RETHINKING LLM-BASED EMBEDDINGS: DECOMPOSE AND RECOMBINE FOR BETTER TEXT EMBEDDINGS

Anonymous authors
Paper under double-blind review

ABSTRACT

Extracting sentence embeddings from large language models (LLMs) has attracted increasing attention without requiring LLM fine-tuning. However, existing methods mainly rely on coarse-grained selection of internal representations. In this work, we decompose hidden states into layer-wise attention and feed-forward network (FFN) components and reveal substantial heterogeneity in their contributions to sentence-level semantics. Based on this observation, we propose **DeRe** (**De**compose and **Re**combine), which assigns coefficients to individual components and recombines them into the final sentence embedding. Since Spearman correlation is non-differentiable, we employ Covariance Matrix Adaptation Evolution Strategy (CMA-ES) to jointly search the component coefficients based on validation performance. DeRe operates on cached representations without additional LLM forward passes or parameter updates. Extensive experiments on STS, MTEB, and multiple LLM backbones demonstrate consistent improvements across different prompts, tasks, and models.

1 INTRODUCTION

Sentence embeddings play a fundamental role in the NLP community, supporting a wide range of applications, from intrinsic tasks such as Semantic Textual Similarity (STS) (Agirre et al., 2012) to downstream tasks, including information retrieval, text classification, and clustering. Recently, inspired by the strong capabilities of LLMs, several studies have explored directly extracting sentence embeddings from LLMs *without fine-tuning the LLMs*. Compared to fine-tuning approaches (Lee et al., 2025), this setting avoids the need for large-scale data collection and the high cost of fine-tuning, enabling strong performance in low-resource and domain-specific scenarios (Tang & Yang, 2024) while preserving the general capabilities of LLMs (Li & Zhou, 2025).

Existing methods (Jiang et al., 2023b; Lei et al., 2024; Thirukovalluru & Dhingra, 2025; Springer et al., 2025) often focus on prompt engineering to elicit embeddings from LLMs. These methods incorporate the "one word restriction" in the prompt to encourage LLMs to *compress* the full sentence semantics into the last token, whose representation is used as the sentence embedding. For example, PromptEOL (Jiang et al., 2023b) proposes a simple yet effective prompt: *This sentence: "*`[TEXT]`*" means in one word: "*, where `[TEXT]` serves as a placeholder for the sentence. Subsequently, several plug-and-play methods (Fu et al., 2025; Cheng et al., 2025) have been proposed, and Zhang et al. (2026) further investigates the embedding quality of outputs from different modules within the Transformer and manually selects and sums the outputs of attention modules from several layers to construct sentence embeddings.

Despite their success, existing methods still rely on coarse-grained selection of internal representations in LLMs. For example, some studies (Lei et al., 2024; Fu et al., 2025; Cheng et al., 2025) find that intermediate-layer representations contain richer semantic information, while Aligned-WVA (Zhang et al., 2026) further manually selects the outputs of a subset of attention layers. However, these methods merely select internal representations from LLMs, without explicitly considering how the constituent components within these representations contribute to sentence-level semantics. This motivates a finer-grained analysis of how individual internal components contribute to sentence representations.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

To this end, we revisit sentence representations through the lens of the residual stream. As shown in Figure 1, each Transformer layer updates the residual stream by adding the outputs of the multi-head self-attention (MHSA) and feed-forward network (FFN) modules. Accordingly, the hidden states can be decomposed into layer-wise attention and FFN components, allowing us to analyze their contributions to sentence-level semantics separately. The results reveal substantial heterogeneity across layers and modules: attention outputs are generally more semantically informative, while uniformly combining attention and FFN components remains suboptimal. These observations motivate a finer-grained recombination of the decomposed components.

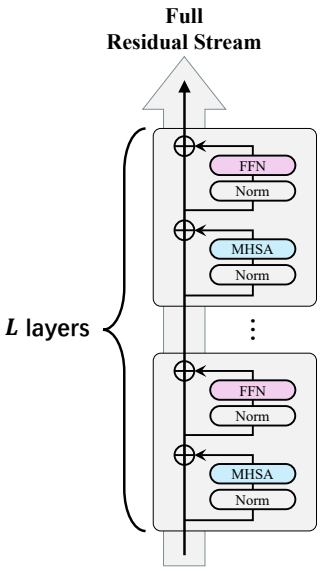


Figure 1: Illustration of the residual stream in LLMs, which can be decomposed into MHSA and FFN outputs.

Based on these observations, we propose **DeRe** (**De**compose and **Re**combine), which constructs sentence embeddings by fine-grained recombination of decomposed internal components. Specifically, DeRe assigns a coefficient to each attention and FFN component, allowing their contributions to the final representation to be flexibly adjusted. Since our optimization objective, i.e., Spearman correlation, is non-differentiable, we employ Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen & Ostermeier, 1996) to jointly search these coefficients based on validation performance. By adapting the covariance of the search distribution, CMA-ES captures dependencies among coefficients and enables coordinated optimization across layers and modules.

In summary, our main contributions are as follows:

- We revisit sentence embedding extraction from LLMs from a fine-grained residual decomposition perspective. By decomposing hidden states into layer-wise attention and FFN components, we reveal substantial heterogeneity in their contributions to sentence-level semantics.
- We propose **DeRe** (**De**compose and **Re**combine), which formulates sentence embedding extraction as a fine-grained component recombination problem and jointly optimizes the contributions of attention and FFN components with CMA-ES. DeRe can be seamlessly integrated with existing prompt-based baselines and operates on cached representations without additional LLM forward passes or parameter updates.
- We conduct extensive experiments on STS benchmarks, MTEB, and multiple LLM backbones. The results demonstrate consistent improvements across different prompts, tasks, and models, showing strong cross-task and cross-model generalization.

2 Related Work

Training-Based Sentence Embedding Models Sentence embedding models aim to represent a sentence as a fixed-size dense vector and have a wide range of applications. Existing methods often use contrastive learning to fine-tune language models for enhancing sentence embeddings. Early works (Gao et al., 2021; Ni et al., 2022; Chanchani & Huang, 2023) usually fine-tuned small pre-trained models, such as BERT (Devlin et al., 2019), to improve embedding quality. A representative work is SimSCE (Gao et al., 2021), which leverages dropout to generate positive pairs for contrastive learning and fine-tune the model.

In recent years, decoder-only LLMs have achieved remarkable success in the field of sentence embeddings (Zhang et al., 2025; Lee et al., 2025), surpassing traditional encoder-only and encoder-decoder models. For example, E5 (Wang et al., 2023) fine-tunes the Mistral-7B using synthetic data generated by the LLM and a standard contrastive learning loss. Qwen3-Embedding (Zhang et al., 2025) employs weakly supervised training on large-scale synthetic data, supervised fine-tuning on high-quality synthetic data, and model merging to enhance performance. However, these methods require large amounts of data and expensive fine-tuning, and they tend to degrade general LLMs into mere embedding models.

Extracting Sentence Embedding from LLMs In contrast to fine-tuning approaches, some recent studies have explored extracting high-quality sentence embeddings from LLMs. Early work focuses on prompt engineering, where the pioneering PromptEOL (Jiang et al., 2023b) introduces a semantic compression prompt that compresses sentence embeddings into the representation of a single token. Following this line of work, MetaEOL (Lei et al., 2024) designs meta-task prompts to guide LLMs to consider multiple perspectives when generating embeddings. GenEOL (Thirukovalluru & Dhingra, 2025) generates multiple meaning-preserving variations of a sentence and averages their embeddings. Pretended CoT (Zhang et al., 2024) uses chain-of-thought prompting to guide the model toward generating higher-quality embeddings. Knowledge Enhancement (Zhang et al., 2024) seeks to explicitly infuse the model with human insights into text summarization through tailored prompts. Echo (Springer et al., 2025) duplicates the text twice in the prompt and obtains the sentence embedding by mean pooling over the tokens in the second occurrence, allowing the latter text to capture the full semantics. Afterwards, some training-free techniques have been proposed to enhance sentence embeddings. TP (Fu et al., 2025) enhances embeddings by prepending the sentence embedding token in early layers. CP (Cheng et al., 2025) introduces contrastive prompts and employs activation steering techniques to improve embedding. Besides, AlignedWVA (Zhang et al., 2026) finds that the outputs of attention modules contain stronger semantic information and manually selects and sums the outputs of multiple attention layers as the final embedding based on validation performance across several tasks. Different from existing methods, we decompose hidden states into layer-wise attention and FFN components and jointly optimize their contributions through fine-grained recombination.

3 METHODOLOGY

In this section, we first introduce sentence embedding extraction from LLMs and present a residual stream decomposition that separates the outputs of attention and FFN modules. We then analyze the semantic contributions of these components and show that their effectiveness varies substantially across layers and modules. Motivated by these observations, we propose to recombine the decomposed components with individually coefficients. Figure 2 provides an overview of our framework.

3.1 PRELIMINARY

Previous work mainly focused on eliciting sentence embeddings from LLMs through prompt engineering. For example, PromptEOL (Jiang et al., 2023b) employs a *semantic compression* prompt that guides LLMs to compress the meaning of a sentence into the final token:

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

where `[TEXT]` is a placeholder for the input sentence. The residual stream of the final token " encodes the compressed semantics of the entire sentence and is used for next-token prediction, thereby being regarded as the Sentence Embedding Token. The phrase "in one word" encourages the LLM to condense the semantic meaning of the sentence into the hidden state of the next token. Since the hidden state of the last token (i.e., ") is used to predict the next token, we can directly use the representation of the last token as the sentence embedding without generating any tokens.

3.2 A RESIDUAL DECOMPOSITION VIEW OF SENTENCE EMBEDDINGS

Previous studies suggest that the multi-layer Transformer architecture can be interpreted as a process of progressively reading from and writing to the residual stream (Elhage et al., 2021; Mickus et al., 2022). As shown in Figure 1, the residual stream at each layer is updated by adding the outputs of the attention and FFN modules:

$$\mathbf{h}^{\ell} = \mathbf{h}^{\ell-1} + \mathbf{a}^{\ell} + \mathbf{m}^{\ell}, \tag{1}$$

where $\mathbf{h}^{\ell}$ denotes the residual stream at layer ℓ, $\mathbf{h}^{\ell-1}$ is the residual stream from the previous layer, and $\mathbf{a}^{\ell}$ and $\mathbf{m}^{\ell}$ denote the outputs of the attention and FFN modules at layer ℓ, respectively.

Applying this update recursively over all L Transformer layers, the final residual stream can be written as

$$\mathbf{h}^{L} = \mathbf{h}^{0} + \sum_{\ell=1}^{L} \mathbf{a}^{\ell} + \sum_{\ell=1}^{L} \mathbf{m}^{\ell}, \tag{2}$$

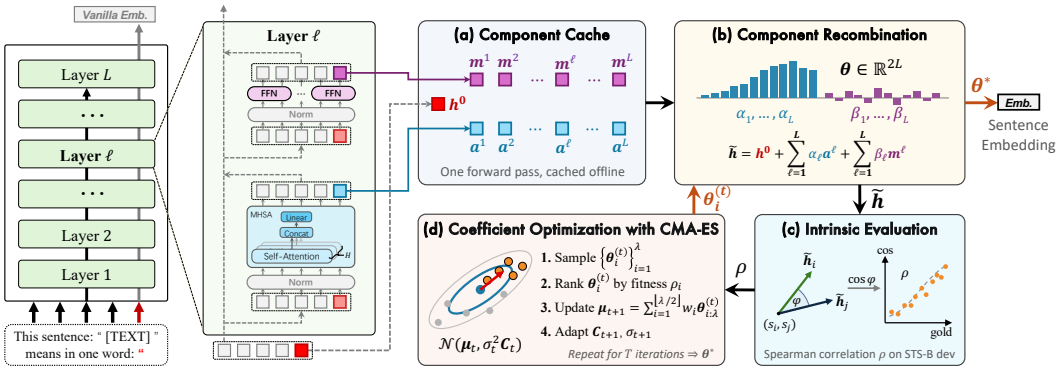
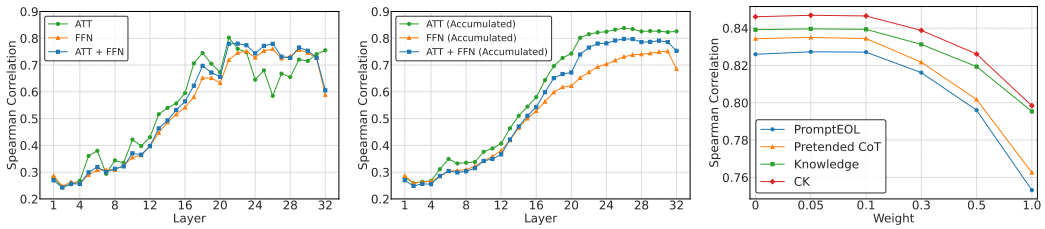


Figure 2: Overview of our framework for sentence embedding extraction via residual decomposition, component recombination, and coefficient optimization.



(a) (b) (c)

Figure 3: Analysis of the decomposed residual stream on the STS-B validation set using LLaMA2-7B. (a) Layer-wise Spearman correlation of attention (ATT), FFN, and their sum under PromptEOL; (b) Spearman correlation of their accumulated representations across layers; (c) Effect of scaling the accumulated FFN component in combination with accumulated ATT across 4 prompting strategies.

where $\mathbf{h}^0$ denotes the initial token representation. This decomposition shows that the final hidden representation is formed by accumulating the outputs of the attention and FFN modules across all Transformer layers. Notably, this process does not modify the forward pass of the LLM.

3.3 Analysis of Component Contributions

To understand how different internal components contribute to sentence embeddings, we analyze the attention and FFN outputs at both the individual-layer and accumulated levels.

Observation 1: Layer-Wise Heterogeneity. We first evaluate the embedding quality of the attention output (ATT), FFN output (FFN), and their sum (ATT+FFN) at each Transformer layer. As shown in Figure 3(a), their performance varies substantially across layers, and no single type of component consistently dominates at every layer. This suggests that different layers and modules contribute unequally to sentence-level semantics.

Observation 2: Cross-Layer Accumulation. We further progressively accumulate ATT, FFN, and ATT+FFN outputs from the bottom to higher layers. Figure 3(b) shows that cross-layer accumulation generally improves embedding quality, indicating that useful semantic information is distributed across multiple layers rather than concentrated in a single layer. Meanwhile, the performance gaps among different accumulated components remain evident, further suggesting that their contributions should not be treated uniformly.

Observation 3: Limitations of Uniform Recombination. We finally examine whether the accumulated attention and FFN components can be effectively combined using a shared scaling weight. Since Accumulated FFN performs worse than Accumulated ATT, we progressively reduce its contribution by constructing the embedding as $\mathrm{ATT(Accumulated)} + \alpha\,\mathrm{FFN(Accumulated)}$ with different values of α. As shown in Figure 3(c), simply down-weighting the FFN component does

Algorithm 1 Coefficient Optimization with CMA-ES

Require: Cached components, STS-B validation set, initial CMA-ES parameters
Ensure: Optimized coefficient vector $\boldsymbol{\theta}^*$
for $t = 0, \ldots, T-1$ **do**
  Sample $\{\boldsymbol{\theta}_i^{(t)}\}_{i=1}^{\lambda} \sim \mathcal{N}(\mu_t, \sigma_t^2 \mathbf{C}_t)$
  for $i = 1, \ldots, \lambda$ **do**
    Compute sentence embeddings by recombining components using $\boldsymbol{\theta}_i^{(t)}$
    Compute cosine similarities for sentence pairs
    Compute Spearman fitness on the STS-B validation set
  end for
  Rank candidates by fitness
  Update μ_{t+1}, $\mathbf{C}_{t+1}$, and σ_{t+1} using CMA-ES
end for
return the best coefficient vector $\boldsymbol{\theta}^*$

not consistently improve performance and can still lead to substantial degradation. This result indicates that uniformly scaling an entire internal component is insufficient for effective recombination, motivating a finer-grained weighting of individual components.

Overall, these observations reveal substantial heterogeneity among internal components across both layers and modules. Rather than manually selecting components or combining them with uniform weights, a more flexible approach is to determine the contribution of each component individually. This motivates the coefficient optimization method introduced in the following section.

3.4 Fine-Grained Component Recombination with CMA-ES

After decomposing the hidden states into layer-wise attention and FFN components, we view sentence embedding extraction as a fine-grained component recombination problem. Since these components jointly constitute the residual stream, their effects on the final representation are not independent and should be coordinated during recombination. We therefore assign a coefficient to each component and jointly optimize these coefficients to construct the final sentence representation. Let $\boldsymbol{\theta} = [\alpha_1, \ldots, \alpha_L, \beta_1, \ldots, \beta_L] \in \mathbb{R}^{2L}$ denote the recombination coefficients, where α_ℓ and β_ℓ control the contributions of the attention and FFN components at layer ℓ, respectively. Given $\boldsymbol{\theta}$, the resulting representation is constructed as

$$\tilde{\mathbf{h}} = \mathbf{h}^0 + \sum_{\ell=1}^{L} \alpha_\ell \mathbf{a}^\ell + \sum_{\ell=1}^{L} \beta_\ell \mathbf{m}^\ell. \tag{3}$$

Since Spearman correlation, which we use as the optimization objective, is non-differentiable, we employ the Covariance Matrix Adaptation Evolution Strategy (CMA-ES) (Hansen & Ostermeier, 1996), a derivative-free optimization algorithm for continuous search spaces. CMA-ES iteratively samples a population of candidate solutions from a multivariate Gaussian distribution and updates the distribution based on the fitness of the candidates. The overall algorithmic procedure is shown in Algorithm 1.

At iteration t, CMA-ES maintains a multivariate Gaussian search distribution $\boldsymbol{\theta} \sim \mathcal{N}\left(\mu_t, \sigma_t^2 \mathbf{C}_t\right)$, where μ_t denotes the search center, σ_t controls the global exploration scale, and $\mathbf{C}_t$ models dependencies among different coefficients. By adapting $\mathbf{C}_t$, CMA-ES can capture dependencies among attention and FFN components across different layers.

A population of λ candidate coefficient vectors $\{\boldsymbol{\theta}_i^{(t)}\}_{i=1}^{\lambda}$ is sampled from this Gaussian distribution. For each candidate, we first recombine the components according to $\boldsymbol{\theta}_i^{(t)}$, compute the resulting embeddings and cosine similarities for sentence pairs on the STS-B validation set, and use the Spearman correlation with the ground-truth similarity scores as the fitness. The candidates are then ranked according to their fitness, and the search center is updated by a weighted combination of the top half of the population:

$$\mu_{t+1} = \sum_{i=1}^{\lfloor \lambda/2 \rfloor} w_i \boldsymbol{\theta}_{i:\lambda}^{(t)}, \tag{4}$$

where $\boldsymbol{\theta}_{i:\lambda}^{(t)}$ denotes the i-th best candidate and w_i is its normalized recombination weight. Based on the selected candidates, CMA-ES updates the covariance matrix $\mathbf{C}_{t+1}$ to increase the search variance along directions that consistently yield better fitness, thereby capturing correlations among different coefficients. Meanwhile, the global step size σ_{t+1} is adaptively increased or decreased according to the consistency of recent search steps, allowing the search to balance exploration and refinement as optimization proceeds.

Coefficient optimization does not require additional LLM forward passes. All decomposed representations are computed and cached in advance, and each CMA-ES evaluation only involves linear recombination, cosine similarity computation, and validation scoring. Since the number of optimized coefficients is small, the search is highly efficient and takes only about five minutes, while keeping the underlying LLM completely frozen. Notably, caching the representations of all 1,500 examples in the STS-B development set requires only about 3 GB of memory.

4 Experiments

4.1 Datasets and Experimental Setting

We use seven Semantic Textual Similarity (STS) datasets to evaluate the performance, 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 the STS datasets is annotated with a pairwise semantic similarity score from 0 to 5. We use Spearman correlation as the evaluation metric, which measures the rank correlation between the predicted similarity scores and annotated similarity scores using a monotonic function. We use cosine similarity to compute the predicted similarity scores.

We initialize the Gaussian distribution with an all-ones vector, corresponding to the model's original hidden states. For the population size λ and the maximum number of iterations, we use the default values of 14 and 800, respectively. We use the STS-B development set to search for the initial step size σ over $\{0.1, 0.25, 0.5, 1, 2\}$.

4.2 Baselines

We compare our method with two categories of baselines. The first category of methods requires unsupervised data for contrastive training. **LLM2Vec** (BehnamGhader et al., 2024) includes three stages to fine-tune LLMs: bidirectional attention enabling, masked token prediction, and unsupervised contrastive training. The second category of methods does not require contrastive training. **ST5-End avg** (Ni et al., 2022) and **LLaMA2 avg** (Touvron et al., 2023) average token embeddings to obtain sentence embeddings using different backbones. **LLaMA2 echo** (Springer et al., 2025) repeats the input twice in context and extracts embeddings from the second occurrence. **PromptEOL** (Jiang et al., 2023b) first proposes using prompts to extract sentence embeddings from LLMs. **MetaEOL** (Lei et al., 2024) leverages a set of diverse (i.e., 8) meta-task prompts to capture semantically rich sentence embeddings. **GenEOL** (Thirukovalluru & Dhingra, 2025) utilizes LLMs to generate 16 semantically equivalent sentence variants and averages their representations to obtain a more robust sentence embedding. **Pretended CoT** (Zhang et al., 2024) and **Knowledge** (Zhang et al., 2024) place a chain-of-thought rationale and a statement containing human experience in text summarization before the **PromptEOL**, respectively. **CK** (Cheng et al., 2025) is a multi-prompt method that averages the embeddings obtained from **Pretended CoT** and **Knowledge**. **CP** (Cheng et al., 2025) introduces auxiliary prompts during inference and performs contrastive activation steering to enhance the sentence embedding. **TP** (Fu et al., 2025) is a plug-and-play technique that prepends a sentence embedding token, enabling all tokens to capture the overall sentence semantics. The detailed prompts for **LLaMA2 echo**, **PromptEOL**, **Pretended CoT**, and **Knowledge** are shown in the Appendix A.

4.3 Main Results

The results on the STS benchmarks are shown in Table 1. Our method consistently improves all four prompt-based embedding methods across all seven datasets. In particular, it improves the average performance of PromptEOL, Pretended CoT, Knowledge, and CK by 9.42, 3.97, 3.40, and 3.31 points, respectively. These consistent gains demonstrate that fine-grained component recombina-

Table 1: Performance comparison on STS tasks (Spearman correlation scaled by 100x) using LLaMA2-7B.

Method	Para.	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
Unsupervised Contrastive Training:									
LLM2Vec-LLaMA2	7B	65.39	79.26	72.98	82.72	81.02	78.32	71.77	75.92
LLM2Vec-Mistral	7B	67.65	83.90	76.97	83.80	81.91	80.42	75.55	78.60
Without LLM Fine-tuning									
STS-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
GenEOL	7B	70.78	83.28	77.75	82.10	80.45	79.83	78.71	78.99
PromptEOL	7B	58.81	77.01	66.34	73.22	73.56	71.66	69.64	70.03
PromptEOL + CP-NS	7B	63.34	82.15	71.73	79.68	77.23	78.71	74.04	75.27
PromptEOL + TP	7B	66.90	83.12	74.31	79.87	80.03	80.67	75.40	77.19
PromptEOL + DeRe (Ours)	7B	70.03 ↑ 11.22	85.81 ↑ 8.80	76.73 ↑ 10.39	83.82 ↑ 10.60	81.66 ↑ 8.10	81.70 ↑ 10.04	76.37 ↑ 6.73	79.45 ↑ 9.42
Pretended CoT	7B	67.45	83.89	74.14	79.47	80.76	78.95	73.33	76.86
Pretended CoT + CP-NS	7B	67.79	83.66	74.52	81.10	80.70	80.39	74.01	77.45
Pretended CoT + TP	7B	68.52	83.44	75.23	79.36	81.33	80.37	74.51	77.54
Pretended CoT + DeRe (Ours)	7B	71.93 ↑ 4.48	86.44 ↑ 2.55	78.57 ↑ 4.43	84.94 ↑ 5.47	83.93 ↑ 3.17	83.41 ↑ 4.46	76.58 ↑ 3.25	80.83 ↑ 3.97
Knowledge	7B	65.60	82.82	74.48	80.75	80.13	80.34	75.89	77.14
Knowledge + CP-NS	7B	67.16	83.43	74.23	81.29	80.03	80.80	75.97	77.56
Knowledge + TP	7B	66.03	83.43	74.50	80.94	81.28	80.45	76.13	77.54
Knowledge + DeRe (Ours)	7B	69.41 ↑ 3.81	86.72 ↑ 3.90	78.23 ↑ 3.75	84.58 ↑ 3.83	83.05 ↑ 2.92	83.85 ↑ 3.51	77.97 ↑ 2.08	80.54 ↑ 3.40
CK	7B	67.11	84.03	75.07	82.42	80.91	81.84	76.24	78.23
CK + CP-NS	7B	68.35	84.21	75.59	82.49	81.50	82.29	76.34	78.68
CK + TP	7B	68.16	84.09	76.08	81.78	82.36	82.09	76.58	78.74
CK + DeRe (Ours)	7B	71.41 ↑ 4.30	87.30 ↑ 3.27	79.48 ↑ 4.41	85.64 ↑ 3.22	84.26 ↑ 3.35	84.69 ↑ 2.85	77.99 ↑ 1.75	81.54 ↑ 3.31

Table 2: Ablation study of PromptEOL using LLaMA2-7B.

Components	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
Ablation of Different Components:								
ATT	69.74	85.46	76.59	**83.83**	**81.67**	81.55	76.36	79.32
FFN	62.51	79.46	69.32	76.06	74.86	74.01	71.49	72.53
ATT + FFN	**70.03**	**85.81**	**76.73**	83.82	81.66	**81.70**	**76.37**	**79.45**
Ablation of Coefficients:								
ATT	67.96	84.97	75.93	81.69	79.99	78.75	74.07	77.62
FFN	53.67	74.25	61.39	66.53	70.06	65.14	65.26	65.19
ATT + FFN (HS)	58.81	77.01	66.34	73.22	73.56	71.66	69.64	70.03

tion is complementary to prompt design and can be seamlessly integrated with different prompting strategies.

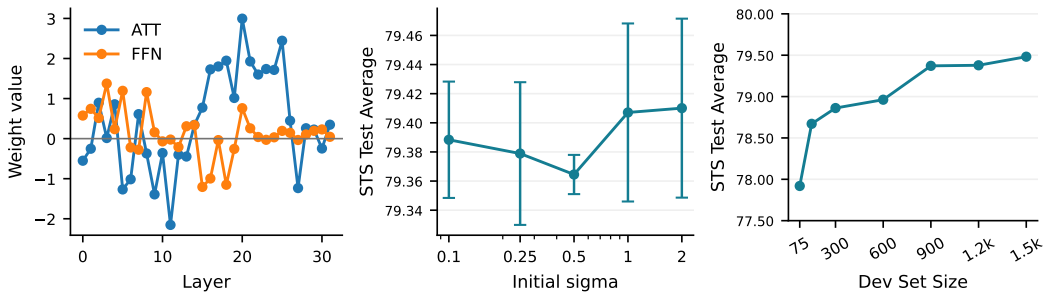
Compared with existing plug-and-play methods, our method also achieves consistently stronger performance. For all four prompting strategies, it outperforms TP on every STS dataset. Notably, even when combined with the basic PromptEOL prompt, our method achieves an average score of 79.45, surpassing the unsupervised contrastive LLM2Vec-Mistral baseline at 78.60. This shows that effectively exploiting the internal components of a frozen LLM can provide strong sentence representations without fine-tuning the model.

4.4 Model Analysis

Ablation Study We further ablate the components and coefficients to verify their effectiveness in Table 2.

First, we ablate different components. Experimental results show that jointly leveraging the outputs of ATT and FFN achieves the best performance. This indicates that, with appropriate weighting, ATT and FFN can be effectively combined to improve semantic representations. Moreover, ATT substantially outperforms FFN, suggesting that the intra-sentence information interaction enabled by self-attention is more effective at capturing the intrinsic semantics of the text. Second, we ablate the coefficients. Similar to Figure 3, without learnable weights, the accumulated attention outputs achieve the best performance. In contrast, the accumulated FFN outputs perform the worst and even degrade the performance of the attention outputs when combined with them.

Visualization of Component Coefficients. Figure 4(a) visualizes the coefficients for the attention and FFN components across different layers. The learned weights exhibit substantial variation across both layers and modules, further confirming that different internal components contribute unequally



(a) (b) (c)

Figure 4: Model analysis using LLaMA2-7B with PromptEOL. (a) coefficients for layer-wise ATT and FFN components; (b) sensitivity to the initial CMA-ES step size σ; (c) effect of the STS-B validation set size on average STS performance.

Table 3: Performance across MTEB(eng, v2) using LLaMA2-7B, including Classification (CLF), Clustering (Clust.), PairClassification (PairCLF), Re-ranking (Rerank), Retrieval (Retr.), STS, and Summarization (Sum.). The number following each task indicates the number of datasets used in that task.

MTEB Tasks	**CLF (8)**	**Clust. (8)**	**PairCLF (3)**	**Rerank (2)**	**Retr. (10)**	**STS (9)**	**Sum. (1)**	**Mean**
PromptEOL	75.74	38.76	60.97	41.44	13.28	66.33	30.42	46.71
PromptEOL + DeRe (***Ours***)	**78.75**	**42.48**	**70.85**	**45.46**	**31.81**	**76.41**	**35.91**	**54.52**

to sentence representations. In particular, attention components in the middle and upper layers tend to receive relatively larger positive weights, whereas FFN coefficients are generally smaller and fluctuate around zero. This pattern is consistent with Figure 3 that attention outputs contain stronger semantic information, while also showing that FFN components are not simply discarded but selectively incorporated through coefficient optimization.

Sensitivity to Initial Step Size. Figure 4(b) studies the effect of the initial CMA-ES step size σ, with error bars showing the variance over five random seeds. Performance varies only modestly across different initializations, with small variance across five random seeds. This suggests that CMA-ES can adapt its search scale effectively, reducing the need for careful tuning of σ.

Effect of Validation Set Size. Finally, Figure 4(c) investigates how the amount of validation data affects performance. Using only a small subset of STS-B already yields substantial improvements, while increasing the validation set size generally leads to further gains and more stable performance. Notably, strong performance can be achieved with only a few hundred sentence pairs, suggesting that learning effective component recombination requires relatively limited supervision.

4.5 Generalization Experiments

Generalization on the MTEB Benchmark Table 3 evaluates the generalization ability of our method on MTEB(eng, v2) (Enevoldsen et al., 2025), with detailed results for each dataset reported in Table 6.

Although the coefficients are optimized only on the STS-B validation set, the resulting representation consistently improves performance across all seven task categories. In particular, the substantial gains on retrieval, pair classification, and STS suggest that the learned recombination captures transferable semantic relationships rather than overfitting to the validation task. The improvements on classification and clustering further show that such benefits extend beyond pairwise similarity estimation to more diverse embedding-based applications. These results consistently demonstrate that our method has strong cross-task generalization ability.

Table 4: 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	STS12	STS13	STS14	STS15	STS16	STS-B	SICK-R	Avg.
`Qwen2.5-7B-base`								
Pretended CoT	65.42	76.88	69.68	76.42	77.04	75.78	70.70	73.13
Pretended CoT + DeRe (Ours)	69.70 ↑ 4.28	82.90 ↑ 6.02	74.93 ↑ 5.25	81.50 ↑ 5.08	80.85 ↑ 3.81	82.33 ↑ 6.55	78.43 ↑ 7.73	78.66 ↑ 5.53
`LLaMA2-13B`								
Pretended CoT	64.27	78.61	69.93	76.37	79.28	75.88	69.04	73.34
Pretended CoT + DeRe (Ours)	70.06 ↑ 5.79	83.53 ↑ 4.92	76.40 ↑ 6.47	84.50 ↑ 8.13	83.22 ↑ 3.94	82.17 ↑ 6.29	74.55 ↑ 5.51	79.20 ↑ 5.86
`LLaMA3.1-8B`								
Pretended CoT	65.77	83.07	72.21	79.46	81.05	77.12	73.05	75.96
Pretended CoT + DeRe (Ours)	70.03 ↑ 4.26	85.03 ↑ 1.96	76.54 ↑ 4.33	84.64 ↑ 5.18	82.99 ↑ 1.94	82.60 ↑ 5.48	77.02 ↑ 3.97	79.83 ↑ 3.87
`Mistral-7B-v0.3`								
Pretended CoT	66.32	83.03	72.61	78.70	79.27	74.70	70.86	75.07
Pretended CoT + DeRe (Ours)	72.78 ↑ 6.46	85.76 ↑ 2.73	77.79 ↑ 5.18	84.41 ↑ 5.71	83.21 ↑ 3.94	82.01 ↑ 7.31	77.29 ↑ 6.43	80.46 ↑ 5.39

Table 5: Top-8 tokens predicted by different methods using LLaMA2-7B.

Sentence	Method	Top-8 Predicted Tokens and Probability
A man with a hard hat is dancing.	PromptEOL	A (0.0879), Dan (0.0408), H (0.0387), hard (0.0243), a (0.0237), A (0.0194), He (0.0187), The (0.0182)
	PromptEOL + DeRe	dan (0.6547), Dan (0.2495), dance (0.0797), Dance (0.0069), Dan (0.0060), dan (0.0014), Hard (0.0009), hard (0.0007)
A young child is riding a horse.	PromptEOL	A (0.1079), a (0.0409), The (0.0395), Child (0.0325), child (0.0272), A (0.0262), a (0.0209), the (0.0196)
	PromptEOL + DeRe	horse (0.9911), horses (0.0068), hors (0.0009), ride (0.0005), rode (0.0004), children (0.0002), hor (0.0000), child (0.0000)

Generalization across Different LLMs Table 4 evaluates the generalization of our method across different LLM backbones, including Qwen2.5-7B-base (Yang et al., 2025), LLaMA2-13B (Touvron et al., 2023), LLaMA3.1-8B (Dubey et al., 2024), and Mistral-7B-v0.3 (Jiang et al., 2023a). The proposed method consistently improves Pretended CoT on all seven STS datasets across all four models, demonstrating that its effectiveness is not tied to a specific model architecture or family. Notably, substantial gains are observed across both relatively older and newer LLMs, suggesting that the benefits of decomposing and recombining internal components are broadly transferable across different backbone models.

4.6 Case Study

Table 5 compares the top predicted tokens before and after applying our weighted recombination. For PromptEOL, the probability mass is relatively dispersed and several high-ranked predictions are generic or function-like tokens. In contrast, our method concentrates the predictions on semantically salient content words, such as “dance” for the first sentence and “horse” for the second. This suggests that the optimized recombination suppresses less informative components while strengthening representations that better capture the core semantics of the input sentence.

5 Conclusion

In this work, we revisit sentence embedding extraction from LLMs by decomposing hidden states into fine-grained attention and FFN components. Our analysis reveals substantial heterogeneity in the semantic contributions of different components across layers and modules, suggesting that directly using hidden states or uniformly combining internal representations can be suboptimal. Based on this observation, we optimize component-wise coefficients with CMA-ES and recombine the decomposed representations to construct more effective sentence embeddings. Extensive experiments on STS benchmarks and MTEB demonstrate consistent improvements across different prompts, models, and task types, showing that fine-grained recombination provides an effective and generalizable way to better exploit the internal representations of LLMs for sentence embeddings.

AI USE STATEMENT

Generative AI tools were used solely to improve the clarity, grammar, and readability of the manuscript. All AI-assisted edits were carefully reviewed and verified by the authors, who take full responsibility for the final content of the paper.

REFERENCES

Eneko Agirre, Daniel Cer, Mona Diab, and Aitor Gonzalez-Agirre. 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)*, pp. 385–393, 7-8 June 2012. URL https://aclanthology.org/S12-1051/.

Eneko Agirre, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, and Weiwei Guo. *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*, pp. 32–43, June 2013. URL https://aclanthology.org/S13-1004/.

Eneko Agirre, Carmen Banea, Claire Cardie, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Weiwei Guo, Rada Mihalcea, German Rigau, and Janyce Wiebe. SemEval-2014 task 10: Multilingual semantic textual similarity. In *Proceedings of the 8th International Workshop on Semantic Evaluation (SemEval 2014)*, pp. 81–91, August 2014. doi: 10.3115/v1/S14-2010. URL https://aclanthology.org/S14-2010/.

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. 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)*, pp. 252–263, June 2015. doi: 10.18653/v1/S15-2045. URL https://aclanthology.org/S15-2045/.

Eneko Agirre, Carmen Banea, Daniel Cer, Mona Diab, Aitor Gonzalez-Agirre, Rada Mihalcea, German Rigau, and Janyce Wiebe. SemEval-2016 task 1: Semantic textual similarity, monolingual and cross-lingual evaluation. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pp. 497–511, June 2016. doi: 10.18653/v1/S16-1081. URL https://aclanthology.org/S16-1081/.

Parishad BehnamGhader, Vaibhav Adlakha, Marius Mosbach, Dzmitry Bahdanau, Nicolas Chapados, and Siva Reddy. LLM2Vec: Large language models are secretly powerful text encoders. In *First Conference on Language Modeling*, 2024. URL https://openreview.net/forum?id=IW1PR7vEBf.

Daniel Cer, Mona Diab, Eneko Agirre, Inigo Lopez-Gazpio, and Lucia Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.

Sachin Chanchani and Ruihong Huang. Composition-contrastive learning for sentence embeddings. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 15836–15848, July 2023. doi: 10.18653/v1/2023.acl-long.882. URL https://aclanthology.org/2023.acl-long.882/.

Zifeng Cheng, Zhonghui Wang, Yuchen Fu, Zhiwei Jiang, Yafeng Yin, Cong Wang, and Qing Gu. Contrastive prompting enhances sentence embeddings in llms through inference-time steering. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025*, pp. 3475–3487, 2025.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 4171–4186, June 2019. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423/.

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

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. A mathematical framework for transformer circuits. *Transformer Circuits Thread*, 2021.

Kenneth C. Enevoldsen, Isaac Chung, Imene Kerboua, Márton Kardos, Ashwin Mathur, David Stap, Jay Gala, Wissam Siblini, Dominik Krzeminski, Genta Indra Winata, Saba Sturua, Saiteja Utpala, Mathieu Ciancone, Marion Schaeffer, Diganta Misra, Shreeya Dhakal, Jonathan Rystrøm, Roman Solomatin, Ömer Veysel Çagatan, Akash Kundu, and et al. MMTEB: massive multilingual text embedding benchmark. In *The Thirteenth International Conference on Learning Representations, ICLR 2025*, 2025. URL https://openreview.net/forum?id=zl3pfz4VCV.

Yuchen Fu, Zifeng Cheng, Zhiwei Jiang, Zhonghui Wang, Yafeng Yin, Zhengliang Li, and Qing Gu. Token prepending: A training-free approach for eliciting better sentence embeddings from llms. In *Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), ACL 2025*, pp. 3168–3181, 2025.

Tianyu Gao, Xingcheng Yao, and Danqi Chen. Simcse: Simple contrastive learning of sentence embeddings. *arXiv preprint arXiv:2104.08821*, 2021.

Nikolaus Hansen and Andreas Ostermeier. Adapting arbitrary normal mutation distributions in evolution strategies: The covariance matrix adaptation. In *Proceedings of 1996 IEEE International Conference on Evolutionary Computation*, pp. 312–317, 1996. URL https://doi.org/10.1109/ICEC.1996.542381.

Albert Q Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, et al. Mistral 7b. *arXiv preprint arXiv:2310.06825*, 2023a.

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

Chankyu Lee, Rajarshi Roy, Mengyao Xu, Jonathan Raiman, Mohammad Shoeybi, Bryan Catanzaro, and Wei Ping. NV-embed: Improved techniques for training LLMs as generalist embedding models. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=lgsyLSsDRe.

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

Ziyue Li and Tianyi Zhou. Your mixture-of-experts LLM is secretly an embedding model for free. In *The Thirteenth International Conference on Learning Representations, ICLR 2025*, 2025.

Marco Marelli, Stefano Menini, Marco Baroni, Luisa Bentivogli, Raffaella Bernardi, and Roberto Zamparelli. 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)*, 2014.

Timothee Mickus, Denis Paperno, and Mathieu Constant. How to dissect a Muppet: The structure of transformer embedding spaces. *Transactions of the Association for Computational Linguistics*, 10:981–996, 2022. doi: 10.1162/tacl_a_00501. URL https://aclanthology.org/2022.tacl-1.57/.

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

2022. doi: 10.18653/v1/2022.findings-acl.146. URL https://aclanthology.org/2022.findings-acl.146/.

Jacob Mitchell Springer, Suhas Kotha, Daniel Fried, Graham Neubig, and Aditi Raghunathan. Repetition improves language model embeddings. In *The Thirteenth International Conference on Learning Representations, ICLR 2025*, 2025.

Yixuan Tang and Yi Yang. Do we need domain-specific embedding models? an empirical investigation. *CoRR*, abs/2409.18511, 2024. URL https://doi.org/10.48550/arXiv.2409.18511.

Raghuveer Thirukovalluru and Bhuwan Dhingra. Geneol: Harnessing the generative power of llms for training-free sentence embeddings. In *Findings of the Association for Computational Linguistics: NAACL 2025*, pp. 2295–2308, 2025.

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

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Improving text embeddings with large language models. *arXiv preprint arXiv:2401.00368*, 2023.

An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, et al. Qwen3 technical report. *arXiv preprint arXiv:2505.09388*, 2025.

Bowen Zhang, Kehua Chang, and Chunping Li. Simple techniques for enhancing sentence embeddings in generative language models. In *International Conference on Intelligent Computing*, pp. 52–64. Springer, 2024.

Yanzhao Zhang, Mingxin Li, Dingkun Long, Xin Zhang, Huan Lin, Baosong Yang, Pengjun Xie, An Yang, Dayiheng Liu, Junyang Lin, et al. Qwen3 embedding: Advancing text embedding and reranking through foundation models. *arXiv preprint arXiv:2506.05176*, 2025.

Yeqin Zhang, Yunfei Wang, Chen Jiaxuan, Ke Qin, Yizheng Zhao, and Cam-Tu Nguyen. LLM-based embeddings: Attention values encode sentence semantics better than hidden states. In *Forty-third International Conference on Machine Learning*, 2026. URL https://openreview.net/forum?id=nAfV9H2Z1u.

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: "

Echo: Rewrite the following paragraph: [TEXT]. The rewritten paragraph: [TEXT]

MetaEOL (**General Category Identification**): In this task, you're presented with a text excerpt. Your task is to categorize the excerpt into a broad category such as 'Education', 'Technology', 'Health', 'Business', 'Environment', 'Politics', or 'Culture'. These categories help in organizing content for better accessibility and targeting. For this task, this sentence : "input sentence" should be classified under one general category in one word: "

MetaEOL (**Opinion vs. Fact Discrimination**): In this task, you're given a statement and you need to determine whether it's presenting an 'Opinion' or a 'Fact'. This distinction is vital for information verification, educational purposes, and content analysis. For this task, this sentence : "input sentence' discriminates between opinion and fact in one word: "

MetaEOL (**Product Review Rating**): In this task, you're given a review from an online platform. Your task is to generate a rating for the product based on the review on a scale of 1-5, where 1 means 'extremely negative' and 5 means 'extremely positive'. For this task, this sentence : "input sentence" reflects the sentiment in one word: "

MetaEOL (**Emotion Detection**): In this task, you're reading a personal diary entry. Your task is to identify the predominant emotion expressed, such as joy, sadness, anger, fear, or love. For this task, this sentence : "input sentence" conveys the emotion in one word: "

MetaEOL (**Similarity Check**): In this task, you're presented with two sentences. Your task is to assess whether the sentences convey the same meaning. Use 'identical', 'similar', 'different', or 'unrelated' to describe the relationship. To enhance the performance of this task, this sentence : "input sentence" means in one word: "

MetaEOL (**Contextual Synonym Detection**): In this task, you're given a sentence and a phrase. Your task is to determine if the phrase can be a contextual synonym within the given sentence. Options include 'yes', 'no', or 'partially'. To enhance the performance of this task, this sentence : "input sentence" means in one word: "

MetaEOL (**Key Fact Identification**): In this task, you're examining a news article. Your task is to extract the most critical fact from the article. For this task, this sentence : "input sentence" encapsulates the key fact in one word: "

MetaEOL (**Entity and Relation Extraction**): In this task, you're reviewing a scientific abstract. Your task is to identify the main entities (e.g., proteins, diseases) and their relations (e.g., causes, treats). For this task, this sentence : "input sentence" highlights the primary entity or relation in one word: "

Table 6: Results on MTEB(eng, v2) using the LLaMA2-7B model.

Dataset	Task	# Test	PromptEOL	Ours
AmazonCounterfactualClassification	Classification	670	69.51	72.28
Banking77Classification	Classification	3 080	78.31	84.21
ImdbClassification	Classification	25 000	92.20	91.03
MTOPDomainClassification	Classification	4 386	90.06	95.35
MassiveIntentClassification	Classification	2 974	73.09	76.44
MassiveScenarioClassification	Classification	2 974	75.94	80.10
ToxicConversationsClassification	Classification	2 048	67.10	70.59
TweetSentimentExtractionClassification	Classification	3 534	59.73	59.98
ArXivHierarchicalClusteringP2P	Clustering	2 048	56.95	57.29
ArXivHierarchicalClusteringS2S	Clustering	2 048	55.75	52.63
BiorxivClusteringP2P.v2	Clustering	53 787	24.51	34.26
MedrxivClusteringP2P.v2	Clustering	37 500	26.47	29.40
MedrxivClusteringS2S.v2	Clustering	37 500	28.84	31.48
StackExchangeClustering.v2	Clustering	2 048	51.89	51.32
StackExchangeClusteringP2P.v2	Clustering	74 914	35.82	37.06
TwentyNewsgroupsClustering.v2	Clustering	59 545	29.81	46.42
SprintDuplicateQuestions	PairClassification	101 000	38.86	60.55
TwitterSemEval2015	PairClassification	16 777	66.13	68.41
TwitterURLCorpus	PairClassification	51 534	77.90	83.59
AskUbuntuDupQuestions	Reranking	7 581	53.30	59.72
MindSmallReranking	Reranking	2 367 791	29.58	31.21
ArguAna	Retrieval	10 080	21.44	41.96
CQADupstackGamingRetrieval	Retrieval	46 896	15.08	36.74
CQADupstackUnixRetrieval	Retrieval	48 454	17.80	30.18
ClimateFEVERHardNegatives	Retrieval	48 416	5.90	14.22
FEVERHardNegatives	Retrieval	164 698	7.42	36.05
FiQA2018	Retrieval	58 286	12.01	23.14
HotpotQAHardNegatives	Retrieval	226 621	6.76	31.80
SCIDOCS	Retrieval	26 657	10.47	18.40
TRECCOVID	Retrieval	171 382	24.60	53.79
Touche2020Retrieval.v3	Retrieval	303 781	11.36	31.77
BIOSSES	STS	100	61.66	74.90
SICK-R	STS	9 927	70.27	76.89
STS12	STS	3 108	61.91	70.26
STS13	STS	1 500	78.99	85.60
STS14	STS	3 750	67.91	76.57
STS15	STS	3 000	72.27	83.63
STS17	STS	250	76.16	86.69
STS22.v2	STS	197	34.84	50.97
STSBenchmark	STS	1 379	73.00	82.20
SummEvalSummarization.v2	Summarization	100	30.42	35.91

Table 7: Comparison between our method and baseline methods across various tasks on the MTEB benchmark.

Model	Clustering			Retrieval			STS			Classification			Reranking		Avg.
	Bior.	Medr.	Twen.	SciF.	NFCo.	Argu.	STS17	SICK-R	STSB.	Bank.	Emot.	Spri.	Stac.	SciD.	
PromptEOL	22.49	21.14	31.47	27.16	13.59	11.65	79.67	73.82	75.32	76.37	47.13	26.08	37.65	66.22	43.55
AlignedWVA	28.82	23.45	31.20	45.31	25.13	32.44	83.33	73.80	76.74	82.09	**52.68**	**60.93**	43.52	75.76	52.51
DeRe	**32.40**	**27.83**	**44.66**	**58.28**	**29.57**	**41.96**	**86.69**	**76.89**	**82.20**	**84.21**	51.16	60.55	**47.62**	**81.95**	**57.57**

B COMPARISON WITH ALIGNEDWVA

Following the experimental setup of AlignedWVA, Table 7 compares our method with PromptEOL and AlignedWVA using LLaMA2-7B across diverse tasks on MTEB. Both AlignedWVA and our

method use PromptEOL as the prompting strategy. Our method still uses only the STS-B validation set, which is substantially smaller than the multi-task validation set used by AlignedWVA.

Our method achieves the best average performance of 57.57, outperforming PromptEOL and AlignedWVA by 14.02 and 5.06 points, respectively. Compared with AlignedWVA, our method obtains better results on 12 out of 14 datasets, with particularly substantial improvements on clustering, retrieval, and reranking tasks. Notably, the gains extend well beyond semantic textual similarity, indicating that the learned component recombination captures transferable semantic information that benefits a broad range of embedding-based tasks. While the performance on two classification datasets is slightly lower than AlignedWVA, our method remains competitive in classification and achieves a higher overall average across the three classification datasets. These results demonstrate that fine-grained recombination of internal components provides stronger and more generalizable sentence representations than directly using hidden states or manually aggregating selected internal representations.