Look Both Ways and No Sink: Converting LLMs into Text Encoders without Training

Ziyong Lin[*1,4], **Haoyi Wu**[*2,3], **Shu Wang**[5], **Kewei Tu**[†2,3], **Zilong Zheng**[†1], **Zixia Jia**[†1],
[1]State Key Laboratory of General Artificial Intelligence, BIGAI, Beijing, China
[2]School of Information Science and Technology, ShanghaiTech University
[3]Shanghai Engineering Research Center of Intelligent Vision and Imaging
[4]Tsinghua University [5]University of California, Los Angeles
linziyon22@mails.tsinghua.edu.cn {wuhy1, tukw}@shanghaitech.edu.cn
shuwang0712@ucla.edu {jiazixia, zlzheng}@bigai.ai

Abstract

Recent advancements have demonstrated the advantage of converting pretrained large language models into powerful text encoders by enabling bidirectional attention in transformer layers. However, existing methods often require extensive training on large-scale datasets, posing challenges in low-resource, domain-specific scenarios. In this work, we show that a pretrained large language model can be converted into a strong text encoder without additional training. We first conduct a comprehensive empirical study to investigate different conversion strategies and identify the impact of the attention sink phenomenon on the performance of converted encoder models. Based on our findings, we propose a novel approach that enables bidirectional attention and suppresses the attention sink phenomenon, resulting in superior performance. Extensive experiments on multiple domains demonstrate the effectiveness of our approach. Our work provides new insights into the training-free conversion of text encoders in low-resource scenarios and contributes to the advancement of domain-specific text representation generation. Our code is available at https://github.com/bigai-nlco/Look-Both-Ways-and-No-Sink.

1 Introduction

High-quality text representations serve as the foundation for a wide range of downstream applications, including text classification (Li et al., 2021), semantic similarity computation (Zhelezniak et al., 2019), and information retrieval (Iida and Okazaki, 2021; Jia et al., 2023). In recent years, the emergence of Large Language Models (LLMs) has led to a surge of interest in converting them into sentence encoders by enabling bidirectional attention and conducting additional training (BehnamGhader et al., 2024; Li and Li, 2024; Li et al., 2023; Lee et al., 2024). These approaches have demonstrated significant advancements in general domains [1], revolutionizing the research on text representations.

However, these methods often rely on extensive training with large-scale datasets. For instance, the NV-Embed model (Lee et al., 2024) collects dozens of datasets and performs two-stage training to convert a decoder model into an encoder model, which requires substantial computational resources and time. This poses a significant challenge in low-resource, domain-specific scenarios, as general representation models often struggle to perform optimally in these domains due to the scarcity of domain data and the presence of domain gaps (Tang and Yang, 2024).

In this work, we aim to address the challenge of generating high-quality text representations in low-resource scenarios and ask the following question: *How can we convert a pretrained LLM to a strong text encoder without additional training?*

To address this challenge, we first conduct a comprehensive empirical study to investigate the impact of different conversion strategies. Specifically, we modify the causal attention masks of some transformer layers in LLMs to allow backward or bidirectional attention. We then build new encoders through different strategies to incorporate the modified layers into the original model. Surprisingly, we find that the models with backward attention layers outperform those with bidirectional attention layers. Further analysis reveals that the attention sink phenomenon (Xiao et al., 2023), where the model tends to focus on the first token, significantly affects the performance of the converted text encoder. By incorporating backward attention, the model effectively eliminates the attention sinks,

[*] Equal Contribution.
[†] Corresponding Authors.

[1] https://huggingface.co/spaces/mteb/leaderboard_legacy

Proceedings of the 63rd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pages 22839–22853
July 27 - August 1, 2025

resulting in improved performance.

Based on these insights, we propose a novel approach to converting a pretrained LLM into a text encoder. Specifically, we enable bidirectional attention and suppress the attention sink phenomenon by masking the first token in the attention mechanism. This approach allows the model to distribute its attention more reasonably across the context, avoiding the excessive focus on the first tokens. Consequently, our converted encoder is capable of generating higher-quality text representations and effectively leveraging the knowledge embedded in the pretrained LLMs. Notably, for encoder models that are converted from transformer decoders by enabling bidirectional attention followed by additional finetuning, masking the first token still benefits their task performance. Through extensive experiments, we demonstrate that this approach significantly enhances the performance of downstream tasks in various domains, surpassing the capabilities of both well-trained domain-general and domain-specific text encoders, reflecting our approach's generalization and applicability.

We hope our work can bring new insights into the training-free conversion of text encoders in low-resource domain-specific scenarios. Our contributions can be summarized as follows:

1. We conduct a comprehensive empirical study to investigate the impact of different strategies to convert a pretrained transformer decoder into an encoder model.

2. Through our analysis, we identify the impact of the attention sink phenomenon on the performance of the converted encoder models.

3. We propose a novel training-free decoder-to-encoder conversion approach that significantly improves the performance of pretrained decoders in various domains.

2 Related Work

Previous studies have investigated various techniques for converting pretrained large language models (LLMs) into text encoders. These approaches aim to capture more contextual information by enabling a) bidirectional attention in all transformer layers (BehnamGhader et al., 2024; Lee et al., 2024) or only in the last layer (Li and Li, 2024), b) concatenating the token representations of a small backwardLM with those of an existing LLM (Goto et al., 2025), or c) repeating the input text to allow each token to attend to its right-hand side context (Springer et al., 2024). Most of these approaches depend on additional datasets to continually pretrain (generally contrastive learning for sentence embedding) the converted models and enhance their performance.

There have also been studies exploring the use of in-context learning of LLMs for sentence embedding tasks (Jiang et al., 2023). However, it is worth noting that this approach introduces significant computational overhead during inference, which limits its direct applicability to tasks that require fast processing, such as information retrieval.

3 Empirical Study

Based on previous research, we conduct a systematic investigation into various architectural configurations for converting a pretrained decoder into an encoder model without additional training. We also introduce several novel configurations that have not been previously explored in the literature.

3.1 Architectural Configurations

As demonstrated in Figure 1, we primarily investigate four architectures for converting a pretrained decoder into an encoder: *INPLACE*, *INTER*, *EXTRA*, and *EXTEND*. Each architecture comes with two settings: backward (*BACK*) and bidirectional (*BIDIR*). These settings differ in how they convert the attention mechanism of a decoder:

- **BIDIR** refers to enabling bidirectional attention in a standard decoder layer (hereafter referred to as the forward layer) by directly removing the causal attention masks, thereby converting the forward layer into a bidirectional layer.

- **BACK** refers to applying reversed causal attention masks in a standard decoder layer, restricting each token to attending to itself and its subsequent tokens, thereby transforming the forward layer into a backward layer.

The four architectures utilize the converted layers in distinct ways:

- **INPLACE** replaces forward layers with converted layers.

- **INTER** places both the forward layer and the converted layer in parallel in the same layer, where they share the same input. Their outputs are summed up to form the final output.

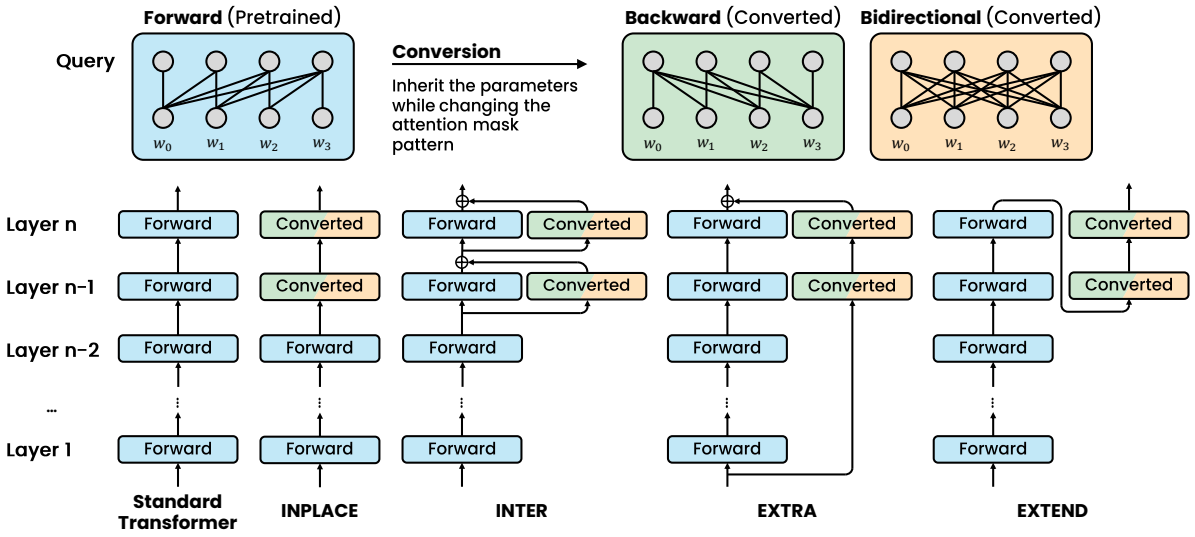

Figure 1: The schematic diagrams of four model architectures (number of converted layers $k = 2$). Either converted backward attention or bidirectional attention is incorporated into all the architectures.

- **EXTRA** involves using an additional independent model that consists of the converted layers. The final outputs are the sum of the outputs from the LLM and the extra model.
- **EXTEND** adds the converted layers on top of the standard LLM.

In each architecture, the converted layers can either use *BACK* or *BIDIR* setting, resulting in eight possible configurations: *INPLACE-BACK*, *INPLACE-BIDIR*, *INTER-BACK*, *INTER-BIDIR*, *EXTRA-BACK*, *EXTRA-BIDIR*, *EXTEND-BACK*, and *EXTEND-BIDIR*. Among these configurations, *INPLACE-BIDIR*, *EXTRA-BACK* and *EXTEND-BIDIR* have been previously explored in the literature (BehnamGhader et al., 2024; Lee et al., 2024; Li and Li, 2024; Goto et al., 2025), while the other configurations are novel and have not been previously investigated.

3.2 Experimental Settings

Focusing on the training-free conversion, we adopt specific initialization strategies to facilitate the direct utilization of pretrained LLMs without additional training. Within the *EXTRA* architecture, the parameters of the transformer blocks are initialized using the corresponding parameters from the LLMs. For the *EXTEND* architecture, the parameters of the additional transformer blocks are initialized with the parameters of the final layer of the LLMs.

Following Li and Li (2024), given a pretrained transformer decoder with n layers, we explore the impact of converting the top k layers in each architecture. Figure 1 illustrates the diagrams of these architectures when $k = 2$. We test the converted model without any further training on two types of tasks: token-level and sentence-level tasks.

▷ For token-level tasks, following Goto et al. (2025), we employ the Named Entity Recognition (NER) task from the CoNLL 2003 benchmark (Sang and De Meulder, 2003). The representation of each input token is obtained from the output of the final transformer block, and then a task-specific linear classifier is fine-tuned on the frozen representations. We adopt the F1 score as the evaluation metric.

▷ For sentence-level tasks, we select 15 representative tasks from the MTEB benchmark (Muennighoff et al., 2022), following the task selection of BehnamGhader et al. (2024). For each task, a task-specific instruction is prepended to the sentence as input. The final sentence representation is then obtained by averaging the embeddings of the tokens, excluding the instruction. Specific prompts used for these tasks can be found in Appendix B. The evaluation metrics of different tasks remain the same as in previous work (BehnamGhader et al., 2024; Lee et al., 2024). We report the average score over the 15 tasks.

In this section, we primarily conduct experiments using the TinyLlama-1.1B model (Zhang et al., 2024).

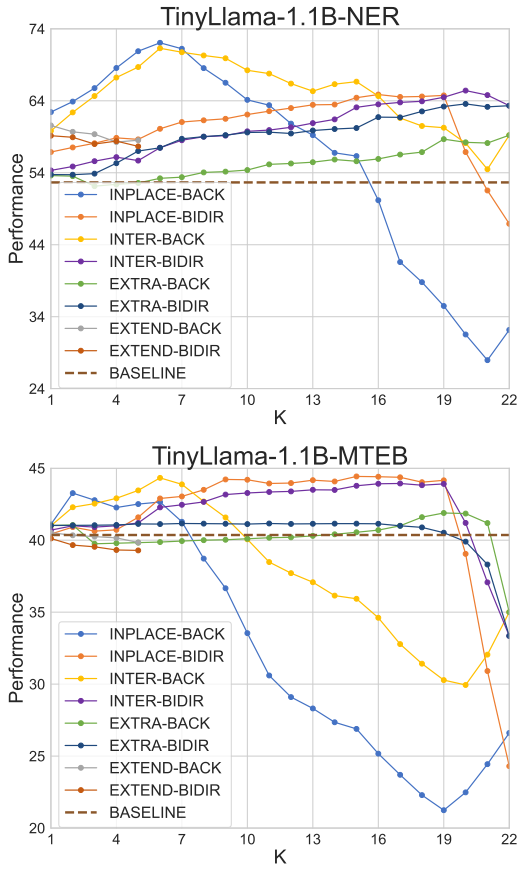

Figure 2: Performance of TinyLlama-1.1B under different architectural configurations varying converted layer number k. In the *EXTEND* architecture, k is bounded by 5 due to the consistent downward trend. BASELINE represents the original TinyLlama-1.1B without any converted layers.

3.3 Results

Our results are shown in Figure 2. Overall, ❶ the best performances are achieved by *INPLACE-BACK* and *INTER-BACK* in the NER task and MTEB tasks, respectively. *INPLACE-BIDIR* outperforms *INTER-BIDIR* in most cases, despite *INTER-BIDIR* requiring more computations. ❷ Both *EXTRA* and *EXTEND* architectures perform similarly to the unconverted original TinyLlama on MTEB tasks, while underperforming in comparison to the *INPLACE* and *INTER* architectures on both NER and MTEB tasks. Based on these preliminary observations, we focus on the *INPLACE* and *INTER* architectures and make more detailed analyses below:

Finding 1: It is beneficial to keep some layers as forward layers In both NER and MTEB tasks, *INPLACE-BACK* and *INTER-BACK* demonstrate a continuous declining trend from $k = 6$, while *INPLACE-BIDIR* and *INTER-BIDIR* also start declining from $k = 19$. This indicates that LLMs rely heavily on the dependencies established by the forward layers. When these dependencies are disrupted by converting too many attention layers to backward or bidirectional, the model's ability to maintain coherent context and generate robust representations diminishes.

Finding 2: Backward attention surpasses bidirectional attention when k is small Since enabling bidirectional attention intuitively provides each token with more context information, potentially leading to better representation, most previous works have focused on directly removing the causal masks to allow bidirectional attention. However, we are surprised to find that converting forward layers to backward layers exhibits superior performance compared with converting to bidirectional layers when k, the number of converted layers, is small, especially in NER tasks.

3.4 Analysis

We analyze the observations highlighted in Finding 1 and Finding 2 in this subsection. To gain deeper insights into these two phenomena, we conduct a further investigation of the underlying attention mechanisms and seek to provide an explanation. To this end, we randomly select 8 samples from the CoNLL 2003 dataset and visualize each layer's mean attention maps of the forward layer, backward layer, and bidirectional layer.

Basic Observation: As illustrated in Figure 3, the distributions of the original forward attention scores exhibit distinct patterns across different layers. In the bottom three layers, these scores are relatively dispersed (*e.g.*, subfigures A and B from Layer 1 and Layer 3, respectively). However, beginning from the fourth layer, the attention scores concentrate on the first token (*e.g.*, subfigures C and D for Layer 4 and Layer 9). This phenomenon is consistent with the findings of Xiao et al. (2023), referred to as the **Attention Sink**. We define an "attention-sink layer" as a forward layer exhibiting this attention-sink behavior.

When attention-sink layers are converted to bidirectional layers, the attention-sink phenomenon persists (compare subfigures E and F in Figure 3), limiting the model's use of contextual information, despite each token gaining access to subsequent tokens However, an advantage of the existing attention sink is that the attention distributions in

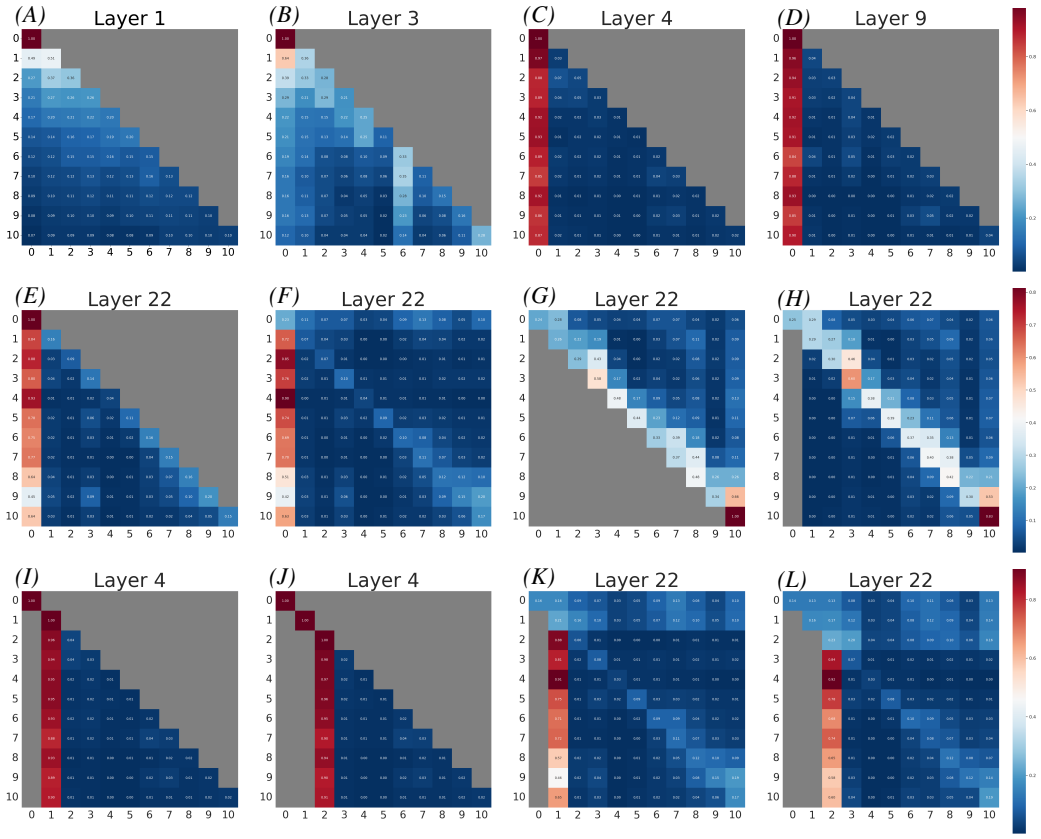

Figure 3: Attention map visualization of different layers and masking strategies. (A)-(E) show attention score distributions in specific layers of a standard Transformer decoder. (F)-(H) display attention scores in the final layer with various masking methods. (I)-(K) demonstrate the attention sink shifting phenomenon, where masking the initial tokens causes the sink token to shift to the subsequent ones.

forward layers are not largely changed when they are converted to bidirectional layers, leading to moderate but stable performance improvements. When bidirectional layers replace non-attention-sink layers (Layers 1–3), the attention distributions undergo drastic changes. This may explain the significant performance drop observed in both *BIDIR* settings when $k = 20$ in Figure 2.

In contrast, converting attention-sink layers to backward layers effectively eliminates the sink effect, as each token can no longer attend to the first token. This adjustment allows tokens to focus more on themselves and their neighboring tokens (subfigure G in Figure 3). However, backward layers inherently limit tokens from attending to preceding tokens, leading to a trade-off between *preserving forward information and mitigating the attention sink effect*. This trade-off may explain why the *BACK* settings outperform the *BIDIR* settings when k is small but begin to degrade in performance when k increases to a certain number, specifically at $k = 6$ in Figure 2. Visualizations of more layers are demonstrated in Appendix F.1.

3.5 Does Addressing Attention Sink Enhance *BIDIR* Settings?

According to the findings and analysis in Section 3.3 and Section 3.4, we conclude a powerful encoder indeed requires each token to attend to both its proceeding and subsequent information. While backward layers exhibit superior performance (due to mitigating the attention sink), their effectiveness is inherently constrained by their inability to incorporate preceding tokens. General bidirectional layers are a more natural choice; however, they are hindered by the attention-sink effect, which prevents them from truly attending to the entire context, resulting in inferior performance compared to backward settings. So in this section, we investigate whether addressing the attention sink issue in the *BIDIR* setting can further enhance representation capabilities.

Setup From Figure 3, we observe the attention sink phenomenon concentrated on the first token. Typically, the first token in each input is a special system token (e.g., <\s> for various Llama mod-

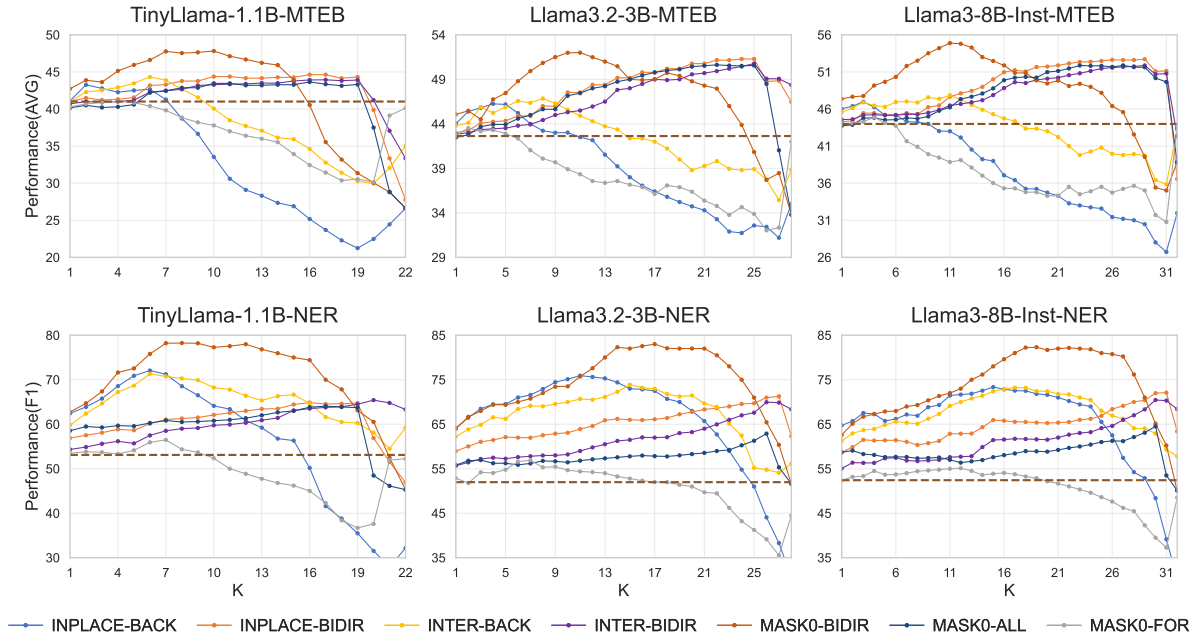

Figure 4: Performance of various models on word-level and sentence-level tasks under different settings.

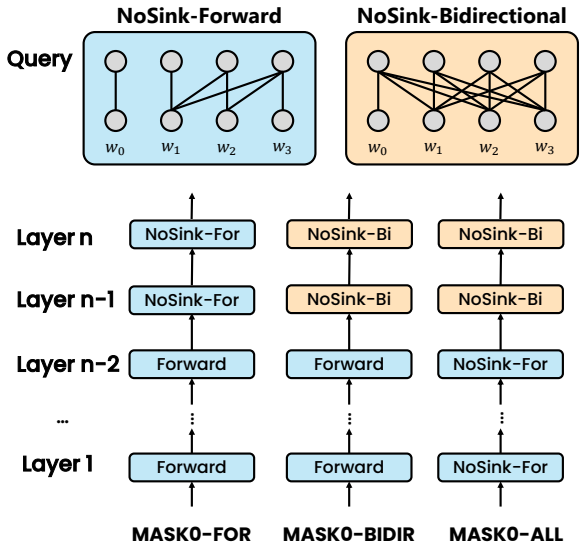

Figure 5: The schematic diagrams of the three model structures. NoSink-For and NoSink-Bi are abbreviations of NoSink-Forward and NoSink-Bidirectional, respectively.

els). A natural approach to mitigate the attention sink is to *mask the first token in forward or bidirectional layers*. Thereby, the forward layers can be converted into *NoSink-Forward* layers (where each token can attend itself and all its previous tokens except for the first one) or *NoSink-Bidirectional* layers (where each token can see all other tokens except the first one), resulting in three architecture configurations, *MASK0-FOR*, *MASK0-BIDIR*, and *MASK0-ALL*, demonstrated in Figure 5 (We adopt *INPLACE* architecture in these configurations because of its computational efficiency). We conduct experiments with the same setting as Section 3.2 but utilizing these three configurations. Furthermore, to assess the generalizability of our findings across different model sizes, we further conduct experiments using the Llama3.2-3B and Llama3-8B-Instruct models (Grattafiori et al., 2024).

Finding 3: *MASK0-BIDIR* achieves the best performance As shown in Figure 4, *MASK0-FOR* and *MASK0-BIDIR* follow a similar trend to the *BACK* settings, which further verify our hypothesis that attention sink influences the representation ability of LLMs. The peak performance of *MASK0-BIDIR* surpasses all other configurations across all model sizes. The visualization of *MASK0-BIDIR*'s attention scores reveals the elimination of the attention sink phenomenon. The attention distribution appears more reasonable, with each token focusing more on itself and its neighboring tokens, as shown in subfigure H in Figure 3.

Finding 4: Attention sink does not disappear if we mask the first token of all layers. We observe in Figure 4 that the curves for *MASK0-ALL* exhibit similar values and trends to those of *INPLACE-BIDIR*. Upon visualizing the attention scores for the *MASK0-ALL* setting, we find that the attention sink becomes concentrated on the second token when the first token is masked across

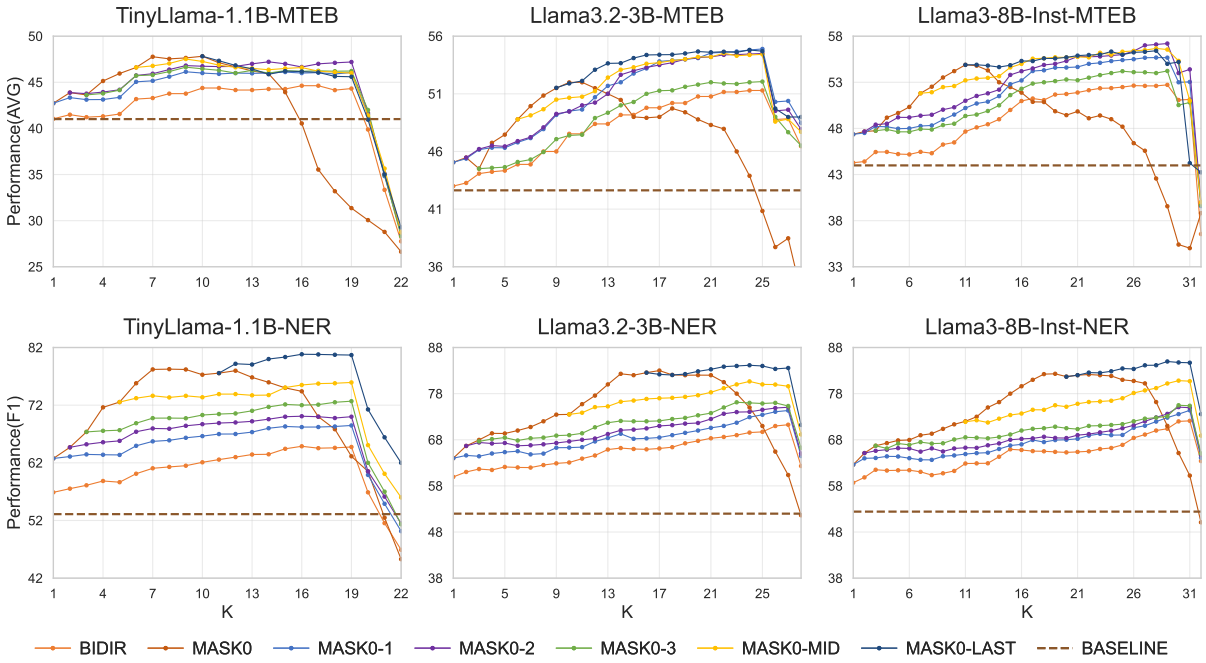

Figure 6: Performance of various models on word-level and sentence-level tasks under *MASK0&BIDIR* settings.

all layers. Moreover, when both the first and second tokens are masked in all layers, the third token emerges as the attention sink (shown in sub-figure I, J for forward layers, K, L for bidirectional layers in Figure 3). Regardless of how many tokens are masked across all layers, the attention sink does not disappear but rather shifts to the first non-masked token (shown in Appendix F.2). We provide some preliminary explanations for this shifting of the attention sink in Appendix C.

3.6 Combine the Advantages of *INPLACE-BIDIR* and *MASK0-BIDIR*

From the results in Section 3.5, we observe that while *MASK0-BIDIR* achieves the best performance across multiple tasks, the optimal value of k is relatively smaller compared to *INPLACE-BIDIR*. For instance, considering TinyLlama-1.1B on the MTEB task, *MASK0-BIDIR* attains its best performance when $k = 10$. Increasing the number of nosink-bidirectional layers beyond this value leads to a decline in model performance. In contrast, for *INPLACE-BIDIR*, adding more bidirectional layers continues to enhance the model's performance. This observation prompts an intriguing question: when *MASK0-BIDIR* reaches its optimal performance, can further augmenting the model with additional bidirectional layers, rather than nosink-bidirectional layers, continue to improve its performance?

Setup Under the new configuration, k still represents the total number of converted layers, but the converted layer i may be two types: either nosink-bidirectional when $i \leq k_0$ or bidirectional when $i > k_0$. Specifically, assuming the model has a total of L layers, the input first passes through $L - k$ forward layers, followed by $k - k_0$ bidirectional layers, and finally through k_0 nosink-bidirectional layers to produce the final output. Consequently, the model can exhibit L^2 possible structural variations under this setup. For simplicity, we selected several representative configurations for testing. *MASK0-1*, *MASK0-2*, and *MASK0-3* denote configurations where k_0 is set to 1, 2, and 3, respectively. *MASK0-LAST* represents the configuration where k_0 is set to the optimal k value under the *MASK0-BIDIR* setting, while *MASK0-MID* indicates k_0 is set to half of the optimal k value. We collectively refer to all these new architecture configurations as *MASK0&BIDIR*.

Finding 5: The performance of the models can still be further improved As shown in Figure 6, most of the models demonstrate noticeable enhancements over *MASK0-BIDIR* across the remaining tasks (except TinyLlama-1.1B on the MTEB task). It is evident that the trend of *MASK0&BIDIR* closely resembles that of *INPLACE-BIDIR*, with *MASK0-LAST* achieving the best performance among all configurations. This suggests that when

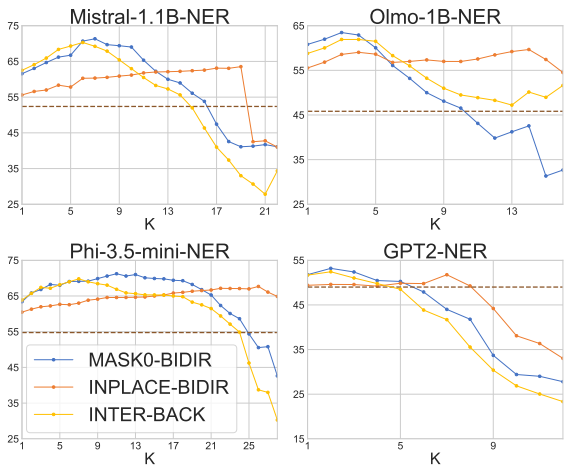

Figure 7: Performance of various models on the CONLL2003 NER task.

k is set to the optimal value for *INPLACE-BIDIR* and k_0 is set to the optimal value for *MASK0-BIDIR*, *MASK0&BIDIR* generally attains its peak performance. Interestingly, current decoder-to-encoder embedders, such as NV-Embed (Lee et al., 2024), which convert large language models (LLMs) into sentence encoders by enabling bidirectional attention and applying additional training, typically activate bidirectional attention across all layers. In this context, converting bidirectional layers to nosink-bidirectional layers in decoder-to-encoder models is analogous to setting k to the total number of model layers and increasing k_0. Therefore, the *MASK0&BIDIR* setting can also be applied to certain decoder-to-encoder embedders beyond original LLM decoders. We demonstrate its effectiveness in the following Experiments section (Section 4).

Verification with Different Decoder Models We primarily investigate the performance of Llama models across different sizes to determine the optimal encoder configuration. To ensure that our conclusion is not specific to a single decoder architecture, we further evaluate the performance of our text encoder with various decoder models on the NER task, including GPT-2 (Radford et al., 2019), Mistral-1.1B[2], Olmo-1B[3] (Groeneveld et al., 2024), and Phi-3.5-mini-instruct[4] (Abdin et al., 2024). As shown in Figure 7, the *MASK0-BIDIR* encoder consistently achieves the highest peak performance across all tested decoders.

[2]https://huggingface.co/optimum/mistral-1.1b-testing
[3]https://huggingface.co/allenai/OLMo-1B-hf
[4]https://huggingface.co/microsoft/Phi-3.5-mini-instruct

General	MTEB	CoNLL	AVG
BGE-M3	66.8	70.1	68.4
NV-Embed-V2	74.7	70.7	72.7
+*MASK0&BIDIR*	**75.8**	**85.9**	**80.8**
GTE-Qwen2-7B	70.9	71.5	71.2
+*MASK0&BIDIR*	74.1	80.2	77.2
E5-mistral	68.4	57.9	63.1
+*INPLACE-BIDIR*	70.9	74.8	72.9
+*INTER-BACK*	69.5	79.8	74.7
+*MASK0-BIDIR*	72.8	83.6	78.2
+*MASK0&BIDIR*	**74.0**	**85.2**	**79.6**

Table 1: Domain-general task performance of various models.

4 Experiments on Various Tasks

Based on the insights from the previous section, both the *MASK0-BIDIR* and *MASK0&BIDIR* configurations yield highly effective text encoders converted from a pretrained transformer decoder without any additional training. In this section, we conduct experiments to verify the effectiveness of these configurations on domain-general tasks and further extend the evaluation to domain-specific tasks.

4.1 Experiment Settings

In addition to the general domain, we follow Cheng et al. (2023) and test three specific domains: medicine, finance, and law. For all tasks, we test domain-general small encoder models (BGE-M3 (Chen et al., 2024), which is a popular and strong encoder) and large decoder-to-encoder embedders (NV-Embed-V2 (Lee et al., 2024) and GTE-Qwen2-7B-Instruct (Li et al., 2023), which have demonstrated their great advantages on MTEB tasks). For domain-general tasks, we additionally evaluate the large pure decoder models (E5-mistral-7B (Wang et al.), which also performs well on MTEB tasks). For domain-specific tasks, we further test domain-specific small encoder models and large decoder models. Details on the specific models used in these domains, as well as task descriptions and evaluation metrics, are provided in Appendix A.

The experimental setup for domain-general tasks remains identical to that described in the previous section, while each domain-specific task is formulated as a classification problem, leveraging the representations derived from the model. We do not concatenate prompts in front of the sentences. After encoding the text sequence, we train an additional logistic regression classifier to classify the text embedding vector. All model parameters are

Medicine	PQAL	Chem	MQP	RCT	AVG
In-Context Learning	**66.0**	47.6	73.0	56.7	60.8
BGE-M3	63.0	68.9	77.7	69.8	69.9
MedicalBert	54.0	53.1	64.8	53.4	56.3
NV-Embed-V2	57.0	77.1	80.8	74.3	72.3
+*MASK0&BIDIR*	62.5	**77.9**	**81.4**	**84.6**	**76.6**
GTE-Qwen2-7B	56.2	78.0	77.7	77.1	72.3
+*MASK0&BIDIR*	63.0	77.9	78.8	82.9	75.7
BioMed-Llama	56.0	77.9	66.8	81.6	70.6
+*INPLACE-BIDIR*	59.0	78.4	73.4	82.5	73.3
+*INTER-BACK*	57.5	78.2	72.8	80.5	72.3
+*MASK0-BIDIR*	63.0	79.0	74.8	**88.0**	76.2
+*MASK0&BIDIR*	**65.5**	**79.4**	**75.2**	87.5	**76.9**

Finance	FIQA	FPB	NER	AVG
In-Context Learning	75.1	62.5	68.2	68.6
BGE-M3	81.0	94.2	68.2	81.3
FinBert	78.0	98.7	59.1	78.6
NV-Embed-V2	86.9	96.2	77.6	86.9
+*MASK0&BIDIR*	**87.9**	**97.8**	**83.1**	**89.6**
GTE-Qwen2-7B	83.3	95.8	67.6	82.2
+*MASK0&BIDIR*	85.0	97.6	81.3	88.0
Finma-7B	91.3	99.1	67.8	86.1
+*INPLACE-BIDIR*	93.2	99.1	69.6	87.3
+*INTER-BACK*	91.9	98.3	81.6	90.3
+*MASK0-BIDIR*	93.9	99.8	87.1	93.6
+*MASK0&BIDIR*	**94.4**	**99.8**	**89.5**	**94.5**

Law	SCOTUS		ToS	AVG
	mic-F1	mac-F1		
In-Context Learning	30.0	20.1	84.9	45.0
BGE-M3	68.4	46.8	84.7	66.6
InlegalBert	66.0	40.1	82.3	62.8
NV-Embed-V2	76.3	68.2	88.8	77.8
+*MASK0&BIDIR*	**78.7**	**71.5**	**91.0**	**80.4**
GTE-Qwen2-7B	73.9	62.6	89.1	75.2
+*MASK0&BIDIR*	77.3	69.9	91.5	79.6
LLama-Lawyer	73.7	61.1	89.7	74.8
+*INPLACE-BIDIR*	75.9	65.6	90.5	77.3
+*INTER-BACK*	74.1	64.6	90.3	76.3
+*MASK0-BIDIR*	75.9	65.6	90.7	77.4
+*MASK0&BIDIR*	**76.8**	**66.0**	**91.5**	**78.1**

Table 2: Domain-specific task performance of various models

freezed except the additional classifier. Following Muennighoff et al. (2022), the training uses LBFGS (Liu and Nocedal, 1989) to optimize the parameters and takes 100 iterations. For tasks with limited sample sizes, the dataset was augmented using the SMOTE technique.

Based on original LLMs, we test four encoder-converted settings: 1) *INPLACE-BIDIR*, 2) *INTER-BACK*, 3) *MASK0-BIDIR*, and 4) *MASK0&BIDIR*. Based on decoder-to-encoder embedders, we specifically test the *MASK0&BIDIR* setting (setting k to the total number of model layers).

We select the optimal value of k and k_0 based on the validation set and then evaluate the model on the test set for each task. Additionally, we report the best result in Cheng et al. (2023) which solves these tasks by in-context learning.

4.2 Results

Table 1 and Table 2 present the performance of various models in domain-general and domain-specific tasks, respectively. It is observed that the in-context learning approach exhibits lower performance compared to most text encoder models. Additionally, smaller encoder models tend to perform worse than larger decoder models, even when the latter are used as encoders without conversion. Furthermore, the application of *MASK0&BIDIR* significantly enhances the performance of large decoder models and decoder-to-encoder models that have undergone extensive training, surpassing all other models. This highlights the effectiveness and generalizability of *MASK0&BIDIR* structure, as it consistently improves performance across different models and tasks.

4.3 Additional Utility in LLM Deployment

In addition to its utility in low-resource domain-specific scenarios, our approach offers a significant advantage in modern LLM applications by directly utilizing the pretrained parameters of the model. This approach has the potential to greatly reduce memory requirements, as both the language model and text encoders can be deployed within the same memory space. For instance, in a Retrieval-Augmented Generation (RAG) (Wang et al., 2024) system, there is no longer a need to deploy a separate retriever for encoding documents. Instead, the chat model itself can serve as the retriever, leading to improved memory efficiency and system performance.

5 Conclusion

In this work, we systematically investigated different approaches to converting transformer decoders into text encoders. We found that the attention sink phenomenon significantly impacts the performance of converted models. By masking the first token and enabling bidirectional attention, we can effectively improve the performance of the converted encoder model. Our approach has been validated on a variety of tasks and domains, demonstrating its effectiveness and potential applicability in modern LLM applications.

Limitation

Our approach primarily focuses on training-free conversion methods, which are particularly advantageous in domains with limited training data. In this work, we do not explore continual pretraining approaches and it is not clear whether these approaches can further enhance the converted text encoder. We leave this for future work.

While our approach demonstrates strong applicability across three domains, conducting comprehensive validation across a broader range of domains would be valuable for further establishing its generalizability and robustness.

The number of conversion layers, denoted as k and k_0, are significant hyperparameters that affect the model's performance on specific tasks. The optimal k and k_0 value vary across different datasets and are non-trivial to determine. Developing efficient approaches to select them for a given task is another important project to explore.

Acknowledgements

The authors thank the reviewers for their insightful suggestions on improving the manuscript. This work is supported by the Opening Project of the State Key Laboratory of General Artificial Intelligence (SKLAGI2024OP08) and the National Natural Science Foundation of China (62376031).

References

Marah Abdin, Jyoti Aneja, Hany Awadalla, Ahmed Awadallah, Ammar Ahmad Awan, Nguyen Bach, Amit Bahree, Arash Bakhtiari, Jianmin Bao, Harkirat Behl, and 1 others. 2024. Phi-3 technical report: A highly capable language model locally on your phone. *arXiv preprint arXiv:2404.14219*.

Julio Cesar Salinas Alvarado, Karin Verspoor, and Timothy Baldwin. 2015. Domain adaption of named entity recognition to support credit risk assessment. In *Proceedings of the Australasian Language Technology Association Workshop 2015*, pages 84–90.

D Araci. 2019. Finbert: Financial sentiment analysis with pre-trained language models. *arXiv preprint arXiv:1908.10063*.

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*.

Ilias Chalkidis, Manos Fergadiotis, Prodromos Malakasiotis, Nikolaos Aletras, and Ion Androutsopoulos. 2020. Legal-bert: The muppets straight out of law school. *arXiv preprint arXiv:2010.02559*.

Jianlv Chen, Shitao Xiao, Peitian Zhang, Kun Luo, Defu Lian, and Zheng Liu. 2024. Bge m3-embedding: Multi-lingual, multi-functionality, multi-granularity text embeddings through self-knowledge distillation. *arXiv preprint arXiv:2402.03216*.

Daixuan Cheng, Shaohan Huang, and Furu Wei. 2023. Adapting large language models via reading comprehension. In *The Twelfth International Conference on Learning Representations*.

Franck Dernoncourt and Ji Young Lee. 2017. Pubmed 200k rct: a dataset for sequential sentence classification in medical abstracts. *arXiv preprint arXiv:1710.06071*.

Takumi Goto, Hiroyoshi Nagao, and Yuta Koreeda. 2025. Acquiring bidirectionality via large and small language models. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 1711–1717.

Aaron Grattafiori, Abhimanyu Dubey, Abhinav Jauhri, Abhinav Pandey, Abhishek Kadian, Ahmad Al-Dahle, Aiesha Letman, Akhil Mathur, Alan Schelten, Alex Vaughan, Amy Yang, Angela Fan, Anirudh Goyal, Anthony Hartshorn, Aobo Yang, Archi Mitra, Archie Sravankumar, Artem Korenev, Arthur Hinsvark, and 542 others. 2024. The llama 3 herd of models. *Preprint*, arXiv:2407.21783.

Dirk Groeneveld, Iz Beltagy, Pete Walsh, Akshita Bhagia, Rodney Kinney, Oyvind Tafjord, Ananya Harsh Jha, Hamish Ivison, Ian Magnusson, Yizhong Wang, Shane Arora, David Atkinson, Russell Authur, Khyathi Chandu, Arman Cohan, Jennifer Dumas, Yanai Elazar, Yuling Gu, Jack Hessel, and 22 others. 2024. Olmo: Accelerating the science of language models. *Preprint*.

Hiroki Iida and Naoaki Okazaki. 2021. Incorporating semantic textual similarity and lexical matching for information retrieval. In *Proceedings of the 35th Pacific Asia Conference on Language, Information and Computation*, pages 582–591.

Zixia Jia, Zhaohui Yan, Wenjuan Han, Zilong Zheng, and Kewei Tu. 2023. Modeling instance interactions for joint information extraction with neural high-order conditional random field. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (ACL)*.

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

Qiao Jin, Bhuwan Dhingra, Zhengping Liu, William W Cohen, and Xinghua Lu. 2019. Pubmedqa: A dataset for biomedical research question answering. *arXiv preprint arXiv:1909.06146*.

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. *arXiv preprint arXiv:2405.17428*.

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, Haoran Xie, and Qing Li. 2021. Merging statistical feature via adaptive gate for improved text classification. In *Proceedings of the AAAI conference on artificial intelligence*, volume 35, pages 13288–13296.

Zehan Li, Xin Zhang, Yanzhao Zhang, Dingkun Long, Pengjun Xie, and Meishan Zhang. 2023. Towards general text embeddings with multi-stage contrastive learning. *arXiv preprint arXiv:2308.03281*.

Marco Lippi, Przemysław Pałka, Giuseppe Contissa, Francesca Lagioia, Hans-Wolfgang Micklitz, Giovanni Sartor, and Paolo Torroni. 2019. Claudette: an automated detector of potentially unfair clauses in online terms of service. *Artificial Intelligence and Law*, 27:117–139.

Dong C Liu and Jorge Nocedal. 1989. On the limited memory bfgs method for large scale optimization. *Mathematical programming*, 45(1):503–528.

Macedo Maia, Siegfried Handschuh, André Freitas, Brian Davis, Ross McDermott, Manel Zarrouk, and Alexandra Balahur. 2018. Www'18 open challenge: financial opinion mining and question answering. In *Companion proceedings of the the web conference 2018*, pages 1941–1942.

Pekka Malo, Ankur Sinha, Pekka Korhonen, Jyrki Wallenius, and Pyry Takala. 2014. Good debt or bad debt: Detecting semantic orientations in economic texts. *Journal of the Association for Information Science and Technology*, 65(4):782–796.

Clara H McCreery, Namit Katariya, Anitha Kannan, Manish Chablani, and Xavier Amatriain. 2020. Effective transfer learning for identifying similar questions: matching user questions to covid-19 faqs. In *Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 3458–3465.

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

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, and 1 others. 2019. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9.

Erik F Sang and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: Language-independent named entity recognition. *arXiv preprint cs/0306050*.

Harold J Spaeth, Lee Epstein, Andrew D Martin, Jeffrey A Segal, Theodore J Ruger, and Sara C Benesh. 2013. Supreme court database, version 2013 release 01. *Database at http://supremecourtdatabase. org*.

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

Olivier Taboureau, Sonny Kim Nielsen, Karine Audouze, Nils Weinhold, Daniel Edsgärd, Francisco S Roque, Irene Kouskoumvekaki, Alina Bora, Ramona Curpan, Thomas Skøt Jensen, and 1 others. 2010. Chemprot: a disease chemical biology database. *Nucleic acids research*, 39(suppl_1):D367–D372.

Yixuan Tang and Yi Yang. 2024. Do we need domain-specific embedding models? an empirical investigation. *arXiv preprint arXiv:2409.18511*.

Liang Wang, Nan Yang, Xiaolong Huang, Linjun Yang, Rangan Majumder, and Furu Wei. Improving text embeddings with large language models.

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

Renxi Wang, Xudong Han, Lei Ji, Shu Wang, Timothy Baldwin, and Haonan Li. 2024. Toolgen: Unified tool retrieval and calling via generation. *arXiv preprint arXiv:2410.03439*.

Guangxuan Xiao, Yuandong Tian, Beidi Chen, Song Han, and Mike Lewis. 2023. Efficient streaming language models with attention sinks. *arXiv preprint arXiv:2309.17453*.

Peiyuan Zhang, Guangtao Zeng, Tianduo Wang, and Wei Lu. 2024. Tinyllama: An open-source small language model. *arXiv preprint arXiv:2401.02385*.

Vitalii Zhelezniak, Aleksandar Savkov, April Shen, and Nils Y Hammerla. 2019. Correlation coefficients and semantic textual similarity. *arXiv preprint arXiv:1905.07790*.

A Introduction to Domain-Specific Tasks

Medicine Domain PubMedQA (Jin et al., 2019) is a biomedical question-answering dataset that requires models to answer research questions using corresponding PubMed abstracts, with answers limited to *yes*, *no*, or *maybe*. ChemProt (Taboureau et al., 2010) focuses on identifying chemical-protein interactions from biomedical texts. MQP (McCreery et al., 2020) contains pairs of medical

questions to evaluate the similarity between medical queries. RCT (Dernoncourt and Lee, 2017) involves classifying sentences in biomedical abstracts into categories such as background, objective, methods, results, and conclusions. In medicine domain, we use MedicalBERT[5] as the small domain-specific encoder model and BioMed-LLaMa[6] as the large domain-specific decoder model. We evaluated the performance of the models on the PubMedQA, ChemProt and RCT tasks using the F1 score as the metric. For the MQP task, we used accuracy as the metric.

Finance Domain FPB (Malo et al., 2014) contains financial phrasebank data and is designed to assess the classification of financial sentiments in economic texts. FiQA-SA (Maia et al., 2018) also focuses on financial sentiment analysis, requiring models to evaluate opinion mining and classify sentiments in financial contexts. NER (Alvarado et al., 2015) involves named entity recognition in financial texts, requiring the identification of key entities such as companies, financial instruments, and dates. In the finance domain, we use FinBert (Araci, 2019) as the small domain-specific encoder model and Finma-7B-nlp[7] as the large domain-specific decoder model. We adopt the F1 score as the evaluation metric in all three tasks.

Law Domain SCOTUS (Spaeth et al., 2013) comprises questions and answers related to Supreme Court cases, requiring models to classify legal concepts and reason about case outcomes. UNFAIR-ToS (Lippi et al., 2019) centers on detecting unfair terms in service agreements, aiming to identify problematic legal clauses. We use InLegalBERT (Chalkidis et al., 2020) as the small domain-specific model and LLaMA-Lawyer[8] as the large domain-specific decoder model. For the SCOTUS dataset, we use micro-F1 and macro-F1 as evaluation metrics, while for the UNFAIRToS dataset, we use weighted-F1.

B Tasks and Corresponding Instructions in the MTEB Subset

The specific tasks and corresponding instructions in the selected MTEB Subset are shown in Table 3. We adopted the same instructions as those used in (Wang et al., 2023), as they demonstrated better performance.

[5]https://huggingface.co/achDev/medicalBert

[6]https://huggingface.co/ContactDoctor/Bio-Medical-Llama-3-8B

[7]https://huggingface.co/TheFinAI/finma-7b-nlp

[8]https://huggingface.co/StevenChen16/llama3-8b-Lawyer

C Possible Reasons for the Shifting in Attention Sink

The effect of the "attention sink shift" phenomenon are analogous to those of the "attention sink" itself. In the original paper that first introduced the concept of the attention sink, it was noted that the sink token is not related to its actual meaning but rather to its position as the first token. Under relative position encoding, the attention result between the i-th token and the j-th token depends on their values and the difference in their positions. This relationship can be mathematically expressed as:

$$\langle f_q(x_m, m), f_k(x_n, n)\rangle = g(x_m, x_n, m-n)$$

When the first token is masked, the computation results for all other tokens become identical to those obtained by shifting the position encoding backward by 1. In other words:

$$\langle f_q(x_m, m), f_k(x_n, n)\rangle = \langle f_q(x_m, m-1), f_k(x_n, n-1)\rangle$$

Thus, the second token effectively becomes the "first token" in the eyes of the other tokens. Consequently, the second token becomes the sink token and acquires the most attention. This shift also reduces the representation ability of models, resulting in unsatisfactory performance in the MASK0-ALL configuration. As for why masking partial sink tokens does not lead to the reappearance of the attention sink, there is currently no related theoretical analysis. We speculate that this is because each token, through interactions with other tokens in the initial layers, gains some information about its own position. This acquired positional information may prevent the re-emergence of the attention sink when partial tokens are masked.

D Formalization

For each attention layer in the Transformer, the attention score computation is given by:

$$\text{AttnLLM}_i(Q, K, V) = \text{SoftMax}\left(\frac{QK^T}{\sqrt{d}} + M\right)V$$

where AttnLLM_i is the i-th head of multi-head self-attention (Vaswani et al., 2017) in the LLM.

Task Name	Instruction
ArguAna	Given a claim, find documents that refute the claim.
NFCorpus	Given a question, retrieve relevant documents that best answer the question.
SciFact	Given a scientific claim, retrieve documents that support or refute the claim.
SciDocsRR	Given a title of a scientific paper, retrieve the titles of other relevant papers.
StackOverflowDupQuestions	Retrieve duplicate questions from StackOverflow forum.
BiorxivClusteringS2S	Identify the main category of Biorxiv papers based on the titles.
MedrxivClusteringS2S	Identify the main category of Medrxiv papers based on the titles.
SprintDuplicateQuestions	Retrieve duplicate questions from Sprint forum.
Banking77Classification	Given an online banking query, find the corresponding intents.
EmotionClassification	Classify the emotion expressed in the given Twitter message into one of the six emotions: anger, fear, joy, love, sadness, and surprise.
MassiveIntentClassification	Given a user utterance as query, find the user intents.
STS*	Retrieve semantically similar text.

Table 3: Instructions used for evaluation on the MTEB benchmark. "STS*" indicates we use the same instructions for all the STS tasks.

The query Q, key K, and value V are computed as:

$$Q = W_q x + b, \quad K = W_k x + b, \quad V = W_v x + b$$

Here, M represents the mask. For the forward layer with causal attention mask:

$$M_{FWD} = \begin{bmatrix} 0 & -\infty & -\infty & \dots & -\infty \\ 0 & 0 & -\infty & \dots & -\infty \\ 0 & 0 & 0 & \dots & -\infty \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & 0 & \dots & 0 \end{bmatrix}$$

For the backward layer, where each token can only see itself and subsequent tokens:

$$M_{BWD} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ -\infty & 0 & 0 & \dots & 0 \\ -\infty & -\infty & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\infty & -\infty & -\infty & \dots & 0 \end{bmatrix}$$

For the bidirectional layer, M is a zero matrix. For the MASK0-BIDIR setting:

$$M_{MASK0} = \begin{bmatrix} 0 & 0 & 0 & \dots & 0 \\ -\infty & 0 & 0 & \dots & 0 \\ -\infty & 0 & 0 & \dots & 0 \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ -\infty & 0 & 0 & \dots & 0 \end{bmatrix}$$

The forward pass of the model is illustrated in Algorithm 1.

E Difference between Token-level Tasks and Sentence-level Tasks

Token-level and sentence-level tasks differ in their objectives, leading to distinct performance characteristics under various settings.

For token-level tasks, such as Named Entity Recognition (NER), the primary goal is to first obtain robust representations for each token that capture their individual meanings, which allows for accurate classification of simple tokens. Subsequently, leveraging contextual information helps resolve ambiguities for tokens that have multiple meanings, leading to more precise classification. Therefore, for token-level tasks, the BACK setting, which focuses each token on itself, provides a significant performance boost compared to the BIDIR setting. The MASK0 approach further enhances this by incorporating preceding context, thereby improving the model's performance even more.

In contrast, sentence-level tasks do not necessarily require each token to have a representation that captures its individual meaning. Instead, as long as each token incorporates contextual information, the resulting averaged representation can effectively reflect the overall content of the sentence. Consequently, the BIDIR setting, which allows each token to access both preceding and following context, generally performs better than BACK for sentence-level tasks. The MASK0 approach, by masking the first token (which has no practical semantic information and can cause the attention sink phenomenon), further refines the representation obtained from BIDIR, resulting in even better performance.

F Extended Visualization of Attention Scores

F.1 Attention Patterns in Forward Layers

Here, we present the attention scores of the original standard TinyLlama-1.1B across more layers, as shown in Figure 8. It can be observed that the

Algorithm 1 Model Forward Pass

```
def forward(self, x):
  # Input: x = (x1, x2, ..., xT)
  # Output: y = (y1, y2, ..., yT)
  h = self.embedding(x)                                  ▷ Embed input sequence
for i in range(self.num_layers) do
   mask = self.get_mask(i)                               ▷ Select mask based on layer ID
   h_attn = self.transformer_layers[i].attn(h, mask)     ▷ Attention layer
   h_ffn = self.transformer_layers[i].ffn(h_attn)        ▷ Feed-forward layer
   h = h + h_ffn if is_residual else h_ffn               ▷ Residual connection
end for
  y = self.output_layer(h)                               ▷ Generate output sequence
  return y
```

model primarily undergoes three phases: In the first three layers, the attention scores are relatively uniform. From the fourth to the ninth layer, the attention scores predominantly focus on the first token. Starting from the tenth layer, this concentration eases somewhat, yet by the final layer, the scores remain relatively concentrated.

F.2 Shifting of Attention Sink

We further illustrate the attention scores when more tokens are masked, as shown in the first two rows of Figure 9. It can be observed that the model consistently learns a new attention sink at the fourth layer. In the last row, we present different *Sink Masks*, namely *MASK0*, *MASK3*, *MASK5*, and *MASK7*. These sink masks can effectively suppress the emergence of attention sink phenomena. The *Reverse Causal Mask* can be regarded as a special type of *Sink Mask*, which in this case is denoted as *MASK9*.

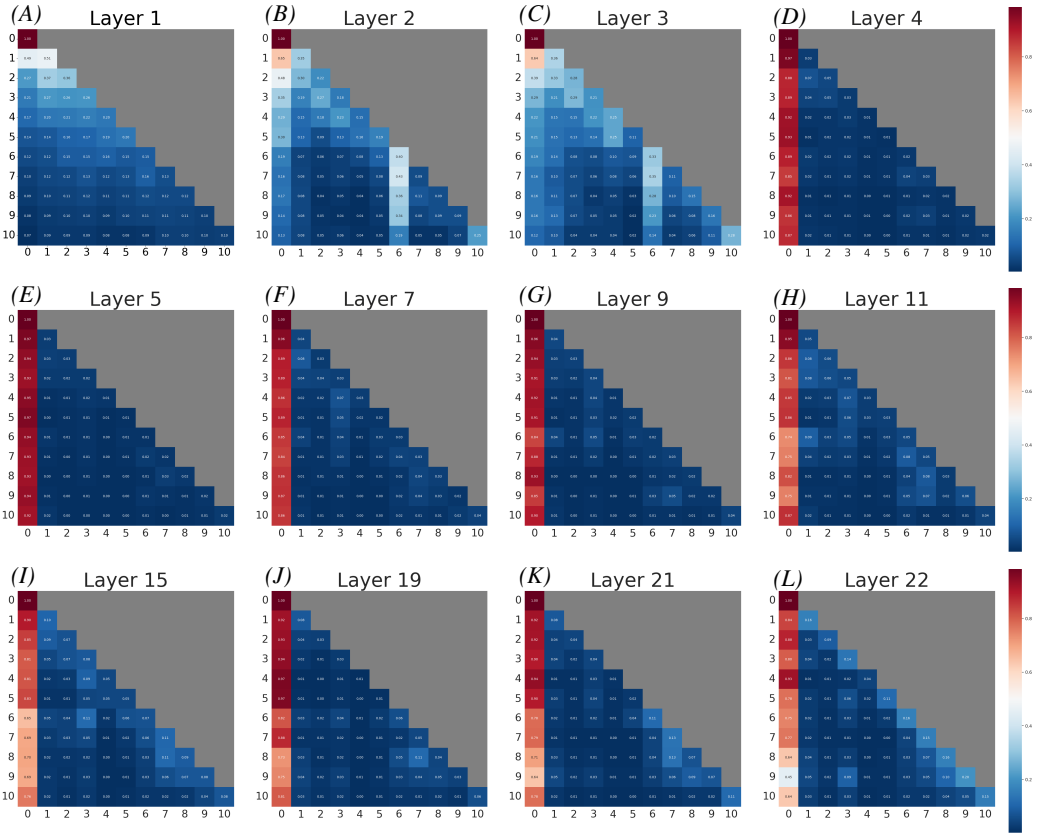

Figure 8: Visualization of Attention Scores in Different Layers of the Original Standard TinyLlama-1.1B Model

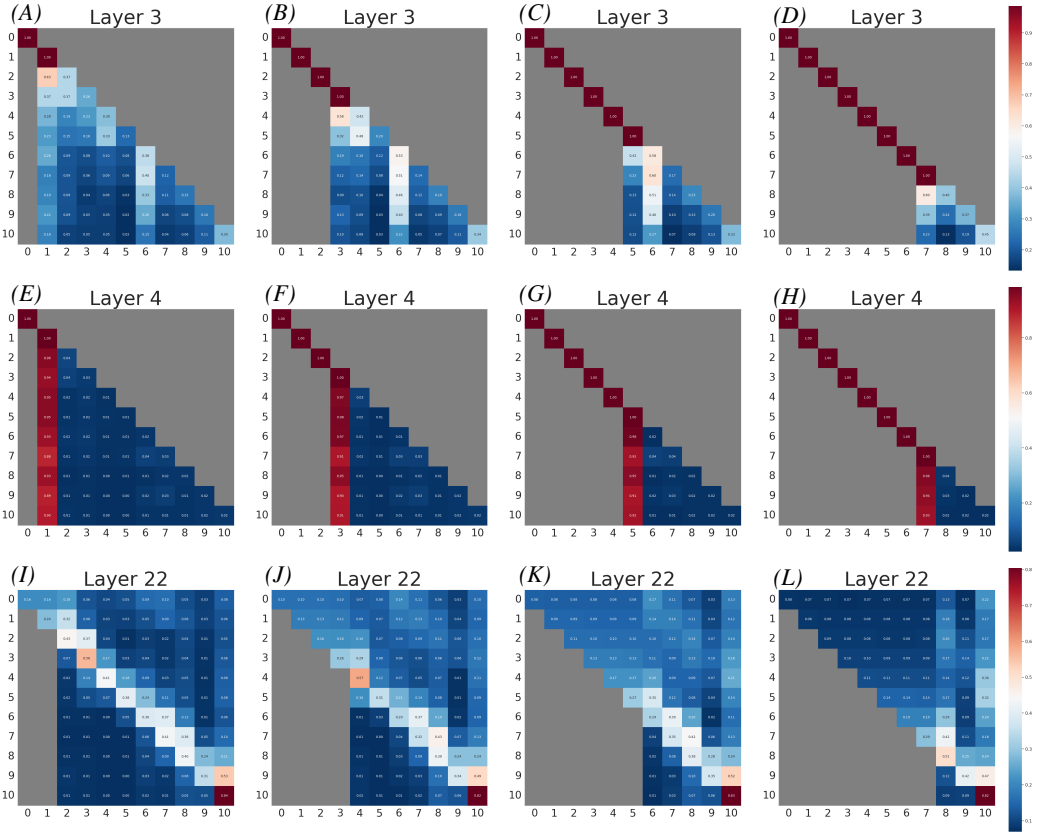

Figure 9: Visualization of Attention Scores with More Tokens Masked Different Sink Mask Variants