LogitRouter: a novel Attention variant for reducing Myopic Routing in Mixture of Experts

Felipe Rodriguez[1], **Marcelo Mendoza**[1,2] *
[1]School of Engineering, Pontificia Universidad Católica de Chile
[2]National Center for Artificial Intelligence (CENIA)

Abstract

Mixture of Experts (MoEs) have emerged as strong alternatives to traditional transformers, offering significant advantages in terms of training and inference efficiency. At the core of this architecture lies the router, responsible for selecting which experts are activated for each token. However, despite these advances, routing mechanisms continue to face stability challenges that the basic architecture fails to fully address. One such issue is Myopic Routing, where each token determines its route independently, without considering the routing decisions made for other tokens. To address this limitation, the LogitAttention mechanism is introduced—a variant of traditional attention—and, building upon it, the LogitRouter, a novel routing architecture that incorporates contextual information about the routing of other tokens. Due to budget constraints, a set of simple experiments is designed to obtain preliminary evidence of performance trends. These experiments are empirically validated on established benchmarks such as BoolQ, MMLU, and ARC. Finally, the work concludes with an in-depth discussion of architectural variants, applicability, limitations, and future directions, which aims to support continued research in this area.

1 Introduction

Large Language Models (LLMs) (Radford et al., 2019; Brown et al., 2020) have represented one of the most significant breakthroughs in Artificial Intelligence in recent years. Since the introduction of the Transformer architecture (Vaswani et al., 2017), LLMs have continuously improved in both performance and capabilities, largely driven by their remarkable scaling laws (Kaplan et al., 2020; Hoffmann et al., 2022). However, these advances have come at the cost of substantial energy and computational requirements (Bender et al., 2021), which has prompted the search for more efficient architectures capable of achieving results comparable to those of traditional Transformers.

In this context, Mixture of Experts (MoE) (Jacobs et al., 1991; Jordan and Jacobs, 1994; Shazeer et al., 2017) has recently emerged as one of the most prominent architectures, serving as the foundation for many current state-of-the-art models (Jiang et al., 2024; Liu et al., 2024; Guo et al., 2025; Yang et al., 2025). The main advantage of MoE lies in its ability to reduce overall computational cost—during both training and inference—without causing a significant drop in final performance. This is achieved by replacing the traditional Feed-Forward Network (FFN) sublayers, which typically hold the majority of a model's parameters, with a Sparse MoE Layer. A Sparse MoE Layer (or simply MoE Layer) is composed of multiple smaller FFN sublayers that share the same architecture (typically SwiGLU (Shazeer, 2020)) and are referred to as experts. These experts operate in parallel. To avoid activating all the parameters in the layer, a specialized component called the Router is introduced. The Router selectively activates and assigns weights to only a subset of experts, leaving the remaining ones unused.

Since the early works in this area (Fedus et al., 2022; Lepikhin et al., 2020; Du et al., 2022), the conventional design of the Router consists of a linear sub-layer, followed by a Top-k selection function that activates only the k experts with the highest logit values. A softmax activation is then applied to ensure that the final weights follow a smooth probability distribution. This approach reduces the total computational load and, consequently, the energy and financial costs of running the network (Lepikhin et al., 2020; Du et al., 2022; Dai et al., 2024), at the expense of a minor (and often negligible) drop in performance.

This setup naturally gives rise to a trade-off between efficiency and performance: scaling laws

* Corresponding author: marcelo.mendoza@uc.cl

Proceedings of the 18th International Natural Language Generation Conference, pages 499–510
October 29–November 2, 2025.

indicate that activating more parameters typically leads to improved outcomes, which means that Mixture of Experts (MoE) models aim to find a balance point where further gains in performance do not justify the associated increase in cost (Krajewski et al., 2024; Abnar et al., 2025). This balance is precisely the responsibility of the Router: given a pool of experts, it decides which ones to activate and which to skip, seeking the optimal compromise between performance and efficiency.
Nevertheless, the standard Router architecture still relies on a relatively basic formulation. While it has proven effective, it is far from definitive. As a result, a significant portion of current MoE research focuses on designing better Routers, as this component is not only central to the architecture but also the only element that represents a true innovation relative to standard transformers.

Literature gap. In this context, the problem known as Myopic Routing is identified. This issue arises when tokens are routed without considering the routing decisions of other tokens. To address this limitation, the LogitAttention mechanism is introduced, along with the LogitRouter architecture—a novel type of Router that mitigates this problem by learning to update routing decisions a priori using contextual information derived from the routing of other tokens. This architecture is evaluated through fine-tuning on the SFT models of OLMoE (Muennighoff et al., 2024), and the resulting models are subsequently compared against the original ones across various benchmarks.

Contributions. The contributions of this work can be summarised as follows:

1. The identification of the Myopic Routing problem in MoE routers.

2. The introduction of LogitAttention, a novel variant of the standard attention mechanism.

3. The proposal of LogitRouter, a new router architecture that leverages contextual information from the a priori routing decisions of other tokens to update its own.

4. An extensive discussion of the results, architectural variants, scope, limitations, and directions for future research, aimed at encouraging continued work in this area.

The structure of this work is as follows: Section 2 discusses the limitations of current MoE architectures, the solutions proposed in the literature, and the motivation behind the present proposal. Next, Section 3 introduces LogitAttention, LogitRouter, and their respective variants. Section 4 presents the experimental setup, while the results and their analysis are detailed in Section 5. Following this, Section 6 outlines potential directions and methods for extending and completing the research. Finally, Section 7 provides a concise summary of the overall contributions of the work.

2 Related work

Since GShard (Lepikhin et al., 2020) first applied Mixture of Experts (MoE) to Transformers, numerous routing architectures have been proposed (Cai et al., 2024; Mu and Lin, 2025; Dimitri et al., 2025), most of which have been evaluated primarily through empirical means. However, it was GLaM (Du et al., 2022) that first demonstrated MoE's ability to outperform traditional Transformers on more complex tasks. Still, this promising architecture soon revealed several training instabilities.

Load Balancing Loss. One challenge inherited from early LSTMs (Hochreiter and Schmidhuber, 1997) is the Load Balancing Loss (Shazeer et al., 2017), where some experts are over-utilized while others are underused, leading to suboptimal performance for the latter and ultimately for the model as a whole. GShard (Lepikhin et al., 2020) addressed this issue by introducing an auxiliary loss function—Auxiliary Load Balancing Loss—which remains widely used today.
Another effective approach was introduced in Switch Transformers (Fedus et al., 2022), which challenged the then-common assumption that at least two experts must be activated to enable meaningful performance comparison. Instead, they adopted a Top-1 gating function while significantly increasing the number of experts, achieving better load distribution. Another notable proposal is Expert Choice Routing (Zhou et al., 2022), which assigns a fixed number of tokens to each expert, rather than assigning a fixed number of experts to each token.

Representation Collapse. One of the earliest empirically observed problems was the well-known Representation Collapse (Chi et al., 2022), where experts fail to utilise their full representational capacity. To mitigate this, the Router Z-Loss auxiliary loss function was introduced (Zoph et al., 2022), and it remains in use across various models.

Dynamic Routing. Beyond empirical observations, several theoretical issues have also been identified. A key concern is that all tokens typically activate the same number of experts, and thus the same number of parameters, contradicting the intuition that different tokens may require different levels of computational effort to reach optimal predictions. Several strategies have been proposed to dynamically allocate the number of experts per token:

- AdaMoE (Zeng et al., 2024) introduces null experts that consume no computational resources. By increasing the value of k, the router can choose varying numbers of null experts, resulting in a token-dependent compute load.
- ReMoE (Wang et al., 2024) replaces the standard Top-k selection with a ReLU activation function (Nair and Hinton, 2010), allowing for a variable number of experts per token and enabling more precise backpropagation (Rumelhart et al., 1985, 1986) due to its differentiability.
- Dynamic MoE (Huang et al., 2024) proposes a Top-p gating mechanism, which selects experts whose combined activation probabilities exceed a threshold p.
- TC-MoE (Yan et al.) adopts ternary weighting per expert, allowing each expert to be weighted by -1, 0, or 1. This enables the participation of experts that may not have contributed positively in prior steps.

Routing Fluctuation. Dynamic routing, while promising, introduces new challenges (Shi et al., 2024). One such issue is routing fluctuation (Su et al., 2024), where tokens repeatedly select different experts throughout training. As a result, many experts are updated at each step, but only a subset is used during inference, leading to poor optimization (Dai et al., 2022; Nguyen et al., 2024). To address this, (S)MoE (Nguyen et al., 2025) leverages token relationships captured through attention mechanisms to inform expert selection more effectively.

However, unlike the aforementioned approaches that mainly focus on the gating mechanism or expert configurations, some recent proposals explore alternative sublayer architectures:

- Yuan 2.0 M-32 (Wu et al., 2024) introduces an Attention Router, which uses the classical attention mechanism to route each token to its corresponding expert.
- Mixture of Routers (Zhang et al., 2025) presents a router-of-routers approach, in which multiple parallel routers propose expert assignments, and a main router selects which routers to activate based on their outputs.
- Another method (Pedicir et al., 2024) employs a Recurrent Neural Network (RNN) (Elman, 1990) as a router, updating token routes based on prior routing information.

Myopic Routing. There is, however, another important problem that has yet to be explicitly identified in the literature: tokens make routing decisions independently, without considering the routing decisions of other tokens. A related idea appears in PMoE (Jung and Kim, 2024), where additional information is incrementally added to routers across layers, although in a different context. Yet none of the aforementioned studies directly addresses this specific issue. Even in cases where cross-token information is considered, it never includes routing paths—only token content itself. Still, it's reasonable to expect situations where the model behaves inconsistently because different tokens activate different experts that interpret them divergently. For instance, two tokens that should activate the same expert might activate different ones (or vice versa), impairing coherence. A potential solution could involve enabling routers to access the prior routing decisions of other tokens and using that information to update each token's routing path. Intuitively, this challenge might best be addressed by rethinking the traditional linear sublayer architecture, where incorporating cross-token routing information feels more natural.

3 LogitRouter

3.1 LogitAttention

LogitAttention is an architecture that implements a generalisation of the attention mechanism, causally designed to incorporate into each token contextual information derived from the prior logits of the other tokens within the context window.

Mathematical Formulation. LogitAttention consists of four main tensors (or matrices, for simplicity), which operate—either directly or indirectly—on each token x:

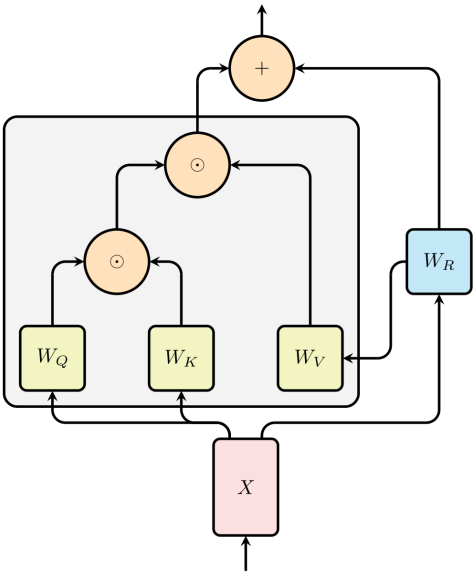

Figure 1: **LogitAttention** mechanism: Queries Q and Keys K operate in the same way as in standard attention. The differences are: (i) the Values V are derived from R, and (ii) the residual connection is applied to R instead of x.

1. First, the matrix W_R maps the token x to its a priori routing representation R. This matrix corresponds to the standard linear sublayer used in routers, meaning that R also represents the conventional logits.

2. Next, following the structure of the standard attention mechanism, matrix W_Q transforms the tokens into Queries Q.

3. Similarly, matrix W_K converts the tokens into Keys K.

4. Finally, there is the matrix W_V, which maps the routings R into Values V. This represents the main innovation of the architecture, as traditionally, Values are computed directly from the tokens. However, this modification allows the contextual information to be derived not from the tokens themselves (which have just been processed in the preceding causal attention mechanism), but from their a priori logits.

As illustrated in Figure 1, the Queries, Keys, and Values computed through the matrices described above are processed in the same way as in standard attention, including the use of a residual connection (He et al., 2016). The design is driven by two main motivations: first, it enables the information exchange to update the logits rather than constituting them directly; and second, it ensures that the updated information pertains to the logits instead of the tokens themselves. Consequently, in this case, the residual connection is computed with respect to R rather than x.

Therefore, by incorporating the standard attention normalization (Vaswani et al., 2017), LogitAttention can be mathematically expressed as:

$$\text{LogitAttention(x):=}$$
$$\text{W}_R(x) + \sigma\left(\frac{1}{\sqrt{d_k}} W_Q(x) \cdot W_K(x)^\top\right) \cdot W_V(W_R(x))$$

where x has dimensionality $[t \times d_{model}]$, W_Q and W_K are matrices of size $[d_{model} \times d_k]$, W_R is a matrix of size $[d_{model} \times n]$, W_V is a matrix of size $[n \times d_k]$, and σ represents the softmax function. Here, d_{model} denotes the embedding dimensionality of the tokens in the model, $d_k \leq d_{model}$ refers to the internal embedding dimensionality used by the mechanism, n is the total number of experts, and t is the sequence length.

Properties. An important observation is that for the residual connection to operate correctly in terms of matrix dimensionality, it is necessary to define $d_k := n$. Moreover, if we also set $n := d_{model}$ and choose $W_R := I_{n \times n}$, the design reduces to the classical single-headed attention mechanism. This implies that the key elements of the proposed architecture are precisely the number of experts n and the matrix W_R. Therefore, the novelty of this design lies in how it contextualises the behaviour of the classical router. It is worth noting that this same architecture is applicable to any layer that performs routing and requires interaction across the logits, such as the final unembedding layer (Press and Wolf, 2016). This motivates the name LogitAttention, and consequently, LogitRouter.

3.2 LogitRouter

The LogitAttention Router, or simply LogitRouter, is a router architecture that employs LogitAttention as its core mechanism.

Mathematical formulation. LogitRouter additionally incorporates a fifth matrix, W_L, of dimension $[n \times n]$, which transforms the output of LogitAttention into the final logits L. The theoretical motivation for this sublayer arises from the observation that, within a transformer block, attention alone is not sufficient; it must be followed by a feed-forward network (FFN). In this analogy, LogitAttention—comprising W_R, W_Q, W_K, and W_V—corresponds to the standard attention mecha-

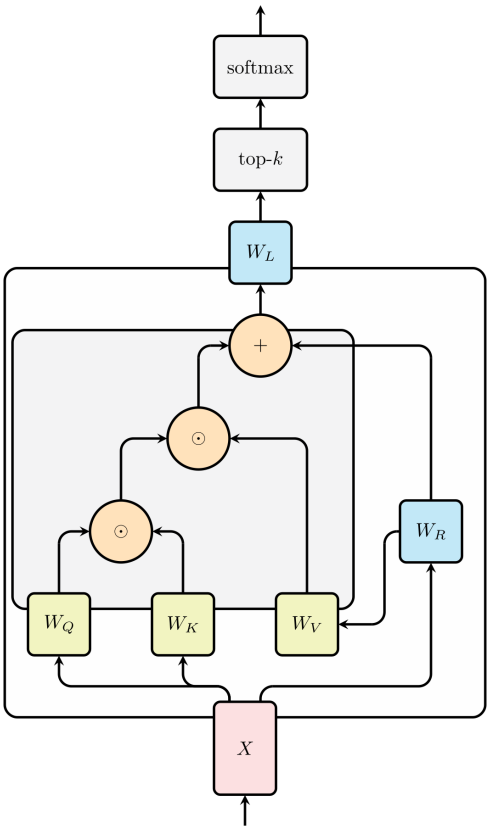

Figure 2: **LogitRouter**: The final logits L are produced via the matrix W_L, and the underlying architecture is based on LogitAttention. The remaining components are identical to those in standard routers.

nism, while W_L plays a role analogous to that of the FFN.

Then, LogitRouter follows the mathematical form given below:

$$\text{LogitRouter(x)} := \sigma\Big(\text{top-}k(W_L(\text{LogitAttention}(x)))\Big)$$

Properties. This architecture possesses the important property of being a generalisation of the classical routing mechanism. Specifically, the classical behaviour is recovered by setting $W_L := I_n$ and $W_V := 0_{n\times n}$. This characteristic is crucial for enabling the experiments described in section 4, as it allows the model's weights to be initialised with those of a classical router, ensuring that its behaviour matches the original one exactly. Moreover, LogitAttention can be seen as a generalisation of the traditional Attention mechanism, and LogitRouter as a generalisation of the Router. This architecture thus highlights the fact that Router and Attention exhibit analogous properties and, as a result, share underlying characteristics (Wu and Wong).

Intuition Behind the Architecture. Intuitively, LogitRouter enables each token to access prior information about the routing of other tokens and attend to them, thereby updating its own routing decisions in an attempt to achieve internal consistency with the group. The queries and keys encode the identity of each token, such that their dot product highlights connections between tokens considered mutually significant. In contrast, the values are computed based on routing information, ensuring that the actual content being exchanged reflects the routing paths rather than the tokens' intrinsic representations.

This design allows each expert to learn how to route coherent groups of tokens together, or alternatively, to prevent semantically similar or complementary tokens from being processed by experts that interpret them differently.

This leads to a fundamental distinction between the contextualisation of the token itself—accomplished in the preceding attention layer—and the contextualisation of its routing, which takes place within the router of the MoE layer. The former refines the identity of each token, while the latter determines how and for what purpose the token should be utilised.

Therefore, it is hypothesised that this architecture will have a greater impact on the context and discussion (Shojaee et al., 2025) surrounding reasoning models (Wei et al., 2022; Kojima et al., 2022) within the Test-Time Compute paradigm (Sun et al., 2020; Ji et al., 2025), as it may allow more efficient use of the context window of each prompt to generate next-token predictions. Its impact could be even more significant in the case of DLMs (Li et al., 2022; Nie et al., 2025), since tokens are not generated sequentially nor fixed once produced. Instead, they are updated iteratively, which makes it possible to leverage these updates along the routing paths.

3.3 Variants

The LogitRouter architecture involves several design choices that, while seemingly arbitrary, were based on a low-level experiment. This experiment consisted of training a small transformer from scratch on the Tiny Shakespeare dataset (Karpathy, 2015), using model loss as the evaluation metric. However, these design decisions may not scale con-

sistently to larger models and datasets. For this reason, it may be worthwhile to reconsider and modify them:

- The final linear sublayer W_L could be removed or placed before the LogitAttention mechanism. The idea of mirroring the structure of a standard transformer block arose from theoretical considerations and was empirically tested in the low-level experiment, although it did not yield significantly better results compared to other alternatives. Furthermore, one can observe that LogitAttention follows directly after the attention block, while the experts follow immediately after the W_L sublayer. This results in two consecutive attention blocks and then two consecutive linear layers. Placing W_L before the attention mechanism could resolve both issues.

- The operation combining the a priori routing R with the output of LogitAttention need not necessarily be addition. An alternative tested in the low-level experiment was the Hadamard product, which yielded comparable, though slightly inferior, results.

- It is possible to use a value of $d_k \neq n$ in order to enforce that the representations from R act as true a priori logits. However, this would require defining a new matrix to ensure dimensional compatibility for residual computations.

- Although LayerNorm (Ba et al., 2016) is typically considered a distinct layer within a transformer, this architecture could be extended to incorporate it. The LayerNorm sublayer could be placed either before or after the mechanism (Xiong et al., 2020).

- This design can also be combined with other previously discussed architectures. For instance, one could use a Mixture of Routers where the main router is of the LogitRouter type, or replace the top-k function with a top-p or ReLU-based function.

4 Experimental setup

To ensure a degree of robustness in the results, two models were selected as the basis of the experiments: `https://huggingface.co/allenai/OLMoE-1B-7B-0924-SFT` and `https://huggingface.co/allenai/OLMoE-1B-7B-0125-SFT` (Muennighoff et al., 2024). These are, to date, the only competitive Mixture of Experts (MoE) models small enough (around 7B parameters) to allow for low-budget training, while providing full access to their weights, training data, and evaluation details. For simplicity, we refer to them as ***OLMoE-0924*** and ***OLMoE-0125***.

The reason for choosing their Supervised Fine-Tuning (SFT (Ouyang et al., 2022)) versions is that the SFT datasets used in both models (`https://huggingface.co/datasets/allenai/tulu-v3.1-mix-preview-4096-OLMoE` and `https://huggingface.co/datasets/allenai/tulu-3-sft-olmo-2-mixture`, respectively) are significantly smaller than their pretraining datasets. As a result, fine-tuning is faster and more cost-efficient.

Based on this, the following experiments were conducted:

LogitRouter Models. First, starting from the *OLMoE-0924* and *OLMoE-0125* checkpoints, we replaced their routers with LogitRouter modules. We then performed fine-tuning on both models using their original SFT datasets, training only the router parameters while keeping all other parameters frozen. The fine-tuning procedure followed the hyperparameters described in the original paper (Muennighoff et al., 2024), with the following exceptions:

- The batch size was set to 6, the largest power of two that fit on the machine used (a single-node H200 GPU). No gradient accumulation steps were employed.

- Training was limited to a single epoch instead of two. Given that the models are already pretrained, it was deemed reasonable to assume that one epoch would suffice.

- We used HuggingFace's 'SFTTrainer' (Wolf et al., 2019; von Werra et al., 2020) with AdamW (Loshchilov and Hutter, 2017; Kingma, 2014) as the optimizer, instead of Open Instruct (Ivison et al., 2023; Wang et al., 2023).

- Parameter precision was set to BF16 (Kalamkar et al., 2019). While the OLMoE paper indicates that BF16 was used for SFT, it does not specify the precision used for

the routers. In practice, routers are typically trained with FP32 precision (Fedus et al., 2022).

Since LogitRouter is a generalisation of the routing architecture used in OLMoE (see 3.2), we initialized it as follows: $W_L := I_n$, $W_V := 0_{n \times n}$, and W_R was initialized using the original router weights of the base models, ensuring that the initial behaviour remained identical. The matrices W_Q and W_K were initialized using a standard truncated normal distribution (Devlin et al., 2019). The resulting models are referred to as ***OLMoE-0924-logit*** and ***OLMoE-0125-logit***, respectively.
However, this experimental setup introduces two distinct sources of variation: (i) the new router architecture, and (ii) the new post-training strategy involving fine-tuning on the original SFT datasets while freezing all non-router parameters.

Base Router Models. To isolate these factors, we conducted a second set of experiments applying only the post-training strategy, without modifying the routing architecture. In other words, using again *OLMoE-0924* and *OLMoE-0125* as base models, we repeated the fine-tuning process exactly as described above, but without replacing the original routers. To ensure a fair comparison under identical conditions, all hyperparameters were kept the same. These models are denoted as ***OLMoE-0924-base*** and ***OLMoE-0125-base***.

5 Results

The models were evaluated across three different benchmarks: ARC (including ARC-Challenge and ARC-Easy), MMLU (Hendrycks et al., 2020), and BoolQ (Clark et al., 2019), following the MCF strategy outlined in the OLMES protocol (Gu et al., 2024). Evaluations were conducted using varying numbers of few-shot examples (0, 1, 2, and 5), and the final performance metric for each model was defined as the maximum score achieved across these few-shot settings for ***OLMoE-0125-base***.
Two separate analyses were conducted: one to compare the performance of the original models with that of the *base* models, and another to compare the *base* models with the *logit* models. The first analysis aims to evaluate the impact of the previously defined post-training strategy on the final model, while the second assesses how changes in architecture affect performance. A summary of the results is presented in Table 1.

Post-training strategy. The rows corresponding to the *base* models show how their performance differs from that of the original models. Overall, it is evident that performance tends to degrade when these models undergo post-training using this strategy. In 5 out of 8 experiments, the post-trained models performed worse, and in 3 of those cases, the performance drop represented the largest absolute differences observed between the two versions. This suggests that there is no consistent improvement trend resulting from the post-training process.
This outcome is expected, as the router becomes decoupled from the rest of the network under this setup—a configuration that is not commonly used in practice. Several studies have shown that this decoupling may lead to network instability (Jiang et al., 2025; Panda et al., 2025). Moreover, it is important to emphasise that this particular experiment, although useful in this work for comparison purposes with the final models, does not constitute a genuine model optimisation procedure. Rather, it corresponds to an arbitrary post-training step. The only potential "improvement" could be attributed to training the model for an additional epoch. However, if the original models did not undergo this extra epoch, it is likely because, after careful hyperparameter tuning and analysis, two epochs were deemed optimal.
In fact, one can think of a well-trained and optimised LLM as having reached a local optimum in the search space (Elsken et al., 2019; He et al., 2021), meaning that any arbitrary modification is likely to degrade performance. Therefore, the fact that performance actually improved over the original model in 3 out of 8 benchmarks suggests that there are still better optima to be found. This also ensures that the subsequent comparison with LogitRouter is fair and cannot be solely attributed to a drop in performance caused by this variable.

LogitRouter architecture. The rows corresponding to the *logit* models display their performance differences relative to the *base* models. This comparison is particularly relevant because it isolates the architectural effect of the router on performance.
The results indicate a general trend toward improved performance, as only 2 out of the 8 experiments showed a performance drop—and in both cases, the decrease was minor. Thus, we can reasonably conclude that switching the router architecture to a LogitRouter produces a positive effect.

Table 1: Evaluations from the **classic fine-tuning experiment** on each benchmark. Percentage differences are computed as follows: (i) for *base* models, relative to the original versions; and (ii) for *logit* models, relative to their respective *base* versions. The Total Difference row compares *logit* models directly against the original ones.

Model Name	MMLU	ARC-C	ARC-E	BoolQ
OLMoE-0125	0.550	0.684	**0.859**	0.744
OLMoE-0125-base	0.551 (↑ 0.1%)	0.674 (↓ 1.0%)	0.849 (↓ 1.0%)	0.746 (↑ 0.2%)
OLMoE-0125-logit	**0.552** (↑ 0.1%)	**0.690** (↑ 1.6%)	0.849 (↑ 0.0%)	**0.752** (↑ 0.6%)
Total Difference	↑ 0.2%	↑ 0.6%	↓ 1.0%	↑ 0.8%
OLMoE-0924	**0.539**	0.654	0.835	**0.781**
OLMoE-0924-base	0.537 (↓ 0.2%)	0.653 (↓ 0.1%)	**0.841** (↑ 0.6%)	0.748 (↓ 3.3%)
OLMoE-0924-logit	0.533 (↓ 0.4%)	**0.660** (↑ 0.7%)	0.839 (↓ 0.2%)	0.754 (↑ 0.6%)
Total Difference	↓ 0.6%	↑ 0.6%	↑ 0.4%	↓ 2.7%

In fact, in more than half of the evaluations (5 out of 8), the *logit* models even outperformed the original models. For instance, in ARC-C, performance increased in both comparisons despite the initial performance drop observed in the *base* models.

6 Next steps

Given the constraints of a limited budget, most of the work remains to be completed. Therefore, all the steps that need to be followed are outlined below, in order to assess the actual impact of LogitRouter and the other ideas previously introduced.

- The model should either be trained from scratch using LogitRouter or undergo full-model fine-tuning. This is important to prevent the router from becoming decoupled from the rest of the model. This will allow us to assess the true potential of the architecture and determine whether it can achieve state-of-the-art performance on its own. As discussed in section 3.2, it is particularly promising to explore this architecture within the context of a reasoning MoE or a DLM.

- Another promising direction would be to distill LogitRouter into a conventional router, as suggested in several prior studies (Fedus et al., 2022; Kudugunta et al., 2021; Zuo et al., 2021; Kim et al., 2021), in order to reduce the parameter count of the routing layer.

- Finally, as discussed in section 3.1, a variant of this architecture could be tested in the final unembedding layer of an LLM or SLM (Gunasekar et al., 2023; Lu et al., 2024), given the large number of parameters typically associated with such a layer. In this context, W_R would serve as the standard unembedding matrix.

7 Conclusion

This work introduced LogitRouter, a novel routing architecture for Mixture of Experts, designed to incorporate contextual information into the token routing process. This approach addresses the newly identified Myopic Routing problem by enriching the routing decisions with broader contextual cues. Despite limitations in budget the results indicate a promising trend. In particular, there were notable improvements on benchmarks such as ARC-C compared to the baseline model, suggesting strong potential for this architecture in future developments. Additionally, the paper provides a thorough discussion of each stage of the research process—ranging from the proposal and design of variants to the experiments and results—and outlines several promising directions for future work.

Ethical considerations

This study does not involve experiments with humans or animals, nor does it include the handling of sensitive or confidential data related to individuals' privacy. The findings presented in this study are reliable. No artificial intelligence tools were used to generate this content or to support any stage of the research process, including the review of related work. The code and datasets employed in this paper are available in a GitHub repository; however, they have been omitted from the manuscript in order to comply with the double-blind review policy.

Acknowledgments

This work was supported by National Center for Artificial Intelligence CENIA FB210017 Basal

ANID, the Millennium Institute for Foundational Research on Data (ANID grant ICN17_002) and ANID FONDECYT grant 1241462.

References

Samira Abnar, Harshay Shah, Dan Busbridge, Alaaeldin Mohamed Elnouby Ali, Josh Susskind, and Vimal Thilak. 2025. Parameters vs flops: Scaling laws for optimal sparsity for mixture-of-experts language models. *arXiv preprint arXiv:2501.12370*.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. 2016. Layer normalization. *arXiv preprint arXiv:1607.06450*.

Emily M Bender, Timnit Gebru, Angelina McMillan-Major, and Shmargaret Shmitchell. 2021. On the dangers of stochastic parrots: Can language models be too big?. In *Proceedings of the 2021 ACM conference on fairness, accountability, and transparency*, pages 610–623.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. 2020. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901.

Weilin Cai, Juyong Jiang, Fan Wang, Jing Tang, Sunghun Kim, and Jiayi Huang. 2024. A survey on mixture of experts. *arXiv preprint arXiv:2407.06204*.

Zewen Chi, Li Dong, Shaohan Huang, Damai Dai, Shuming Ma, Barun Patra, Saksham Singhal, Payal Bajaj, Xia Song, Xian-Ling Mao, et al. 2022. On the representation collapse of sparse mixture of experts. *Advances in Neural Information Processing Systems*, 35:34600–34613.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. Boolq: Exploring the surprising difficulty of natural yes/no questions. *arXiv preprint arXiv:1905.10044*.

Damai Dai, Chengqi Deng, Chenggang Zhao, RX Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Yu Wu, et al. 2024. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. *arXiv preprint arXiv:2401.06066*.

Damai Dai, Li Dong, Shuming Ma, Bo Zheng, Zhifang Sui, Baobao Chang, and Furu Wei. 2022. Stablemoe: Stable routing strategy for mixture of experts. *arXiv preprint arXiv:2204.08396*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers)*, pages 4171–4186.

Vasily Dimitri, Barbara Regina, and Magdolna Alfonz. 2025. A survey on mixture of experts: Advancements, challenges, and future directions. *Authorea Preprints*.

Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, et al. 2022. Glam: Efficient scaling of language models with mixture-of-experts. In *International conference on machine learning*, pages 5547–5569. PMLR.

Jeffrey L Elman. 1990. Finding structure in time. *Cognitive science*, 14(2):179–211.

Thomas Elsken, Jan Hendrik Metzen, and Frank Hutter. 2019. Neural architecture search: A survey. *Journal of Machine Learning Research*, 20(55):1–21.

William Fedus, Barret Zoph, and Noam Shazeer. 2022. Switch transformers: Scaling to trillion parameter models with simple and efficient sparsity. *Journal of Machine Learning Research*, 23(120):1–39.

Yuling Gu, Oyvind Tafjord, Bailey Kuehl, Dany Haddad, Jesse Dodge, and Hannaneh Hajishirzi. 2024. Olmes: A standard for language model evaluations. *arXiv preprint arXiv:2406.08446*.

Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. 2023. Textbooks are all you need. *arXiv preprint arXiv:2306.11644*.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Ruoyu Zhang, Runxin Xu, Qihao Zhu, Shirong Ma, Peiyi Wang, Xiao Bi, et al. 2025. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778.

Xin He, Kaiyong Zhao, and Xiaowen Chu. 2021. Automl: A survey of the state-of-the-art. *Knowledge-based systems*, 212:106622.

Dan Hendrycks, Collin Burns, Steven Basart, Andy Zou, Mantas Mazeika, Dawn Song, and Jacob Steinhardt. 2020. Measuring massive multitask language understanding. *arXiv preprint arXiv:2009.03300*.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation*, 9(8):1735–1780.

Jordan Hoffmann, Sebastian Borgeaud, Arthur Mensch, Elena Buchatskaya, Trevor Cai, Eliza Rutherford, Diego de Las Casas, Lisa Anne Hendricks, Johannes Welbl, Aidan Clark, et al. 2022. Training compute-optimal large language models. *arXiv preprint arXiv:2203.15556*.

Quzhe Huang, Zhenwei An, Nan Zhuang, Mingxu Tao, Chen Zhang, Yang Jin, Kun Xu, Liwei Chen, Songfang Huang, and Yansong Feng. 2024. Harder tasks need more experts: Dynamic routing in moe models. *arXiv preprint arXiv:2403.07652*.

Hamish Ivison, Yizhong Wang, Valentina Pyatkin, Nathan Lambert, Matthew Peters, Pradeep Dasigi, Joel Jang, David Wadden, Noah A Smith, Iz Beltagy, et al. 2023. Camels in a changing climate: Enhancing lm adaptation with tulu 2. *arXiv preprint arXiv:2311.10702*.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural computation*, 3(1):79–87.

Yixin Ji, Juntao Li, Hai Ye, Kaixin Wu, Jia Xu, Linjian Mo, and Min Zhang. 2025. Test-time computing: from system-1 thinking to system-2 thinking. *arXiv preprint arXiv:2501.02497*.

Albert Q Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, et al. 2024. Mixtral of experts. *arXiv preprint arXiv:2401.04088*.

Wangyi Jiang, Yaojie Lu, Hongyu Lin, Xianpei Han, and Le Sun. 2025. Improved sparse upcycling for instruction tuning. In *Proceedings of the 31st International Conference on Computational Linguistics*, pages 9485–9498.

Michael I Jordan and Robert A Jacobs. 1994. Hierarchical mixtures of experts and the em algorithm. *Neural computation*, 6(2):181–214.

Min Jae Jung and JooHee Kim. 2024. Pmoe: Progressive mixture of experts with asymmetric transformer for continual learning. *arXiv preprint arXiv:2407.21571*.

Dhiraj Kalamkar, Dheevatsa Mudigere, Naveen Mellempudi, Dipankar Das, Kunal Banerjee, Sasikanth Avancha, Dharma Teja Vooturi, Nataraj Jammalamadaka, Jianyu Huang, Hector Yuen, et al. 2019. A study of bfloat16 for deep learning training. *arXiv preprint arXiv:1905.12322*.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. 2020. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*.

Andrej Karpathy. 2015. char-rnn. https://github.com/karpathy/char-rnn.

Young Jin Kim, Ammar Ahmad Awan, Alexandre Muzio, Andres Felipe Cruz Salinas, Liyang Lu, Amr Hendy, Samyam Rajbhandari, Yuxiong He, and Hany Hassan Awadalla. 2021. Scalable and efficient moe training for multitask multilingual models. *arXiv preprint arXiv:2109.10465*.

Diederik P Kingma. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Takeshi Kojima, Shixiang Shane Gu, Machel Reid, Yutaka Matsuo, and Yusuke Iwasawa. 2022. Large language models are zero-shot reasoners. *Advances in neural information processing systems*, 35:22199–22213.

Jakub Krajewski, Jan Ludziejewski, Kamil Adamczewski, Maciej Pióro, Michał Krutul, Szymon Antoniak, Kamil Ciebiera, Krystian Król, Tomasz Odrzygóźdź, Piotr Sankowski, et al. 2024. Scaling laws for fine-grained mixture of experts. *arXiv preprint arXiv:2402.07871*.

Sneha Kudugunta, Yanping Huang, Ankur Bapna, Maxim Krikun, Dmitry Lepikhin, Minh-Thang Luong, and Orhan Firat. 2021. Beyond distillation: Task-level mixture-of-experts for efficient inference. *arXiv preprint arXiv:2110.03742*.

Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. 2020. Gshard: Scaling giant models with conditional computation and automatic sharding. *arXiv preprint arXiv:2006.16668*.

Xiang Li, John Thickstun, Ishaan Gulrajani, Percy S Liang, and Tatsunori B Hashimoto. 2022. Diffusion-lm improves controllable text generation. *Advances in neural information processing systems*, 35:4328–4343.

Aixin Liu, Bei Feng, Bing Xue, Bingxuan Wang, Bochao Wu, Chengda Lu, Chenggang Zhao, Chengqi Deng, Chenyu Zhang, Chong Ruan, et al. 2024. Deepseek-v3 technical report. *arXiv preprint arXiv:2412.19437*.

Ilya Loshchilov and Frank Hutter. 2017. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*.

Zhenyan Lu, Xiang Li, Dongqi Cai, Rongjie Yi, Fangming Liu, Xiwen Zhang, Nicholas D Lane, and Mengwei Xu. 2024. Small language models: Survey, measurements, and insights. *arXiv preprint arXiv:2409.15790*.

Siyuan Mu and Sen Lin. 2025. A comprehensive survey of mixture-of-experts: Algorithms, theory, and applications. *arXiv preprint arXiv:2503.07137*.

Niklas Muennighoff, Luca Soldaini, Dirk Groeneveld, Kyle Lo, Jacob Morrison, Sewon Min, Weijia Shi, Pete Walsh, Oyvind Tafjord, Nathan Lambert, et al. 2024. Olmoe: Open mixture-of-experts language models. *arXiv preprint arXiv:2409.02060*.

Vinod Nair and Geoffrey E Hinton. 2010. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814.

Nam V Nguyen, Thong T Doan, Luong Tran, Van Nguyen, and Quang Pham. 2024. Libmoe: A library for comprehensive benchmarking mixture of experts in large language models. *arXiv preprint arXiv:2411.00918*.

Tam Nguyen, Ngoc N Tran, Khai Nguyen, and Richard G Baraniuk. 2025. Improving routing in sparse mixture of experts with graph of tokens. *arXiv preprint arXiv:2505.00792*.

Shen Nie, Fengqi Zhu, Zebin You, Xiaolu Zhang, Jingyang Ou, Jun Hu, Jun Zhou, Yankai Lin, Ji-Rong Wen, and Chongxuan Li. 2025. Large language diffusion models. *arXiv preprint arXiv:2502.09992*.

Long Ouyang, Jeffrey Wu, Xu Jiang, Diogo Almeida, Carroll Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, et al. 2022. Training language models to follow instructions with human feedback. *Advances in neural information processing systems*, 35:27730–27744.

Ashwinee Panda, Vatsal Baherwani, Zain Sarwar, Benjamin Therien, Supriyo Chakraborty, and Tom Goldstein. 2025. Dense backpropagation improves training for sparse mixture-of-experts. *arXiv preprint arXiv:2504.12463*.

Ethan Pedicir, Lucas Miller, and Liam Robinson. 2024. Novel token-level recurrent routing for enhanced mixture-of-experts performance.

Ofir Press and Lior Wolf. 2016. Using the output embedding to improve language models. *arXiv preprint arXiv:1608.05859*.

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

David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. 1986. Learning representations by back-propagating errors. *nature*, 323(6088):533–536.

David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. 1985. Learning internal representations by error propagation.

Noam Shazeer. 2020. Glu variants improve transformer. *arXiv preprint arXiv:2002.05202*.

Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. 2017. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*.

Chufan Shi, Cheng Yang, Xinyu Zhu, Jiahao Wang, Taiqiang Wu, Siheng Li, Deng Cai, Yujiu Yang, and Yu Meng. 2024. Unchosen experts can contribute too: Unleashing moe models' power by self-contrast. *arXiv preprint arXiv:2405.14507*.

Parshin Shojaee, Iman Mirzadeh, Keivan Alizadeh, Maxwell Horton, Samy Bengio, and Mehrdad Farajtabar. 2025. The illusion of thinking: Understanding the strengths and limitations of reasoning models via the lens of problem complexity. *arXiv preprint arXiv:2506.06941*.

Zhenpeng Su, Zijia Lin, Xue Bai, Xing Wu, Yizhe Xiong, Haoran Lian, Guangyuan Ma, Hui Chen, Guiguang Ding, Wei Zhou, et al. 2024. Maskmoe: Boosting token-level learning via routing mask in mixture-of-experts. *arXiv preprint arXiv:2407.09816*.

Yu Sun, Xiaolong Wang, Zhuang Liu, John Miller, Alexei Efros, and Moritz Hardt. 2020. Test-time training with self-supervision for generalization under distribution shifts. In *International conference on machine learning*, pages 9229–9248. PMLR.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need.(nips), 2017. *arXiv preprint arXiv:1706.03762*, 10:S0140525X16001837.

Leandro von Werra, Younes Belkada, Lewis Tunstall, Edward Beeching, Tristan Thrush, Nathan Lambert, Shengyi Huang, Kashif Rasul, and Quentin Gallouédec. 2020. Trl: Transformer reinforcement learning. https://github.com/huggingface/trl.

Yizhong Wang, Hamish Ivison, Pradeep Dasigi, Jack Hessel, Tushar Khot, Khyathi Chandu, David Wadden, Kelsey MacMillan, Noah A Smith, Iz Beltagy, et al. 2023. How far can camels go? exploring the state of instruction tuning on open resources. *Advances in Neural Information Processing Systems*, 36:74764–74786.

Ziteng Wang, Jun Zhu, and Jianfei Chen. 2024. Remoe: Fully differentiable mixture-of-experts with relu routing. *arXiv preprint arXiv:2412.14711*.

Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Fei Xia, Ed Chi, Quoc V Le, Denny Zhou, et al. 2022. Chain-of-thought prompting elicits reasoning in large language models. *Advances in neural information processing systems*, 35:24824–24837.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

Shaohua Wu, Jiangang Luo, Xi Chen, Lingjun Li, Xudong Zhao, Tong Yu, Chao Wang, Yue Wang, Fei Wang, Weixu Qiao, et al. 2024. Yuan 2.0-m32: Mixture of experts with attention router. *arXiv preprint arXiv:2405.17976*.

Taiqiang Wu and Ngai Wong. A unified view for attention and moe.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tieyan Liu. 2020. On layer normalization in the transformer architecture. In *International conference on machine learning*, pages 10524–10533. PMLR.

Shen Yan, Xingyan Bin, Sijun Zhang, Yisen Wang, and Zhouchen Lin. Tc-moe: Augmenting mixture of experts with ternary expert choice. In *The Thirteenth International Conference on Learning Representations*.

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

Zihao Zeng, Yibo Miao, Hongcheng Gao, Hao Zhang, and Zhijie Deng. 2024. Adamoe: Token-adaptive routing with null experts for mixture-of-experts language models. *arXiv preprint arXiv:2406.13233*.

Jia-Chen Zhang, Yu-Jie Xiong, Xi-He Qiu, Chun-Ming Xia, and Fei Dai. 2025. Mixture of routers. *arXiv preprint arXiv:2503.23362*.

Yanqi Zhou, Tao Lei, Hanxiao Liu, Nan Du, Yanping Huang, Vincent Zhao, Andrew M Dai, Quoc V Le, James Laudon, et al. 2022. Mixture-of-experts with expert choice routing. *Advances in Neural Information Processing Systems*, 35:7103–7114.

Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. 2022. St-moe: Designing stable and transferable sparse expert models. *arXiv preprint arXiv:2202.08906*.

Simiao Zuo, Xiaodong Liu, Jian Jiao, Young Jin Kim, Hany Hassan, Ruofei Zhang, Tuo Zhao, and Jianfeng Gao. 2021. Taming sparsely activated transformer with stochastic experts. *arXiv preprint arXiv:2110.04260*.