Contrastive Prompting Enhances Sentence Embeddings in LLMs through Inference-Time Steering

Zifeng Cheng* **Zhonghui Wang*** **Yuchen Fu*** **Zhiwei Jiang†**
Yafeng Yin **Cong Wang** **Qing Gu**
State Key Laboratory for Novel Software Technology, Nanjing University, China
{chengzf,yuchenfu,zhonghuiwang}@smail.nju.edu.cn,
{jzw,yafeng}@nju.edu.cn, cw@smail.nju.edu.cn, guq@nju.edu.cn

Abstract

Extracting sentence embeddings from large language models (LLMs) is a practical direction, as it requires neither additional data nor fine-tuning. Previous studies usually focus on prompt engineering to guide LLMs to encode the core semantic information of the sentence into the embedding of the last token. However, the last token in these methods still encodes an excess of non-essential information, such as stop words, limiting its encoding capacity. To this end, we propose a Contrastive Prompting (CP) method that introduces an extra auxiliary prompt to elicit better sentence embedding. By contrasting with the auxiliary prompt, CP can steer existing prompts to encode the core semantics of the sentence, rather than non-essential information. CP is a plug-and-play inference-time intervention method that can be combined with various prompt-based methods. Extensive experiments on Semantic Textual Similarity (STS) tasks and downstream classification tasks demonstrate that our method can improve the performance of existing prompt-based methods across different LLMs. Our code will be released at https://github.com/zifengcheng/CP.

1 Introduction

Sentence embeddings (Nie et al., 2024) play a fundamental role in various real-world applications, such as information retrieval, text classification, clustering, and so on. Considering that large language models (LLMs) have achieved success in zero-shot settings across various tasks, some works (Liu et al., 2024a; Lei et al., 2024; Fu et al., 2024) have started to directly extract sentence embeddings from the hidden states of LLMs without the need for additional data or fine-tuning. Since data are likely to be scarce in practice and the cost of fine-tuning LLM is expensive, such zero-shot setting is more practical and promising, while preserving the general capabilities of LLMs.

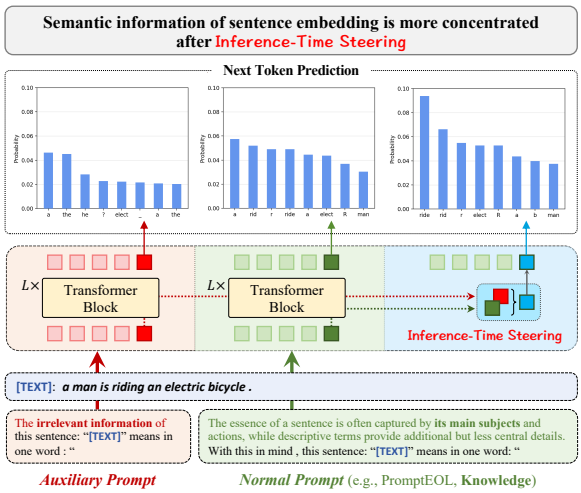

Figure 1: The comparison of sentence embeddings obtained by the auxiliary prompt, normal prompt, and our proposed inference-time steering method. The decoding probabilities of Next Token Prediction are used to reflect the semantic information contained in the corresponding sentence embeddings.

Existing works (Jiang et al., 2023; Lei et al., 2024; Zhang et al., 2024) typically focus on prompt engineering to compress the full semantics of a sentence into the last token and use the hidden state of that token as the sentence embedding. PromptEOL (Jiang et al., 2023) first utilizes a simple and effective prompt: *This sentence:* "[TEXT]" *means in one word: "*, to generate sentence embeddings, where [TEXT] serves as the sentence slot. Subsequently, some works (Lei et al., 2024; Zhang et al., 2024) have designed various prompts to enable the last token to capture the core semantics of the sentence, rather than focusing on non-essential information. Specifically, MetaEOL (Lei et al., 2024) uses a diverse set of meta-task prompts. Pretended CoT (Zhang et al., 2024) employs a Chain-of-Thought prompt. Knowledge (Zhang et al.,

*Equal Contribution.
†Corresponding Author.

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 3475–3487
July 27 - August 1, 2025

2024) directs the LLM to focus more on the sentence's main subjects and actions.

Existing methods can be viewed as an *indirect* approach, where different prefix prompts *indirect* alter the representation of the last token, making it more focused on the core semantics of the sentence. However, through the pilot next token prediction decoding experiment, their method still overly encodes non-essential information, such as stopwords, as shown in Figure 1. Although the Knowledge (Zhang et al., 2024) prompt emphasizes grasping the main subjects and actions, the token with the highest probability remains the non-essential stopword "a".

In this paper, we propose a simple and effective **C**ontrastive **P**rompting (**CP**) method that can be combined with existing methods. CP additionally introduces an auxiliary prompt to encode non-essential information of the sentence and to contrast it with the normal prompt (i.e., existing prompt-based methods), enabling CP to *directly* modify the hidden state of the last token in the normal prompt during inference. By contrasting with the auxiliary prompt, CP can steer existing prompts to encode the core semantics of the sentence while filtering out non-essential information. Specifically, we first forward propagate the sentences wrapped with the auxiliary and normal prompts to the specific layer and extract the sentence embeddings. Then, we directly compare the embeddings of the two sentences and replace the representation of the last token in the normal prompt to steer its focus toward the core semantics of the sentence. Due to the change in the norm of the sentence embedding before and after the intervention, we propose two strategies to control the norm of the sentence embedding after the intervention. Finally, we continue to forward propagate the normal prompt to obtain the sentence embeddings.

Our main contributions are as follows:

- We first propose enhancing the quality of sentence embeddings through inference-time activation steering.
- We propose the Contrastive Prompting method, which guides LLMs to encode the core semantics of a sentence into its embedding. Several specifically designed auxiliary prompts are explored, and two norm adjustment strategies are introduced during activation steering.
- We conduct extensive experiments on Semantic Textual Similarity (STS) benchmarks and downstream classification tasks. Experimental results demonstrate that our proposed method significantly improves the performance of existing prompt-based methods across different LLMs. Additionally, since the auxiliary prompt only needs to propagate to the lower layers of LLMs, the extra time overhead is relatively minimal.

2 Related Work

Sentence Embeddings Sentence embedding aims to represent the semantic information of a sentence into a fixed-size vector representation. Previous methods often focus on various data augmentation techniques and contrastive losses to fine-tune smaller pre-trained language models for enhancing sentence embeddings (Gao et al., 2021; Jiang et al., 2022; Ni et al., 2022b; Chanchani and Huang, 2023; Su et al., 2023). Due to the exceptional capabilities of LLMs, recent works (Li and Li, 2024; BehnamGhader et al., 2024; Lee et al., 2024; Muennighoff et al., 2024) have begun fine-tuning LLMs to obtain sentence embeddings. In addition, some studies (Li et al., 2025; Zhuang et al., 2024) focus on extracting scalable sentence embeddings from the intermediate layers of language models. However, these methods require data and fine-tuning, leading to a high cost and a loss of LLMs' other general capabilities. Thus, this paper focuses on directly extracting sentence embeddings from LLMs without the need for fine-tuning or data.

Extracting Sentence Embeddings from LLMs Existing methods on extracting sentence embeddings from LLMs mainly focus on prompt engineering. PromptEOL (Jiang et al., 2023) first demonstrates the potential of LLMs in generating sentence embeddings by leveraging prompt engineering and compressing the semantics of a sentence into a single token. Echo embeddings (Springer et al., 2024) repeat the input twice within the context, allowing later tokens to access earlier tokens, and extract embeddings from the second occurrence. MetaEOL (Lei et al., 2024) designs meta-task prompts via ChatGPT-4 to guide LLMs to consider sentence representations from multiple perspectives. Pretended CoT (Zhang et al., 2024) uses CoT to inspire LLMs to output better embeddings. Knowledge Enhancement (Zhang et al., 2024) directs the LLM to focus more on the sentence's main subjects and

actions through prompts. In this paper, we propose a plug-and-play method to further improve the various prompt-based methods.

Activation Steering Activation steering (Zou et al., 2023; Rimsky et al., 2024; Li et al., 2023; Leong et al., 2023) creates a steering vector to modify the activations of the LLM, thereby controlling its generation. The steering vector is derived by calculating the difference between activations from pairs of positive and negative supervision samples. In contrast to these methods, we generate an activation vector for each sentence using two different prompts to refine the representation of the normal prompt, without the need for supervised data.

3 Preliminary

Extracting Embeddings from LLMs Previous work mainly focuses on prompt engineering to extract sentence embeddings from LLMs. PromptEOL (Jiang et al., 2023) introduces a widely adopted template for extracting sentence embeddings from LLMs:

This sentence: "[TEXT]" means in one word: "

where [TEXT] denotes the placeholder for the input sentence and the hidden state of the last token " is considered as the sentence embedding. The phrase "in one word" is a constraint that can prevent LLMs from generating long sentences, limiting a sentence to being represented by the embedding of a single word.

Multi-Head Attention in LLMs The ℓ-th multi-head attention layer (Vaswani et al., 2017) contains three projection matrices $W_Q^\ell, W_K^\ell, W_V^\ell \in \mathbb{R}^{d\times d}$ and an output matrix $W_O^\ell \in \mathbb{R}^{d\times d}$. The columns of each projection matrix and the rows of the output matrix can be split into H heads, yielding $W_Q^{\ell,h}, W_K^{\ell,h}, W_V^{\ell,h} \in \mathbb{R}^{d\times \frac{d}{H}}$ for $h \in [1, H]$. The h-th attention head computes the attention weight matrix $A^{\ell,h} \in \mathbb{R}^{N\times N}$ as follows:

$$A^{\ell,h} = \varphi\left(\frac{\left(\mathbf{x}^{\ell-1}W_Q^{\ell,h}\right)\left(\mathbf{x}^{\ell-1}W_K^{\ell,h}\right)^T}{\sqrt{d/H}} + M^{\ell,h}\right),$$

where φ denotes the row-wise softmax function, $\mathbf{x}^{\ell-1}$ denotes the output of $(\ell-1)$-th Transformer layer, and $M^{\ell,h}$ is a mask matrix that ensures the attention is causal. Then, the output of h-th multi-head attention can be computed as follows:

$$\mathbf{v}^{\ell,h} = A^{\ell,h}\left(\mathbf{x}^{\ell-1}W_V^{\ell,h}\right)$$

where each $\mathbf{v}_i^{\ell,h} \in \mathbb{R}^d$ is a contextualized value vector of h-head at position i. Once all heads have computed their individual outputs, these outputs are concatenated and passed through the output matrix:

$$\begin{aligned}\mathbf{a}^\ell &= \text{concat}(\mathbf{v}^{\ell,1}, \cdots, \mathbf{v}^{\ell,H})W_O^\ell \\ &= \mathbf{v}^\ell W_O^\ell\end{aligned}$$

Compared to the FFN block, multi-head attention directly facilitates information interaction between tokens, and we intervene on the contextualized value vectors.

4 Method

Our proposed CP method is a plug-and-play inference-time intervention algorithm that requires no additional data or fine-tuning of the LLM, and can integrate with existing prompt engineering techniques. CP consists of three steps to obtain sentence embeddings and requires forward propagation for both normal and auxiliary prompts, as illustrated in Figure 2. In the first step, it inputs the auxiliary prompt into the LLM up to the ℓ-th multi-head attention layer to obtain the non-essential information vector. Next, it performs forward propagation with the normal prompt and use the non-essential information vector to contrast with it, steering its representation to better emphasize the core semantics. Finally, we adjust the norm of contextualized value vector before and after the intervention, and forward propagate the adjusted value vectors to extract the sentence embeddings.

4.1 Auxiliary Prompt

In the first step, we construct an auxiliary prompt to extract non-essential information in the sentence. Specifically, we use the auxiliary prompt (i.e., The irrelevant information of this sentence: "[TEXT]" means in one word: ") to wrap the text and feed it into the LLM to obtain the contextualized value vectors from the ℓ-th multi-head attention layer.

Formally, given the input $T_{\text{aux}} = [t_1, ..., t_{N_{\text{aux}}}]$ wrapped in the auxiliary template, we pass them into the ℓ-th Transformer layers of LLMs and obtain the auxiliary contextualized value vectors (i.e., $\mathbf{v}^{\text{aux},(\ell)} = [\mathbf{v}_1^{\text{aux},(\ell)}, \cdots, \mathbf{v}_{N_{\text{aux}}}^{\text{aux},(\ell)}]$) from the ℓ-th multi-head attention layer.

The overhead of introducing an auxiliary prompt is minimal, as it only needs to propagate to the ℓ-th layer, rather than all layers. Additionally, for methods with multiple prompts, compared to

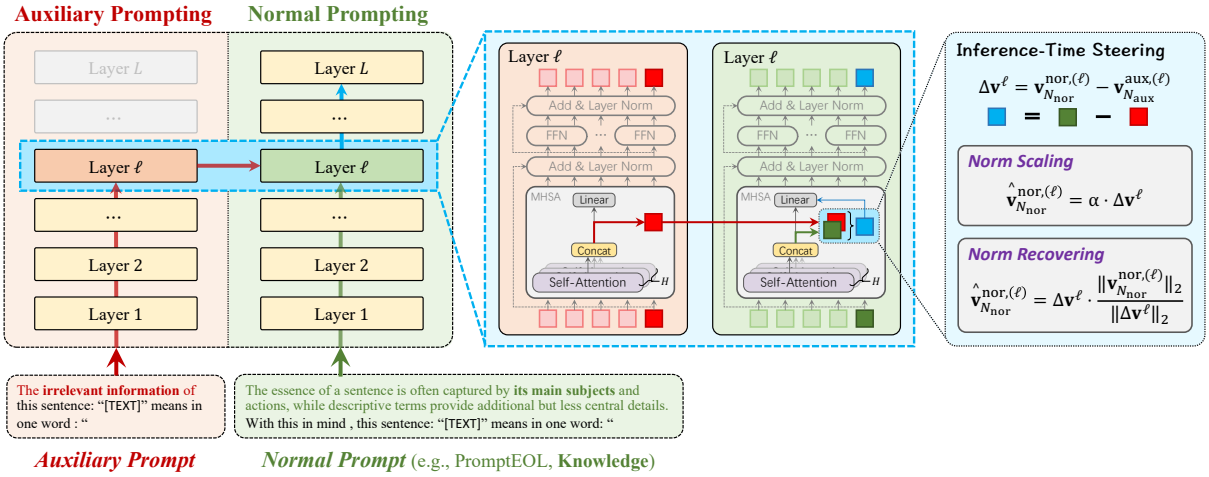

Figure 2: Illustration of the contrastive prompting method.

each normal prompt requiring propagation through the middle and later layers of the LLM, the auxiliary prompt only needs to propagate through the lower layers of the LLM and can refine all normal prompts with a single auxiliary prompt, further reducing the overhead.

4.2 Contrastive Activation Steering

In the second step, it performs forward propagation with the normal prompt and use the non-essential information vectors to intervene and refine the normal prompt.

We also use the normal prompt to wrap the text and feed the input $T_{\text{nor}} = [t_1, ..., t_{N_{\text{nor}}}]$ into the LLM to obtain the contextualized value vectors (i.e., $\mathbf{v}^{\text{nor},(\ell)} = [\mathbf{v}_1^{\text{nor},(\ell)}, \cdots, \mathbf{v}_{N_{\text{nor}}}^{\text{nor},(\ell)}]$) from the ℓ-th multi-head attention layer. The normal prompt can be any existing prompt-based methods, such as PromptEOL (Jiang et al., 2023), Pretended CoT (Zhang et al., 2024), and Knowledge (Zhang et al., 2024).

Then, we obtain the semantic activation vector $\Delta\mathbf{v}^{\ell}$ by contrasting the contextualized value vectors derived from the normal and auxiliary prompts. Intuitively, the result of the difference removes the non-essential information, allowing it to focus more on the core semantics of a sentence. Specifically, this vector is calculated as:

$$\Delta\mathbf{v}^{\ell} = \mathbf{v}_{N_{\text{nor}}}^{\text{nor},(\ell)} - \mathbf{v}_{N_{\text{aux}}}^{\text{aux},(\ell)}. \tag{1}$$

The semantic activation vector is sentence-dependent, as the contextualized value vectors (i.e., $\mathbf{v}_{N_{\text{nor}}}^{\text{nor},(\ell)}$ and $\mathbf{v}_{N_{\text{aux}}}^{\text{aux},(\ell)}$) for each sentence differ. Since we focus solely on sentence embeddings and the lengths of the normal and auxiliary prompts may differ, we intervene only on the last token.

4.3 Norm Adjustment

In the third step, it adjusts the norm of the contextualized value vector after intervention, and then continues the forward propagation with the adjusted contextualized value vector. Although we obtain semantic activation vectors, the norm of the contextualized value vector significantly changes after intervention. Thus, we further propose two strategies for adjusting the norm: norm scaling and norm recovering.

Norm Scaling (NS) additionally introduces a hyperparameter to control the norm:

$$\hat{\mathbf{v}}_{N_{\text{nor}}}^{\text{nor},(\ell)} = \alpha \cdot \Delta\mathbf{v}^{\ell}, \tag{2}$$

where α is a scaling factor that controls the extent of contrastive activation steering.

Norm Recovering (NR) strategy ensures that the norm remains consistent before and after intervention and avoids the introduction of additional hyperparameters. NR strategy can ensure that the subsequent output matrix receives inputs similar to the original ones, thereby maintaining the model's capabilities. Specifically, we renormalize the updated value vector to align with the L^2-norm before the intervention:

$$\hat{\mathbf{v}}_{N_{\text{nor}}}^{\text{nor},(\ell)} = \Delta\mathbf{v}^{\ell} \cdot \frac{\|\mathbf{v}_{N_{\text{nor}}}^{\text{nor},(\ell)}\|_2}{\|\Delta\mathbf{v}^{\ell}\|_2}. \tag{3}$$

After getting $\hat{\mathbf{v}}_{N_{\text{nor}}}^{\text{nor},(\ell)}$, we further use $\hat{\mathbf{v}}_{N_{\text{nor}}}^{\text{nor},(\ell)}$ to replace $\mathbf{v}_{N_{\text{nor}}}^{\text{nor},(\ell)}$ and obtain the new input $\hat{\mathbf{v}}^{\text{nor},(\ell)}$ for the output matrix. Specifically,

$$\hat{\mathbf{v}}^{\text{nor},(\ell)} = [\mathbf{v}_1^{\text{nor},(\ell)}, \cdots, \hat{\mathbf{v}}_{N_{\text{nor}}}^{\text{nor},(\ell)}]. \tag{4}$$

Method	Params	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
BERT avg	110M	30.87	59.89	47.73	60.29	63.73	47.29	58.22	52.57
BERT prompt†	110M	60.96	73.83	62.18	71.54	68.68	70.60	67.16	67.85
ST5-Enc avg†	4.8B	34.97	60.19	47.59	66.40	70.62	62.83	63.57	58.02
LLaMA2 avg	7B	35.49	53.15	40.12	55.35	53.26	42.10	49.96	47.06
LLaMA2 echo†	7B	52.40	72.40	61.24	72.67	73.51	65.73	64.39	66.05
MetaEOL†	7B	64.16	81.61	73.09	81.11	78.94	77.96	74.86	75.96
PromptEOL†	7B	58.81	77.01	66.34	73.22	73.56	71.66	69.64	70.03
PromptEOL + CP-NS (***Ours***)	7B	63.34 ↑ 4.53	82.15 ↑ 5.14	71.73 ↑ 5.39	79.68 ↑ 6.46	77.23 ↑ 3.67	78.71 ↑ 7.05	74.04 ↑ 4.40	75.27 ↑ 5.24
PromptEOL + CP-NR (***Ours***)	7B	63.37 ↑ 4.56	81.95 ↑ 4.94	71.90 ↑ 5.56	79.54 ↑ 6.32	77.29 ↑ 3.73	78.36 ↑ 6.70	74.00 ↑ 4.36	75.20 ↑ 5.17
Pretended CoT†	7B	67.45	83.89	74.14	79.47	80.76	78.95	73.33	76.86
Pretended CoT + CP-NS (***Ours***)	7B	67.79 ↑ 0.34	83.66 ↓ 0.23	74.52 ↑ 0.38	81.10 ↑ 1.63	80.70 ↓ 0.06	80.39 ↑ 1.44	74.01 ↑ 0.68	77.45 ↑ 0.59
Pretended CoT + CP-NR (***Ours***)	7B	67.62 ↑ 0.17	83.69 ↓ 0.20	74.53 ↑ 0.41	81.13 ↑ 1.66	80.74 ↓ 0.02	80.39 ↑ 1.44	74.02 ↑ 0.69	77.45 ↑ 0.59
Knowledge†	7B	65.60	82.82	74.48	80.75	80.13	80.34	75.89	77.14
Knowledge + CP-NS (***Ours***)	7B	67.16 ↑ 1.56	83.43 ↑ 0.61	74.23 ↓ 0.25	81.29 ↑ 0.54	80.03 ↓ 0.10	80.80 ↑ 0.46	75.97 ↑ 0.08	77.56 ↑ 0.42
Knowledge + CP-NR (***Ours***)	7B	66.65 ↑ 1.05	83.21 ↑ 0.39	74.21 ↓ 0.27	81.19 ↑ 0.44	79.74 ↓ 0.39	80.70 ↑ 0.36	76.07 ↑ 0.18	77.40 ↑ 0.26
CK	7B	67.11	84.03	75.07	82.42	80.91	81.84	76.24	78.23
CK + CP-NS (***Ours***)	7B	68.35 ↑ 1.24	84.21 ↑ 0.18	75.59 ↑ 0.52	82.49 ↑ 0.07	81.50 ↑ 0.59	82.29 ↑ 0.45	76.34 ↑ 0.10	78.68 ↑ 0.45
CK + CP-NR (***Ours***)	7B	68.09 ↑ 0.98	84.14 ↑ 0.11	75.58 ↑ 0.51	82.46 ↑ 0.04	81.39 ↑ 0.48	82.17 ↑ 0.33	76.39 ↑ 0.15	78.60 ↑ 0.37

Table 1: Results on STS tasks using LLaMA2-7B as the backbone. †denotes the results from the original paper.

Next, we feed the adjusted contextualized value vector $\hat{\mathbf{v}}^{\text{nor},(\ell)}$ into the output matrix W_O^ℓ of the ℓ-th multi-head attention layer and continue the standard forward propagation to extract sentence embeddings.

4.4 Intermediate Embedding Eliciting

Recent studies (Liu et al., 2024b; Jin et al., 2024) have confirmed that each layer of the LLM serves a different purpose, with the embeddings extracted from the final layer primarily used for prediction, and they may not yield the best performance. Thus, we use embeddings extracted from intermediate layers, rather than the final layer, as sentence embeddings. Extracting embeddings from the intermediate layers not only provides high quality embeddings but also avoids the high cost of propagating through to the final layer, thereby accelerating the extraction of embeddings. We can use the validation set to determine which layer's embeddings to use, and the overhead of this process is lightweight.

5 Experiments

5.1 Datasets and Experimental Settings

We evaluate sentence embeddings on seven semantic textual similarity (STS) datasets, including STS 2012-2016 (Agirre et al., 2012, 2013, 2014, 2015, 2016), STS-B (Cer et al., 2017), and SICK-R (Marelli et al., 2014). Each sentence pair in these datasets is annotated with a pairwise semantic similarity score ranging from 0 to 5. We use cosine similarity to calculate the predicted similarity scores and evaluate them using Spearman correlation, which assesses the degree of rank correlation between the predicted similarity scores and the annotated similarity scores.

We use grid search on the STS-B development set to search for the intervention layer ℓ in {3, 4, 5, 6, 7} and the scaling factor of the norm scaling α in {0.5, 1, 2, 3, 4} for each prompt. The intervention layer of PromptEOL is the 5th layer, while the intervention layer of Pretended CoT and Knowledge is the 7th layer. The norm scaling of PromptEOL is 2, while the norm scaling of Pretended CoT and Knowledge is 3. We use the 27th layer as the output layer for PromptEOL and Pretended CoT, while the penultimate layer is used for Knowledge.

5.2 Baselines

We combine our method with some baselines to demonstrate effectiveness. **BERT avg** (Devlin et al., 2019), **ST5-Enc avg** (Ni et al., 2022a), and **LLaMA2 avg** (Touvron et al., 2023) generate sentence embeddings by averaging all token embeddings, each utilizing a different backbone. **BERT prompt** (Jiang et al., 2022) proposes to represent a sentence with a prompt using BERT. **LLaMA2 echo** (Springer et al., 2024) repeats the input twice in context and uses mean-pooling to extract embeddings from the second occurrence. **PromptEOL** (Jiang et al., 2023) compresses the semantics of a sentence into a single token to extract embeddings. **MetaEOL** (Lei et al., 2024) leverages a diverse set of meta-task prompts to capture multiple representations of sentences from distinct perspectives. **Pretended CoT** (Zhang et al., 2024) uses CoT to inspire LLMs to extract better sentence embeddings. **Knowledge** (Zhang et al., 2024) provides explicit guidance to LLMs by conveying human experience in text summarization

Method	Backbone	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
Pretended CoT	LLaMA2-7B	67.45	83.89	74.14	79.47	80.76	78.95	73.33	76.86
Pretended CoT + CP-NS (***Ours***)	LLaMA2-7B	66.70 ↓ 0.75	84.26 ↑ 0.37	74.44 ↑ 0.30	81.39 ↑ 1.92	80.71 ↓ 0.05	80.56 ↑ 1.61	74.07 ↑ 0.74	77.45 ↑ 0.59
Pretended CoT + CP-NR (***Ours***)	LLaMA2-7B	66.77 ↓ 0.68	84.31 ↑ 0.42	74.55 ↑ 0.41	81.45 ↑ 1.98	80.73 ↓ 0.03	80.51 ↑ 1.56	74.08 ↑ 0.75	77.49 ↑ 0.63
Pretended CoT	LLaMA2-13B	64.27	78.61	69.93	76.37	79.28	75.88	69.04	73.34
Pretended CoT + CP-NS (***Ours***)	LLaMA2-13B	64.41 ↑ 0.14	78.79 ↑ 0.18	69.71 ↓ 0.22	76.59 ↑ 0.22	79.35 ↑ 0.07	77.37 ↑ 1.49	71.18 ↑ 2.14	73.91 ↑ 0.57
Pretended CoT + CP-NR (***Ours***)	LLaMA2-13B	64.08 ↓ 0.19	79.00 ↑ 0.39	69.61 ↓ 0.32	76.90 ↑ 0.53	79.43 ↑ 0.15	77.23 ↑ 1.35	70.60 ↑ 1.56	73.84 ↑ 0.50
Pretended CoT	LLaMA3.1-8B	61.71	81.29	69.48	77.88	78.92	76.31	72.92	74.07
Pretended CoT + CP-NS (***Ours***)	LLaMA3.1-8B	63.25 ↑ 1.54	82.55 ↑ 1.26	70.73 ↑ 1.25	79.16 ↑ 1.28	80.06 ↑ 1.14	77.52 ↑ 1.21	73.28 ↑ 0.36	75.22 ↑ 1.15
Pretended CoT + CP-NR (***Ours***)	LLaMA3.1-8B	63.74 ↑ 2.03	82.46 ↑ 1.17	70.59 ↑ 1.11	78.92 ↑ 1.04	80.10 ↑ 1.18	77.45 ↑ 1.14	73.35 ↑ 0.43	75.23 ↑ 1.16

Table 2: Results on STS tasks (Spearman correlation scaled by 100x) using different backbones. Since Pretended CoT generalizes better across different LLMs, we use it for the experiment.

Method	w/o CP	w/ CP
PromptEOL	27 (1×)	31 (1.15×)
Pretended CoT	27 (1×)	31 (1.15×)
Knowledge	31 (1.15×)	37 (1.37×)
CK	54 (2×)	60 (2.22×)

Table 3: The number of layers in forward propagation.

through prompts. We additionally create a multi-prompt baseline **CK**, using the average of the embeddings extracted by **CoT** and **Knowledge** as the sentence embedding. The detailed prompts for **PromptEOL**, **CoT**, and **Knowledge** are shown in Appendix A.

5.3 Results

The results of our method on the STS benchmark are shown in Table 1. Our method shows improvement in 48 out of 56 cases, surpassing previous methods in average performance. This indicates that our method can effectively steer the existing prompt-based methods to focus more on semantics. Compared to the norm recovering strategy, the norm scaling strategy typically achieves better performance. This suggests that maintaining the consistency of the norm before and after the intervention is not essential for extracting sentence embeddings.

Our method achieves the greatest improvement on PromptEOL. This is because PromptEOL is the simplest compared to the other prompts, which limits its semantic encoding ability and makes it more necessary to refine it with auxiliary prompts. Our method can narrow the gap between different prompts, to some extent avoiding the variance caused by different prompts.

Finally, our method can improve the performance of CK, which demonstrates that our approach is still effective for averaging the embeddings of multiple prompts. This indicates that averaging the embeddings of multiple prompts still contains non-essential information. In addition, our method combined with CK achieves improvements on all datasets. This indicates that combining multiple prompts makes the performance improvement more robust.

5.4 Effects of Different Backbones

Table 2 shows the performance across various model backbones, including LLaMA2-7B (Touvron et al., 2023), LLaMA2-13B (Touvron et al., 2023), and LLaMA3.1-8B (Dubey et al., 2024).

The results demonstrate that our method can achieve performance improvements across various LLMs, highlighting its generalizability. Additionally, similar to the previous findings (Lei et al., 2024), LLaMA2-13B and LLaMA3.1-8B do not achieve better performance than LLaMA2-7B. This may be because different LLMs require different prompts to achieve optimal performance.

5.5 Analysis of the Number of Forward Propagation Layers

We further report the number of layers of forward propagation to estimate the time overhead in Table 3. The overhead of introducing auxiliary prompts is minimal, as they only need to propagate to the lower layers of the LLM, such as the 5th or 7th layer. The multi-prompt method CK often has a significant time overhead because each normal prompt needs to propagate to the higher layers of the LLM, whereas our auxiliary prompt only needs to propagate once to affect all the normal prompts, which further reduces the time overhead of our method.

5.6 Model Analysis

We further analyze the effects of auxiliary prompt, intervention position, scaling factor, and output layers.

Effects of Auxiliary Prompt We further explore the effects of auxiliary prompts using Knowledge +

Prompt	Knowledge+CP-NS
The irrelevant information of this sentence: "[TEXT]" means in one word:"	82.26 ↑ 0.43
The redundant information of this sentence: "[TEXT]" means in one word:"	82.41 ↑ 0.58
The background of this sentence: "[TEXT]" means in one word:"	82.12 ↑ 0.29
The descriptive term of this sentence: "[TEXT]" means in one word:"	82.48 ↑ 0.65
The sentence: "[TEXT]" reflects the sentiment in one word:"	77.17 ↓ 4.66
The sentence: "[TEXT]" highlights the primary entity or relation in one word:"	81.62 ↓ 0.21

Table 4: Effects of auxiliary prompt on the STS-B development set.

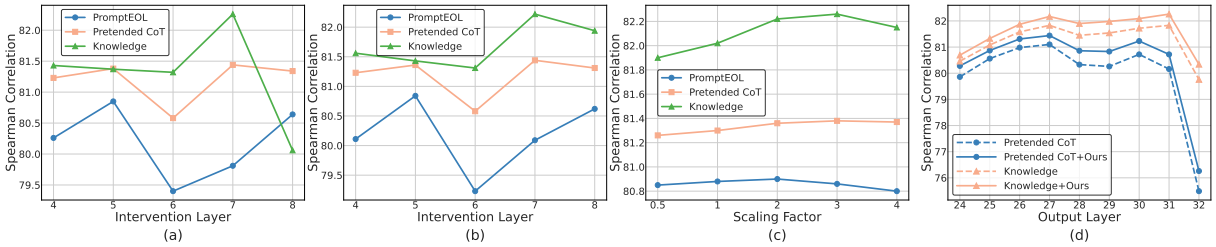

Figure 3: Effects of intervention layer, scaling factor, and output layer. (a) Effects of the intervention layer on norm scaling. (b) Effects of the intervention layer on norm recovering. (c) Effects of the scaling factor on norm scaling. (d) The effects of the output layer on Knowledge+CP-NS.

Position	Knowledge+CP-NS
Head	82.61 ↑ 0.78
FFN	81.93 ↑ 0.10
Hidden	82.53 ↑ 0.70

Table 5: Effects of intervention position on the STS-B development set. FFN denotes the output of FFN block, and Hidden denotes the output of the Transformer layer.

NP-NS on the STS-B development set in Table 4.

Using the first five semantically relevant prompts as auxiliary prompts often improves performance. This suggests that our method is not sensitive to auxiliary prompts. However, when we use other prompts that focus on sentiment and entities, they often do not improve performance. This is because these prompts do not focus on non-essential information, but instead focus on certain attributes of the sentence.

Effects of the Intervention Position We further explore the effects of intervention position and layer. Intervening in all three positions improves performance, indicating the effectiveness of the intervention, as shown in Table 5. Among them, the intervention on the multi-head attention achieves the best performance. This may be because the information interaction between tokens primarily occurs in the multi-head attention layer (Elhage et al., 2021).

The optimal intervention layer for PromptEOL is the 5th layer, while Pretended CoT and Knowledge are the 7th layer, as shown in Figure 3(a) and (b). In addition, the optimal intervention layer for both NS and NR is consistent under each prompt.

Effects of Scaling Factor We investigate the effects of the scaling factor in LLaMA2-7B using three prompts, as shown in Figure 3(c). The optimal scaling factor for PromptEOL is 2, while Pretended CoT and Knowledge are 3. As the scaling factor increases, all three prompts show an initial increase followed by a decrease. When the factor is 2 or 3, NS can achieve good performance across all three prompts.

Effects of Output Layers We investigate the effects of output layers in LLaMA2-7B using Pretended CoT and Knowledge prompt, as shown in Figure 3(d). CP-NS consistently improves both Pretended CoT and Knowledge across all layers, with more significant gains in the deeper layers. This suggests that our method can enhance the embeddings of all middle and later layers in the LLM, rather than just a single layer. Similar to previous findings (Li and Li, 2024; Lei et al., 2024), the sentence embedding of the last layer is not optimal for the STS tasks. Pretended CoT achieves optimal performance at the 27th layer, and Knowledge is at the 31st layer. This variation indicates that the optimal output layer varies for different prompts.

5.7 Transfer Learning Tasks

We further evaluate the performance of our method on transfer learning tasks, utilizing the standard

Method	Params	MR	CR	SUBJ	MPQA	SST2	TREC	MRPC	Avg.
Fine-tuning on supervised datasets									
SimCSE-RoBERTa	123M	84.92	92.00	94.11	89.82	91.27	88.80	75.65	88.08
ST5-Enc	4.8B	90.83	94.44	96.33	91.68	94.84	95.40	77.91	91.63
Without fine-tuning									
PromptEOL	7B	90.63	92.87	96.32	91.19	95.00	95.40	75.19	90.94
PromptEOL + CP-NS (***Ours***)	7B	90.49 ↓ 0.14	93.11 ↑ 0.24	96.97 ↑ 0.65	91.10 ↓ 0.09	95.94 ↑ 0.94	97.00 ↑ 1.60	77.51 ↑ 2.32	91.73 ↑ 0.79
PromptEOL + CP-NR (***Ours***)	7B	90.33 ↓ 0.30	92.64 ↓ 0.23	96.74 ↑ 0.42	91.11 ↓ 0.08	95.83 ↑ 0.83	96.40 ↑ 1.00	76.00 ↑ 0.81	91.29 ↑ 0.35

Table 6: Accuracy on transfer learning tasks using LLaMA2-7B.

Sentence	Method	Top-predicted Tokens and Probability
It is the first named storm to develop in the Caribbean in December.	Knowledge	It (0.0747), it (0.0233), ir (0.0231), I (0.0221) Dec (0.0219), It (0.0211), The (0.0192), What (0.0140)
	Knowledge+CP-NS	Dec (0.1092), St (0.0566), H (0.0537), It (0.0460) First (0.0391), Car (0.0360), first (0.0289), The (0.0289)
The rule - When in doubt throw it out!	Knowledge	Don (0.0401), Throw (0.0355), Do (0.0337), When (0.0303) The (0.0297), throw (0.0206), D (0.0181), If (0.0140)
	Knowledge+CP-NS	Throw (0.2071), throw (0.1414), Th (0.0466), D (0.0403) Th (0.0341), th (0.0226), TH (0.0185), Don (0.0177)

Table 7: Top-8 tokens predicted by different methods using LLaMA2-7B.

transfer learning tasks provided by SentEval, in line with prior works (Gao et al., 2021; Lei et al., 2024). These tasks include MR (Pang and Lee, 2005), CR (Hu and Liu, 2004), SUBJ (Pang and Lee, 2004), MPQA (Wiebe et al., 2005), SST-2 (Socher et al., 2013), TREC (Voorhees and Tice, 2000), and MRPC (Dolan and Brockett, 2005). For each task, we use the sentence embeddings generated by our method as features to train logistic regression classifiers. We additionally include two supervised contrastive trained models (SimCSE and ST5-Enc) for reference. Notably, ST5-Enc, with 4.8 billion parameters, is extensively trained on natural language inference (NLI) data and two billion question-answer pairs.

The results of our method on the transfer learning tasks are shown in Table 6. Our method both outperforms previous methods in average performance, and the norm scaling strategy outperforms the supervised method ST5-Enc. This further highlights the superiority of our method, which can outperform a 4.8B model trained with supervised contrastive learning, without requiring any additional training. On relatively simple sentiment classification datasets (such as MR, CR, MPQA, and SST2), the improvement is relatively limited. However, on more challenging tasks (such as SUBJ, TREC, and MRPC), the improvement is more significant. This is because different tasks require different embeddings. Complex tasks require embeddings with stronger semantic understanding capabilities, while for simpler sentiment classification, embeddings extracted from PromptEOL can perform well.

Additionally, the intervention layer is the 7th layer, and the length of norm scaling is 4. This illustrates the generalizability of the intervention layer, which can avoid extensive hyperparameter search.

5.8 Case Study

We further show the top-8 tokens predicted by different methods in Table 7. The example illustrates that Knowledge creates sentence embeddings focusing on stop-word tokens (such as it, I, the, what, Do, When), which convey non-essential information. In contrast, our method decodes the essential tokens of the sentence, such as Dec, St, First, Car, and Throw. This further intuitively demonstrates the effectiveness of our method.

6 Conclusion

In this paper, we introduce Contrastive Prompting, a plug-and-play inference-time steering method, to extract high-quality sentence embeddings from LLMs without the need for training or additional data. CP additionally introduces an auxiliary prompt to contrast with the normal prompt, steering it to encode the core semantics of the sentence,

rather than non-essential information. Extensive experiments show that our method can effectively elicit sentence embeddings across a range of diverse LLMs with varying sizes on both STS and transfer learning tasks.

Limitations

Firstly, our method preliminarily attempts to use auxiliary prompts to refine the representation of normal prompts. It is worth further exploring how to generate the optimal auxiliary prompt for each normal prompt. Secondly, the NR strategy needs to search for a hyperparameter (i.e., intervention layer), and the NS strategy needs to search for two hyperparameters (i.e., intervention layer and scaling factor) to elicit better sentence embeddings. Fortunately, in most cases, intervening at the 5th or 7th layer can achieve good performance. Finally, we conduct experiments only on the English dataset, and we plan to explore this in more languages in the future.

Acknowledgments

We would like to thank the anonymous reviewers for their insightful comments. This work is supported by the National Natural Science Foundation of China under Grants Nos. 62441225, 61972192, 62172208, 61906085. This work is partially supported by Collaborative Innovation Center of Novel Software Technology and Industrialization. This work is supported by the Fundamental Research Funds for the Central Universities under Grant No. 14380001.

References

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel M. Cer, Mona T. 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@NAACL-HLT 2015*, pages 252–263. The Association for Computer Linguistics.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel M. Cer, Mona T. 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@COLING 2014*, pages 81–91. The Association for Computer Linguistics.

Eneko Agirre, Carmen Banea, Daniel M. Cer, Mona T. 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@NAACL-HLT 2016*, pages 497–511. The Association for Computer Linguistics.

Eneko Agirre, Daniel M. Cer, Mona T. Diab, and Aitor Gonzalez-Agirre. 2012. Semeval-2012 task 6: A pilot on semantic textual similarity. In *Proceedings of the 6th International Workshop on Semantic Evaluation, SemEval@NAACL-HLT 2012*, pages 385–393. The Association for Computer Linguistics.

Eneko Agirre, Daniel M. Cer, Mona T. Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. 2013. *sem 2013 shared task: Semantic textual similarity. In *Proceedings of the Second Joint Conference on Lexical and Computational Semantics, *SEM 2013*, pages 32–43. 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. *arXiv preprint arXiv:2404.05961*.

Daniel M. Cer, Mona T. Diab, Eneko Agirre, Iñigo Lopez-Gazpio, and Lucia Specia. 2017. Semeval-2017 task 1: Semantic textual similarity - multilingual and cross-lingual focused evaluation. *CoRR*, abs/1708.00055.

Sachin Chanchani and Ruihong Huang. 2023. Composition-contrastive learning for sentence embeddings. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15836–15848. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019*, pages 4171–4186.

Bill Dolan and Chris Brockett. 2005. Automatically constructing a corpus of sentential paraphrases. In *Third international workshop on paraphrasing (IWP2005)*.

Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Amy Yang, Angela Fan, et al. 2024. The llama 3 herd of models. *arXiv preprint arXiv:2407.21783*.

Nelson Elhage, Neel Nanda, Catherine Olsson, Tom Henighan, Nicholas Joseph, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, Tom Conerly, Nova DasSarma, Dawn Drain, Deep Ganguli, Zac Hatfield-Dodds, Danny Hernandez, Andy Jones,

Jackson Kernion, Liane Lovitt, Kamal Ndousse, Dario Amodei, Tom Brown, Jack Clark, Jared Kaplan, Sam McCandlish, and Chris Olah. 2021. A mathematical framework for transformer circuits. *Transformer Circuits Thread*. Https://transformer-circuits.pub/2021/framework/index.html.

Yuchen Fu, Zifeng Cheng, Zhiwei Jiang, Zhonghui Wang, Yafeng Yin, Zhengliang Li, and Qing Gu. 2024. Token prepending: A training-free approach for eliciting better sentence embeddings from llms. *CoRR*, abs/2412.11556.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. 2021. Simcse: Simple contrastive learning of sentence embeddings. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6894–6910.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 168–177.

Ting Jiang, Shaohan Huang, Zhongzhi Luan, Deqing Wang, and Fuzhen Zhuang. 2023. Scaling sentence embeddings with large language models. *arXiv 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. 2022. Prompt-bert: Improving BERT sentence embeddings with prompts. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing, EMNLP 2022*, pages 8826–8837.

Mingyu Jin, Qinkai Yu, Jingyuan Huang, Qingcheng Zeng, Zhenting Wang, Wenyue Hua, Haiyan Zhao, Kai Mei, Yanda Meng, Kaize Ding, Fan Yang, Mengnan Du, and Yongfeng Zhang. 2024. Exploring concept depth: How large language models acquire knowledge at different layers? *CoRR*, abs/2404.07066.

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. 2024. Nv-embed: Improved techniques for training llms as generalist embedding models. *CoRR*, abs/2405.17428.

Yibin Lei, Di Wu, Tianyi Zhou, Tao Shen, Yu Cao, Chongyang Tao, and Andrew Yates. 2024. Meta-task prompting elicits embedding from large language models. *arXiv preprint arXiv:2402.18458*.

Chak Tou Leong, Yi Cheng, Jiashuo Wang, Jian Wang, and Wenjie Li. 2023. Self-detoxifying language models via toxification reversal. In *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing, EMNLP 2023*, pages 4433–4449. Association for Computational Linguistics.

Kenneth Li, Oam Patel, Fernanda B. Viégas, Hanspeter Pfister, and Martin Wattenberg. 2023. Inference-time intervention: Eliciting truthful answers from a language model. In *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*.

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.

Xianming Li, Zongxi Li, Jing Li, Haoran Xie, and Qing Li. 2025. ESE: espresso sentence embeddings. In *The Thirteenth International Conference on Learning Representations, ICLR 2025*.

Tian Yu Liu, Matthew Trager, Alessandro Achille, Pramuditha Perera, Luca Zancato, and Stefano Soatto. 2024a. Meaning representations from trajectories in autoregressive models. In *The Twelfth International Conference on Learning Representations*.

Zhu Liu, Cunliang Kong, Ying Liu, and Maosong Sun. 2024b. Fantastic semantics and where to find them: Investigating which layers of generative llms reflect lexical semantics. *CoRR*, abs/2403.01509.

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 2014*, pages 216–223.

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

Niklas Muennighoff, Nouamane Tazi, Loïc Magne, and Nils Reimers. 2022. Mteb: Massive text embedding benchmark. *arXiv preprint arXiv:2210.07316*.

Jianmo Ni, Gustavo Hernandez Abrego, Noah Constant, Ji Ma, Keith Hall, Daniel Cer, and Yinfei Yang. 2022a. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874.

Jianmo Ni, Gustavo Hernández Ábrego, Noah Constant, Ji Ma, Keith B. Hall, Daniel Cer, and Yinfei Yang. 2022b. Sentence-t5: Scalable sentence encoders from pre-trained text-to-text models. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1864–1874.

Zhijie Nie, Zhangchi Feng, Mingxin Li, Cunwang Zhang, Yanzhao Zhang, Dingkun Long, and Richong Zhang. 2024. When text embedding meets large language model: A comprehensive survey. *CoRR*, abs/2412.09165.

Bo Pang and Lillian Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL-04)*, pages 271–278.

Bo Pang and Lillian Lee. 2005. Seeing stars: Exploiting class relationships for sentiment categorization with respect to rating scales. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*, pages 115–124.

Nina Rimsky, Nick Gabrieli, Julian Schulz, Meg Tong, Evan Hubinger, and Alexander Matt Turner. 2024. Steering llama 2 via contrastive activation addition. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2024*, pages 15504–15522. Association for Computational Linguistics.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D Manning, Andrew Y Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1631–1642.

Jacob Mitchell Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raghunathan. 2024. Repetition improves language model embeddings. *arXiv preprint arXiv:2402.15449*.

Hongjin Su, Weijia Shi, Jungo Kasai, Yizhong Wang, Yushi Hu, Mari Ostendorf, Wen-tau Yih, Noah A. Smith, Luke Zettlemoyer, and Tao Yu. 2023. One embedder, any task: Instruction-finetuned text embeddings. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 1102–1121. Association for Computational Linguistics.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, et al. 2023. Llama 2: Open foundation and fine-tuned chat models. *arXiv preprint arXiv:2307.09288*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017,*, pages 5998–6008.

Ellen M Voorhees and Dawn M Tice. 2000. Building a question answering test collection. In *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 200–207.

Janyce Wiebe, Theresa Wilson, and Claire Cardie. 2005. Annotating expressions of opinions and emotions in language. *Language resources and evaluation*, 39:165–210.

Bowen Zhang, Kehua Chang, and Chunping Li. 2024. Simple techniques for enhancing sentence embeddings in generative language models. *arXiv preprint arXiv:2404.03921*.

Shengyao Zhuang, Shuai Wang, Bevan Koopman, and Guido Zuccon. 2024. Starbucks: Improved training for 2d matryoshka embeddings. *CoRR*, abs/2410.13230.

Andy Zou, Long Phan, Sarah Chen, James Campbell, Phillip Guo, Richard Ren, Alexander Pan, Xuwang Yin, Mantas Mazeika, Ann-Kathrin Dombrowski, Shashwat Goel, Nathaniel Li, Michael J. Byun, Zifan Wang, Alex Mallen, Steven Basart, Sanmi Koyejo, Dawn Song, Matt Fredrikson, J. Zico Kolter, and Dan Hendrycks. 2023. Representation engineering: A top-down approach to AI transparency. *CoRR*, abs/2310.01405.

A Baselines

We report the detailed prompts for the baselines as follows:

> **PromptEOL:** This sentence: "[TEXT]" means in one word: "
> **Pretended CoT:** After thinking step by step, this sentence: "[TEXT]" means in one word: "
> **Knowledge:** The essence of a sentence is often captured by its main subjects and actions, while descriptive terms provide additional but less central details. With this in mind , this sentence: "[TEXT]" means in one word: "

B Performance on STS Tasks under In-Context Learning

In this section, we further explore whether CP can enhance sentence embedding under the in-context learning setting.

We observe that CP also improves performance under in-context learning settings in Table 8, demonstrating its generalizability. The gains are less pronounced compared to the zero-shot scenario, possibly because the additional context already guides the model to focus on the core semantics of the sentence.

C Effects of Hyperparameters across Domains

In this section, we further explore the effect of hyperparameters on the results across different domains, including the STS-12, STS-13, and STS-14 datasets.

The optimal intervention layer for Knowledge is the 7th layer on all three datasets in Table 9. The optimal scaling factors for the three datasets are 4, 4, and 2, respectively. Therefore, these two hyperparameters do not vary significantly across different domains.

D Multi-Task Evaluation

We further evaluate the CP method across classification task, clustering task, reranking task, and pair classification task from the MTEB benchmark (Muennighoff et al., 2022). Due to the large size of the MTEB dataset, we evaluate the effectiveness of our method on only a subset of the data.

Our method consistently improves performance across classification, pair classification, reranking, and clustering tasks, as demonstrated in Tables 10, 11, 12, and 13.

Method	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
PromptEOL+ICL† (Jiang et al., 2023)	70.65	84.51	75.01	83.51	82.00	81.12	76.77	79.08
PromptEOL+ICL + CP-NS (***Ours***)	71.44 ↑ 0.79	84.82 ↑ 0.31	75.42 ↑ 0.41	84.29 ↑ 0.78	82.51 ↑ 0.51	81.85 ↑ 0.73	75.30 ↓ 1.47	79.38 ↑ 0.30
PromptEOL+ICL + CP-NR (***Ours***)	70.61 ↓ 0.04	84.61 ↑ 0.10	75.24 ↑ 0.23	83.69 ↑ 0.18	82.16 ↑ 0.16	81.41 ↑ 0.29	76.72 ↓ 0.05	79.21 ↑ 0.13

Table 8: Results on STS tasks using OPT-6.7B as the backbone. †denotes the results from the original paper.

STS12	**Layer = 6**	**Layer = 7**	**Layer = 8**
α **= 1**	65.48	66.13	66.39
α **= 2**	65.44	66.74	66.61
α **= 3**	65.57	67.16	66.78
α **= 4**	65.57	**67.23**	66.85
α **= 5**	65.64	66.93	66.79
STS13	**Layer = 6**	**Layer = 7**	**Layer = 8**
α **= 1**	83.37	82.95	82.97
α **= 2**	83.33	83.24	83.01
α **= 3**	83.22	84.43	83.06
α **= 4**	83.08	**84.52**	83.14
α **= 5**	82.96	83.38	83.17
STS14	**Layer = 6**	**Layer = 7**	**Layer = 8**
α **= 1**	73.72	73.92	73.99
α **= 2**	73.61	**74.24**	73.92
α **= 3**	73.52	74.23	73.84
α **= 4**	73.44	74.09	74.81
α **= 5**	73.37	73.82	73.82

Table 9: Effects of hyperparameters across domains. Layer denotes the intervention layer.

Method	PromptEOL	PromptEOL+CP-NS
AmazonCounterfactual	70.83	73.75 ↑ 2.92
Banking77	78.94	81.54 ↑ 2.60
Emotion	48.35	50.96 ↑ 2.61
Average (3)	66.04	68.75 ↑ 2.71

Table 10: Accuracy on classification datasets using LLaMA2-7B.

Method	PromptEOL	PromptEOL+CP-NS
SprintDuplicateQuestions	43.02	48.60 ↑ 5.58
TwitterSemEval2015	65.61	68.55 ↑ 2.94
Average (2)	54.32	58.58 ↑ 4.26

Table 11: Accuracy on pair classification datasets using LLaMA2-7B.

Method	PromptEOL	PromptEOL+CP-NS
AskUbuntuDupQuestions	53.88	57.02 ↑ 3.14
SciDocsRR	71.38	77.94 ↑ 6.56
StackOverflowDupQuestions	40.63	43.04 ↑ 2.41
Average (3)	55.30	59.33 ↑ 4.03

Table 12: Average precision on reranking datasets using LLaMA2-7B.

Method	PromptEOL	PromptEOL+CP-NS
TwentyNewsgroupsClustering	27.61	35.53 ↑ 7.92

Table 13: V-measure on clustering datasets using LLaMA2-7B.