS), clustering, and classification tasks reveal that training with only a few tokens can yield comparable or even superior performance to LoRA-based fine-tuning.

Additionally, we thoroughly analyze the impact of bi-directional attention and extra post-training tasks, finding that regardless of the pooling method or attention mechanism used, causal attention without post-training consistently delivers better performance when SPT is applied.

In summary, the contribution of this work includes:

- We propose a simple method that adapts LLMs to text encoders without requiring extra adjustment for bi-directional attention, which is applied in previous methods.
- We investigate the utilization of suffix prompt

[1]The amount of one trainable token varies according to different models, e.g., 768 for OPT and 4096 for LLaMA.

tuning other than LoRA fine-tuning for the adaptation, providing the flexibility to further reduce the trainable parameters.

2 Related Works

2.1 Turning LLMs into Text Encoders

Current methods for adapting LLMs into text encoders can be mainly categorized into two types based on the attention mechanism they use.

w/ Causal Attention. Since most LLMs are pre-trained with casual attention, it is natural to keep this mechanism for sentence representation, using the output of the last input token as sentence embedding. Jiang et al. (2023) make the first attempt to adapt LLMs for sentence embedding. They propose PromptEOL, which utilizes the prompt *This sentence: "[text]" means in one word:"* to generate sentence embedding. Li and Li (2023) later extend this prompt-based method on LLaMA2 using angle optimization to address the gradient vanishing problem in CL. In our work, we also prioritize the natural use of causal attention, while aiming for a simple but effective approach.

w/ Bidirectional Attention On the other hand, some methods transform the causal attention into bi-directional attention for better representation ability. Li and Li (2024) observe that an LLM's sentence representation ability with causal attention initially improves across layers but begins to degrade after reaching a critical turning point (a particular layer). By modifying the layers after the turning point to use bi-directional attention, the LLM improves its sentence encoding ability. BehnamGhader et al. (2024) introduce a three-step pipeline for converting LLMs into text encoders, including enabling bi-directional attention, masked next token prediction (MNTP) and CL-based fine-tuning. MNTP, which requires the model to predict the masked token based solely on the tokens before it, is applied to help LLMs adapt to bi-directional attention. GRITLM (Muennighoff et al., 2024), which utilizes instruction tuning, applies bidirectional attention for embedding tasks and causal attention for generation tasks. However, these methods often require more complex design, potential post-training tasks and rely on much bigger datasets, which is far from simple. To this end, we propose a more efficient and effective method to easily adapt LLMs for high-quality sentence embedding based on causal attention.

2.2 Soft Prompt Tuning in LLMs

Prompts normally refer to the physical tokens additionally provided to the model towards specific tasks (Brown et al., 2020; Zhou et al., 2022; Ouyang et al., 2024). Soft prompt tuning (Lester et al., 2021; Li and Liang, 2021; Liu et al., 2022), which provides virtual tokens (continuous vectors) prepended to the input texts, offers an efficient alternative for fine-tuning LMs. Soft prompt tuning can mitigate overfitting by freezing the model's parameters and updating only the parameters within the soft prompts. Recent works continue to seek for more efficient prompt tuning methods with even fewer parameters (Shi and Lipani, 2024). In the field of sentence embedding, Jiang et al. (2022b) incorporate soft prompts into each layer of the transformer encoder. In contrast, we focus on decoder-only causal attention LLMs and append soft prompts exclusively into the input embedding layer for better efficiency.

3 Methods

CL has become the common practice for learning sentence embeddings with pre-trained LMs (Gao et al., 2021; Wu et al., 2022; Zhao et al., 2024; Miao et al., 2024). It is performed with one anchor sentence, one positive instance and multiple negative instances. Given a sentence X_i, it can be tokenized into $x_1, x_2, ..., x_{|X_i|}$, where $|\cdot|$ denotes the number of tokens in X_i. Our method, SPT, is simple and straightforward. It additionally appends a soft prompt, namely, a few trainable tokens $p = \{p_1, p_2, ..., p_k\}$, to the sentence X_i. This constructs the input as $x_1, x_2, ..., x_{|X_i|}, p_1, p_2, ..., p_k$. Here, k is the length of the soft prompt and the trainable parameters in the soft prompt equal to `[k, hidden_size]`.

Similar to existing methods, the text encoder then transforms X_i into a fixed size dense vector $\mathbf{h}_i$. We use the output of the appended soft prompt for sentence embedding when $k = 1$, and the output of the last soft prompt token p_k as the sentence embedding when $k > 1$.

Our training objective is consistent with previous works. The main idea of CL is to pull the distance between the representation of anchor sentence $\mathbf{h}_i$ and its positive example's representation $\mathbf{h}_i^+$ close while keeping $\mathbf{h}_i$ and other negative examples' representations far away. Moreover, hard negatives (Kalantidis et al., 2020), which are instances that are particularly challenging for models

to distinguish from the anchor sentence, are usually adopted to improve CL. We also use the training objective with the aforementioned ideas, as follows:

$$l_i = -\log \frac{e^{sim(\mathbf{h}_i, \mathbf{h}_i^+)/\tau}}{\sum_{j=1}^{N}(e^{sim(\mathbf{h}_i, \mathbf{h}_j^+)/\tau} + e^{sim(\mathbf{h}_i, \mathbf{h}_j^-)/\tau})}, \quad (1)$$

where $sim(\cdot, \cdot)$ is a similarity metric, N is the size of a mini-batch, and τ is the temperature parameter. $\mathbf{h}_i^-$ is the representation of hard negative X_i^- for anchor sentence X_i. The training objective remains the same for LLMs with or without post-training tasks.

4 Experiments

4.1 Experimental Setup

In order to demonstrate the effectiveness of SPT, we conduct experiments across models of three different sizes: base, 7B and 8B. Specifically, for base size models, we choose OPT-125M[2] (Zhang et al., 2022) while for 7B models, LLaMA2-7B[3] (Touvron et al., 2023b) serves as our backbone model. Finally, for 8B models, we select LLaMA3-8B[4]. All of them are decoder-only auto-regressive models whose `hidden_size` is 768 for OPT-125M and 4096 for LLaMA2-7B and LLaMA3-8B. Following (BehnamGhader et al., 2024), we set MNTP as the post-training task.

4.2 Implementation Details

The training dataset we use is the NLI dataset[5] from Gao et al. (2021), which is a supervised dataset containing one positive example and one hard negative example for each anchor sentence with about 275k data examples in total. We use cosine similarity as the similarity metric and τ is set to 0.05 in Equation 1. For SPT, all of our models are trained for one epoch, with evaluation on the development set of STS-B (Cer et al., 2017) and SICK-R (Marelli et al., 2014) conducted every 125 steps to find the best checkpoint. Batch size is set to 32 for all models. Learning rate is grid-searched from {0.02, 0.01, 0.005, 0.001}. Weight decay is set to 0.01 with AdamW optimizer (Loshchilov and Hutter, 2017) implemented for all models. The input sequence length is set to 32 following Gao et al. (2021). All of our experiments for SPT are conducted on one A100 80GB GPU.

4.3 Evaluation Tasks

We evaluate our models across a diverse set of tasks, including retrieval, Semantic Textual Similarity (STS), clustering and classification. Considering the input length of the NLI training dataset, we prioritize relatively shorter datasets for evaluation.

Retrieval tasks require the model to identify the most relevant sentence among a large set of documents for a specific given query sentence. The tested model will first transform the query sentences and documents into embeddings and then find the most relevant ones based on metrics such as cosine similarity. We choose the QuoraRetrieval dataset (DataCanary et al., 2017) from the MTEB benchmark (Muennighoff et al., 2023) to evaluate the retrieval performance of our models and report the nDCG@10 metric.

STS tasks evaluate the model's sentence representation ability by calculating the cosine similarity for the two given sentences after transforming them into embeddings. We utilize the SentEval (Conneau and Kiela, 2018) toolkit which includes STS12-16 (Agirre et al., 2012, 2013, 2014, 2015, 2016), STS-B (Cer et al., 2017) and SICK-R (Marelli et al., 2014). Spearman's correlation scores are reported for STS tasks.

Clustering tasks evaluate the models' ability to group sentences based on their semantic similarity, typically across different domains. The model assigns sentences to clusters such that similar sentences are grouped together, without relying on pre-defined labels. To assess our models' clustering performance, we specifically select the Twenty Newsgroup Clustering dataset (Mitchell, 1997) from MTEB and report the Validity Measure (V-measure) metric.

Classification tasks involve training an additional classifier layer on top of the tested model to evaluate its ability to correctly categorize input sentences into predefined classes. In our experiments, we specifically choose the Tweet Sentiment Extraction Classification dataset (Maggie et al., 2020) from MTEB. This task requires the model to identify and classify the sentiment (e.g., positive, nega-

[2] https://huggingface.co/facebook/opt-125m

[3] https://huggingface.co/meta-llama/Llama-2-7b-chat-hf

[4] https://huggingface.co/meta-llama/Meta-Llama-3-8B

[5] https://huggingface.co/datasets/princeton-nlp/datasets-for-simcse/resolve/main/nli_for_simcse.csv

tive, neutral) of tweets. Accuracy is reported as the evaluation metric.

4.4 Baselines

We choose several strong baselines and compare them with our models based on the four kinds of evaluation tasks. For base size models, we first choose SimCSE (Gao et al., 2021) as a commonly used encoder-only sentence embedder. Besides, we fully fine-tune OPT-125M under CL objective as a baseline. For larger size models, we first choose LLM2Vec (BehnamGhader et al., 2024) as the SOTA model. Notice that LLM2Vec is post-trained with MNTP and further fine-tuned on a larger dataset, E5 (Wang et al., 2024), using LoRA with an input sequence length of 128. The E5 dataset contains about 1.5m training examples (much bigger than 275k NLI) from different data sources such as retrieval, QA, and ranking. Due to limited computational resources and to ensure a fairer comparison between LLM2Vec and our models, we reproduce LLM2Vec with MNTP post-training using our NLI dataset by initiating from the released checkpoint[6]. Finally, we also fine-tune LLaMA models under CL with LoRA as a general baseline. We specifically set γ=16 and α=32, following settings introduced in BehnamGhader et al. (2024). Implementation details of baselines can be found in Appendix A.1.

4.5 Experimental Results

The performance of various models on four different evaluation tasks is shown in Table 1. We report results using a fixed seed=42 in our main experiments. Details of trainable parameters for each model can be found in Appendix A.2, while full results of seven STS tasks are shown in Appendix A.3.

In Table 1, models-bi. refers to models trained with bi-directional attention after post-training tasks while models-uni indicates those trained on causal attention without additional post-training tasks. The LoRA-bi. equals to LLM2Vec fine-tuned on the same NLI dataset as other models. Except for LoRA-bi., where mean pooling is used as suggested in BehnamGhader et al. (2024), all the other models use the output of the last token as sentence embedding. We will discuss the effect of different pooling methods in Section 5.1. For our SPT, we report two variants for each model: one with a soft prompt length of 1, representing the fewest trainable parameters, and the other with the optimal soft prompt length that achieves the best performance. The process for determining the best length will be discussed in Section 5.2.

Upon observing the results of base size models, we find that the best average score for the four evaluated tasks is given by the fully fine-tuned OPT-125M under CL. While encoder-only models like SimCSE outperform decoder-only models in the traditional STS task, decoder-only models excel SimCSE especially in retrieval and clustering tasks. As for our SPT, it demonstrates competitive performance with SimCSE even with a soft prompt length of just 1, with only a 0.33 point difference in the average scores. Extending the soft prompt length to 16 further narrows the gap (0.2 average performance differences) between our model and the fully fine-tuned OPT-125M under CL, despite our model updating just 0.0098% of the total parameters, compared to 100% in the fully fine-tuned model. Note that the best scores for both retrieval and classification tasks are from our SPT, with a soft prompt length of 16. From the first part of Table 1, we can see that uni-directional models show strong performance at the base size.

As for larger 7B and 8B models, we first focus on the first six rows and observe that the best average performance is achieved by our SPT in uni-directional attention without extra post-training tasks for both LLaMA2-7B and LLaMA3-8B with optimal soft prompt lengths. Their performances consistently outperform the reproduced LLM2Vec models fine-tuned on the NLI dataset (referred as LoRA-bi. in Table 1). Notably, our SPT on LLaMA2-7B with k=16 achieves the highest classification accuracy while SPT on LLaMA3-8B with k=5 delivers the best scores in retrieval, STS and classification across all four tasks. Importantly, SPT with the optimal soft prompt length requires significantly fewer trainable parameters than other baselines, underscoring the effectiveness and efficiency of our approach. Moreover, as observed in the base-sized models, even with just one trainable token (4096 parameters), SPT greatly improves LLMs' sentence embedding capabilities, trailing LoRA-based fine-tuned models by only about 1 point in average score. To this end, simply fine-tuning LLMs using SPT results in comparable or even better performances compared to fine-tuning with more trainable parameters and models with MNTP post-training.

[6] https://huggingface.co/McGill-NLP/LLM2Vec-Llama-2-7b-chat-hf-mntp

Model	Params%	Retrieval	STS	Clustering	Classification	avg.
Base models ($\leq$ 125M)						
SimCSE	100%	79.62	**81.57**	34.86	59.73	63.95
OPT w/o fine-tuning	0%	18.65	14.23	9.63	43.57	21.52
OPT w/ fine-tuning	100%	81.33	79.69	**39.46**	59.53	**65.00**
OPT w/ SPT (ours)						
-k=1	0.000613%	80.32	78.06	36.61	59.50	63.62
-k=16	0.009812%	**81.39**	78.71	39.21	**59.84**	64.79
LLaMA2-7B						
w/o fine-tuning	0%	52.93	35.48	11.69	48.39	37.12
LoRA-uni.	0.59%	85.64	**85.24**	**45.97**	61.79	69.66
LoRA-bi.	0.59%	85.24	84.43	44.07	61.30	68.76
SPT-uni. (ours)						
k=1	0.000061%	85.10	83.60	43.25	62.31	68.57
k=16	0.000973%	85.34	84.93	45.74	**62.77**	**69.70**
SPT-bi. (ours)						
-k=1	0.000061%	85.07	83.87	44.07	62.12	68.78
-k=16	0.000973%	**86.01**	84.34	45.48	62.20	69.51
LLaMA3-8B						
w/o fine-tuning	0%	48.04	28.33	21.91	44.87	35.79
LoRA-uni.	0.56%	86.65	85.87	49.63	62.81	71.24
LoRA-bi.	0.56%	85.78	85.65	47.54	63.46	70.61
SPT-uni. (ours)						
-k=1	0.000051%	85.20	84.32	48.25	62.55	70.08
-k=5	0.000255%	**87.18**	**86.00**	**49.98**	**63.75**	**71.73**
SPT-bi. (ours)						
-k=1	0.000051%	85.47	84.55	48.08	62.34	70.11
-k=5	0.000255%	87.06	85.59	49.37	63.07	71.27

Table 1: Different models' performance on four different evaluation tasks. **Params%** stands for the percentage of trainable parameters in each model. Models-bi. refers to models trained with bi-directional attention after post-training tasks, while models-uni. indicates uni-directional attention models without post-training tasks. LoRA-bi. here equals to LLM2Vec trained on the NLI dataset. We highlight the best result for each task in bold and the second-best result with an underline in each section of the table. Except for results of SimCSE, which are quoted from its paper, other results are from our own implementation.

Next, we focus on the last three rows for LLaMA2-7B and LLaMA3-8B in Table 1. To better demonstrate the advantages of naturally using causal attention, we implement SPT on bi-directional models post-trained with MNTP, referred to as SPT-bi.. In these variants, bi-directional attention is employed during post-training and fine-tuning. Comparing SPT-uni. and SPT-bi., we observe that post-training with MNTP and enabling bi-directional attention does not provide additional benefits over the natural use of causal attention without MNTP training. SPT with only a few trainable tokens on models with MNTP still achieves strong performance, particularly when the soft prompt length k is set to 1. In this case, bi-directional attention with MNTP post-training shows a slightly higher average score than the uni-directional model without MNTP, but the increase is minimal (0.2 for LLaMA2-7B and 0.03 for LLaMA3-8B). However, the better average scores are consistently achieved by SPT w/o MNTP for both LLaMA2-7B and LLaMA3-8B when setting k to the optimal length. Considering the significant extra training efforts required of post-training tasks for LLMs, we move on to discuss the necessity of applying post-training tasks for turning LLMs into text encoders in Section 5.1.

5 Discussion

5.1 Do We Really Need Bi-directional with MNTP Post-training?

In this section, we explore the usage of MNTP post-training, which is designed to help LLMs effectively get adapted to the bi-directional attention mechanism. Notably, enabling bi-directional attention allows us to prepend soft prompts to the input layer. We specifically select LLaMA2-7B and report different variants' performances on the STS task. We compare them based on the attention mechanism, the soft prompt position and the

Methods	Attention	Soft Prompt Length	Soft Prompt Position	Pooling Method	avg STS Scores
			LLaMA2-7B		
SPT w/o MNTP	bi	1	append	EOSP	82.74
	bi	16	append	EOSP	83.77
	bi	10	append	Mean	83.59
SPT w/o MNTP	bi	1	prepend	SOSP	82.09
	bi	20	prepend	SOSP	83.56
	bi	16	prepend	Mean	83.41
SPT w/ MNTP	bi	1	append	EOSP	83.87
	bi	16	append	EOSP	84.34
	bi	16	append	Mean	84.22
SPT w/ MNTP	bi	1	prepend	SOSP	83.03
	bi	10	prepend	SOSP	<u>84.56</u>
	bi	16	prepend	Mean	84.32
SPT w/o MNTP	uni	1	append	EOSP	83.60
	uni	16	append	EOSP	**84.93**
	uni	20	append	Mean	84.60

Table 2: Comparison of models with different attention mechanisms, soft prompt positions and pooling methods. EOSP refers to the end token of soft prompt while SOSP indicates the start token of soft prompt. Mean stands for the average pooling for all soft prompts. Best and second-best scores are highlighted in bold and with underline.

pooling method.

We present the average results of seven STS tasks in Table 2, reporting outcomes for both k=1 and the optimal searched k. For the optimal length k, we also include results from different pooling methods: the output of the last soft prompt token for appending (referred to as EOSP), the output of the first token for prepending (SOSP), and the average pooling of all soft prompts for both appending and prepending (Mean). Notice that the optimal k may vary across different pooling methods.

We first examine the bi-directional models in Table 2 and observe that models with MNTP consistently outperform those without MNTP across different soft prompt positions and pooling methods, a trend that demonstrates the benefits brought by post-training tasks for bi-directional models. However, when compared to our SPT with causal attention and without MNTP post-training, the highest STS score across various soft prompt lengths, positions, and pooling methods is still achieved by our simpler approach. Despite the gains brought by post-training, our results show that SPT with causal attention, without the need for MNTP post-training, can still achieve superior performance on key tasks like STS. This highlights the simplicity of leveraging causal attention naturally, offering competitive results without the added complexity and computational cost of MNTP post-training. Thus, while MNTP enhances bi-directional models, the simplicity and effectiveness of our approach make it a strong alternative for sentence embedding tasks.

5.2 Search for the Optimal Length *k*

In this section, we explore the effect of the length k for the soft prompts. We range k from {1, 2, 5, 10, 16, 20} and test them with OPT-125m, LLaMA2-7B and LLaMA3-8B on the seven STS tasks. We particularly evaluate the best settings, where causal attention is preserved and soft prompts are appended with no MNTP post-training. As shown in Figure 3 and discussed in former sections, average pooling on soft prompt tokens yields a slightly worse performance compared to using the output of the last token. For both OPT-125m and LLaMA2-7B, the best performance on STS is achieved at k=16, while for LLaMA3-8B, the optimal length is found at k=5. However, when k exceeds a certain threshold, the performance deteriorates, which is a consistent observation as noted by Li and Liang (2021). We will introduce a possible solution on how to involve more trainable parameters in Section 5.3.

5.3 More Trainable Parameters

As discussed in the aforementioned section, the performance of SPT hits its limit when the prompt length k exceeds a particular threshold. However, it is possible to implement more trainable tokens through a variant of soft prompt tuning, which is p-tuning v2 (Liu et al., 2022). Instead of only inserting trainable tokens into the input embedding layer, p-tuning v2 introduces more trainable parameters by inserting trainable tokens into each layer of the model. We specifically choose LLaMA2-7B for p-tuning v2 implementation and evaluate its

Model	Params%	Retrieval	STS	Clustering	Classification	avg.
			LLaMA2-7B			
LoRA-bi.	0.59%	85.24	84.43	44.07	61.30	68.76
LoRA-uni.						
r=1 †	0.04%	85.19	84.86	<u>46.02</u>	60.88	69.24
r=16	0.59%	**85.64**	<u>85.24</u>	45.97	61.79	69.66
SPT-uni. (ours)						
k=1	0.000061%	85.10	83.60	43.25	62.31	68.57
k=16	0.000973%	85.34	84.93	45.74	<u>62.77</u>	<u>69.70</u>
SPT v2-uni. (ours)						
k=1	0.004%	85.39	84.95	45.74	62.50	69.65
k=10	0.039%	<u>85.58</u>	**85.29**	**47.66**	**63.15**	**70.42**

Table 3: Evaluation results of SPT v2. All the models are trained on the same NLI dataset. Models-bi. refers to models trained with bi-directional attention after post-training tasks, while models-uni indicates uni-directional attention models without post-training tasks. The best and second-best results are highlighted in bold and with underline, respectively. To ensure a fair comparison with a similar number of trainable parameters, we reproduced LoRA with r=1†.

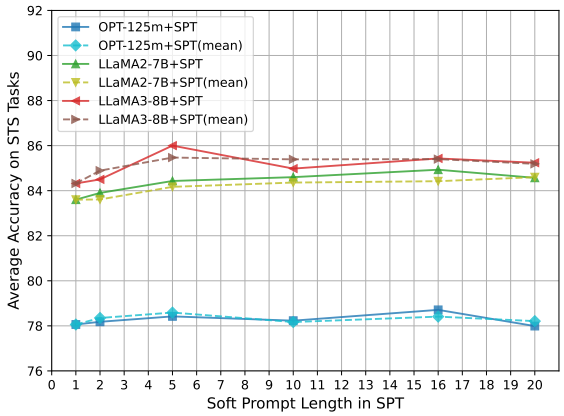

Figure 3: Performance on the STS tasks of models with different pooling methods and k values.

performance on four evaluation tasks. The results are shown in Table 3.

Notice that the official implementation of p-tuning v2[7] achieves this by prepending tokens to the `past_key_values` (one token for key and one token for value in each layer), so the actual number of trainable parameters would be `[2, num_layers, k, hidden_size]`. This method does not alter the number of input tokens. Since we maintain causal attention, we follow Liu et al. (2022) and prepend soft prompts to each layer, while still using the output of the last token as sentence embedding.

As shown in Table 3, by increasing the number of trainable parameters, our SPT v2 models outperform all other baselines fine-tuned on the same NLI dataset. Notably, our SPT v2 with k=10 achieves the best performance in three out of the four evaluated tasks, even surpassing the strong LoRA-based fine-tuned models. We reproduced LoRA with r=1, as it has nearly the same number of trainable parameters as our SPT v2 with k=10. These results demonstrate that implementing p-tuning v2 with SPT can achieve higher performance than LoRA-based fine-tuning, while requiring fewer trainable parameters. This highlights the simplicity and effectiveness of our approach in optimizing model performance with minimal parameter overhead.

6 Conclusion

In this work, we first investigate a simple method to adapt LLMs for sentence embedding by tuning a few learnable tokens. We append trainable tokens to the inputs and utilize the output of the last one as the sentence embedding. Our approach can achieve the adaptation with less than 0.001% trainable parameters, which is unattainable with LoRA. Experimental results on various tasks demonstrate that only a few tokens with our approach can achieve competitive performance with fine-tuning with LoRA. Moreover, we also find that directly using causal attention in decoder-only LLMs is capable of adapting them for sentence embedding. Specifically, our simple method with causal attention outperforms bi-directional attention baselines with extra post-training tasks, offering insights on the adaptation of LLMs for sentence embedding.

[7] https://github.com/THUDM/P-tuning-v2

Limitations

In this work, we demonstrate the effectiveness of using soft prompts for sentence representation in LLMs. However, whether this kind of adaptation works in other tasks remains unclear. The optimal soft prompt length relies on various factors, including the training dataset and model size, which require extra searches. Also, the multilingual scenario could be taken into consideration.

Ethics Statement

This paper aims at adapting LLMs for sentence representation with low cost. All the data and models we used are publicly open-sourced and contain no privacy-related ones. We agree to the license and privacy policy of the corresponding models and datasets. We try to make a positive contribution to the community. Our work does not introduce ethical biases as well. We use ChatGPT to check the grammar in our writing.

Acknowledgment

Kaiyan Zhao was supported by JST SPRING, Grant Number JPMJSP2108.

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.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. 2024. LLM2vec: Large language models are secretly powerful text encoders. In *First Conference on Language Modeling*.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel Ziegler, Jeffrey Wu, Clemens Winter, Chris Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc.

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.

Yung-Sung Chuang, Rumen Dangovski, Hongyin Luo, Yang Zhang, Shiyu Chang, Marin Soljacic, Shang-Wen Li, Scott Yih, Yoon Kim, and James Glass. 2022. DiffCSE: Difference-based contrastive learning for sentence embeddings. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 4207–4218, Seattle, United States. Association for Computational Linguistics.

Alexis Conneau and Douwe Kiela. 2018. SentEval: An evaluation toolkit for universal sentence representations. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018)*, Miyazaki, Japan. European Language Resources Association (ELRA).

Alexis Conneau, Douwe Kiela, Holger Schwenk, Loïc Barrault, and Antoine Bordes. 2017. Supervised learning of universal sentence representations from natural language inference data. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 670–680, Copenhagen, Denmark. Association for Computational Linguistics.

DataCanary, hilfialkaff, Jiang Lili, Risdal Meg, Dandekar Nikhil, and tomtung. 2017. Quora question pairs.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Conference on Empirical Methods in Natural Language Processing*.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. 2022. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations*.

Ting Jiang, Shaohan Huang, Zhongzhi Luan, Deqing Wang, and Fuzhen Zhuang. 2023. Scaling sentence embeddings with large language models. *Preprint*, arXiv:2307.16645.

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*, pages 8826–8837, Abu Dhabi, United Arab Emirates. 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.

Yannis Kalantidis, Mert Bulent Sariyildiz, Noe Pion, Philippe Weinzaepfel, and Diane Larlus. 2020. Hard negative mixing for contrastive learning. *Advances in Neural Information Processing Systems*, 33:21798–21809.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-tuning: Optimizing continuous prompts for generation. 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 4582–4597, Online. Association for Computational Linguistics.

Xianming Li and Jing Li. 2023. Angle-optimized text embeddings. *Preprint*, arXiv:2309.12871.

Xianming Li and Jing Li. 2024. BeLLM: Backward dependency enhanced large language model for sentence embeddings. In *Proceedings of the 2024 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (Volume 1: Long Papers)*, pages 792–804, Mexico City, Mexico. Association for Computational Linguistics.

Xiao Liu, Kaixuan Ji, Yicheng Fu, Weng Tam, Zhengxiao Du, Zhilin Yang, and Jie Tang. 2022. P-tuning: Prompt tuning can be comparable to fine-tuning across scales and tasks. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 61–68, Dublin, Ireland. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. In *International Conference on Learning Representations*.

Ang Lv, Kaiyi Zhang, Shufang Xie, Quan Tu, Yuhan Chen, Ji-Rong Wen, and Rui Yan. 2023. Are we falling in a middle-intelligence trap? an analysis and mitigation of the reversal curse. *arXiv preprint arXiv:2311.07468*.

Maggie, Culliton Phil, and Chen Wei. 2020. Tweet sentiment extraction.

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. In *Findings of the Association for Computational Linguistics: NAACL 2024*, pages 3225–3236, Mexico City, Mexico. Association for Computational Linguistics.

Tom Mitchell. 1997. Twenty Newsgroups. UCI Machine Learning Repository. DOI: https://doi.org/10.24432/C5C323.

Niklas Muennighoff, Hongjin Su, Liang Wang, Nan Yang, Furu Wei, Tao Yu, Amanpreet Singh, and Douwe Kiela. 2024. Generative representational instruction tuning. *arXiv preprint arXiv:2402.09906*.

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.

OpenAI. 2022. Introducing chatgpt. `https://openai.com/blog/chatgpt`.

OpenAI. 2023. Gpt-4 technical report.

Liangyang Ouyang, Ruicong Liu, Yifei Huang, Ryosuke Furuta, and Yoichi Sato. 2024. Actionvos: Actions as prompts for video object segmentation. *arXiv preprint arXiv:2407.07402*.

Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. In *Conference on Empirical Methods in Natural Language Processing*.

Zhengxiang Shi and Aldo Lipani. 2024. Dept: Decomposed prompt tuning for parameter-efficient fine-tuning. In *International Conference on Learning Representations*.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. Llama: Open and efficient foundation language models. *ArXiv*, abs/2302.13971.

Hugo Touvron, Louis Martin, Kevin R. Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Daniel M. Bikel, Lukas Blecher, Cristian Cantón Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony S. Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel M. Kloumann, A. V. Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta, Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, R. Subramanian, Xia Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zhengxu Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. *ArXiv*, abs/2307.09288.

Aäron van den Oord, Yazhe Li, and Oriol Vinyals. 2018. Representation learning with contrastive predictive coding. *ArXiv*, abs/1807.03748.

Liang Wang, Nan Yang, Xiaolong Huang, Binxing Jiao, Linjun Yang, Daxin Jiang, Rangan Majumder, and Furu Wei. 2024. Text embeddings by weakly-supervised contrastive pre-training. *Preprint*, arXiv:2212.03533.

Ledell Wu, Adam Fisch, Sumit Chopra, Keith Adams, Antoine Bordes, and Jason Weston. 2018. Starspace: Embed all the things! In *Proceedings of the AAAI conference on artificial intelligence*, volume 32.

Qiyu Wu, Chongyang Tao, Tao Shen, Can Xu, Xiubo Geng, and Daxin Jiang. 2022. PCL: Peer-contrastive learning with diverse augmentations for unsupervised sentence embeddings. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 12052–12066, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona T. Diab, Xian Li, Xi Victoria Lin, Todor Mihaylov, Myle Ott, Sam Shleifer, Kurt Shuster, Daniel Simig, Punit Singh Koura, Anjali Sridhar, Tianlu Wang, and Luke Zettlemoyer. 2022. Opt: Open pre-trained transformer language models. *ArXiv*, abs/2205.01068.

Kaiyan Zhao, Qiyu Wu, Xin-Qiang Cai, and Yoshimasa Tsuruoka. 2024. Leveraging multi-lingual positive instances in contrastive learning to improve sentence embedding. In *Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 976–991, St. Julian's, Malta. Association for Computational Linguistics.

Kaiyang Zhou, Jingkang Yang, Chen Change Loy, and Ziwei Liu. 2022. Learning to prompt for vision-language models. *International Journal of Computer Vision*, 130(9):2337–2348.

A Appendix

A.1 Baseline Implementation Details

We introduce the implementation details for baseline reproduction in this section. For baselines directly fine-tuned with Low-Rank Adaptation (LoRA), we follow the implementation introduced in Jiang et al. (2023), with the modification of changing the batch size to 32 and evaluating the model every 125 steps on the development set (compared to 50 steps in the original paper). α is consistently set to 16. We do not include the proposed PromptEOL method in Jiang et al. (2023) for a fairer comparison and utilize the output of the last input token as the sentence embedding. The baseline reproduction is carried out on two A100 80GB GPUs.

For other baselines with Masked Next Token Prediction (MNTP) post-training, we initiate the models from the released checkpoint, while training with SPT follows the settings introduced in Section 4.2.

A.2 Details of Trainable Parameters

We count the trainable parameters based on PEFT library [8]. The number of trainable parameters, total parameters and the percentage of trainable parameters for each model are shown in Table 4. The hidden state size for OPT-125M and LLaMA is 768 and 4096, respectively.

Model	Trainable Param	Total Param	Percentage
OPT-125M + CL	125239296	125239296	100%
OPT-125M + SPT, k=1	768	125240064	0.00061%
OPT-125M + SPT, k=16	12288	125251584	0.0098%
LLaMA2 + LoRA r=1	2498560	6740914176	0.04%
LLaMA2 + LoRA r=16	39976960	6778392576	0.59%
LLaMA2 + LLM2Vec r=16	39976960	6778392576	0.59%
LLaMA2 + SPT, k=1	4096	6738419712	0.000061%
LLaMA2 + SPT, k=2	8192	6738423808	0.00012%
LLaMA2 + SPT, k=5	20480	6738436096	0.0003%
LLaMA2 + SPT, k=10	40960	6738456576	0.00061%
LLaMA2 + SPT, k=16	65536	6738481152	0.00097%
LLaMA2 + SPT, k=20	81920	6738497536	0.0012%
LLaMA2 + SPT v2, k=1	262144	6738677760	0.004%
LLaMA2 + SPT v2, k=10	2621440	6741037056	0.04%
LLaMA3 + LoRA, r=16	45088768	8075350016	0.56%
LLaMA3 + LLM2Vec, r=16	45088768	8075350016	0.56%
LLaMA3 + SPT, k=1	4096	8030265344	0.000051%
LLaMA3 + SPT, k=2	8192	8030269440	0.00001%
LLaMA3 + SPT, k=5	20480	8030281728	0.00026%
LLaMA3 + SPT, k=10	40960	8030302208	0.00051%
LLaMA3 + SPT, k=16	65536	8030326784	0.0008%
LLaMA3 + SPT, k=20	81920	8030343168	0.001%

Table 4: Comparison of trainable parameters.

[8] https://huggingface.co/docs/peft/main/en/index

A.3 Full STS Results

We show full results on seven STS tasks in Table 5.

Model	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	avg.
Base models ($\leq$ 125M)								
SimCSE	75.30	84.67	80.19	85.40	80.82	84.25	80.39	81.57
OPT w/o fine-tuning †	7.47	9.48	8.30	19.63	22.45	7.40	24.91	14.23
OPT w/ fine-tuning	73.80	81.97	77.58	83.42	79.50	83.05	78.51	79.69
OPT w/ SPT, k=1 **(ours)**	71.72	80.92	74.66	82.77	79.87	81.41	75.09	78.06
OPT w/ SPT, k=16 **(ours)**	73.43	81.03	75.57	82.73	79.26	82.33	76.60	78.71
LLaMA2-7B								
w/o fine-tuning †	22.30	30.92	27.10	38.92	52.95	33.66	42.54	35.48
LoRA w/o MNTP, *r*=16	78.29	89.11	84.26	88.97	85.36	87.83	82.34	85.24
LoRA w/ MNTP (LLM2Vec, NLI)	78.01	87.96	83.06	88.45	85.51	87.57	80.44	85.24
SPT w/o MNTP, *k*=1 **(ours)**	75.46	87.91	82.84	87.00	84.66	87.15	80.16	83.60
SPT w/o MNTP, *k*=16 **(ours)**	76.53	89.12	83.26	89.21	85.21	88.34	82.86	84.93
SPT w/ MNTP, *k*=1 **(ours)**	76.60	87.70	81.97	88.38	84.07	87.35	81.00	83.87
SPT w/ MNTP, *k*=16 **(ours)**	76.13	88.54	82.69	88.82	85.12	87.80	81.29	84.34
LLaMA3-8B								
w/o fine-tuning †	10.35	38.69	24.72	34.55	37.46	23.07	29.49	28.33
LoRA w/o MNTP, *r*=16	79.04	89.66	85.95	89.41	85.96	88.54	82.57	85.87
LoRA w/ MNTP (LLM2Vec, NLI)	78.59	89.67	85.40	89.83	85.16	88.41	82.46	85.65
SPT w/o MNTP, *k*=1 **(ours)**	75.20	88.79	83.60	88.60	84.15	87.82	82.05	84.32
SPT w/o MNTP, *k*=5 **(ours)**	78.61	90.23	85.10	89.53	86.87	89.33	82.38	86.00
SPT w/ MNTP, *k*=1 **(ours)**	76.59	88.11	84.16	88.97	85.16	87.03	81.87	84.55
SPT w/ MNTP, *k*=5 **(ours)**	78.74	89.63	84.72	89.30	86.04	87.92	82.81	85.59

Table 5: Full results of seven STS tasks. † marks models without further training, for which we take the output of last input token as sentence embedding. Results with * are quoted from the MTEB leaderboard (Muennighoff et al., 2023). Results of SimCSE is quoted from its paper.