Research Article

Petr Babkin and Oleg Bakhteev

Structure Aware Neural Architecture Search for Mixture of Experts

https://doi.org/10.1515/sample-YYYY-XXXX
Received Month DD, YYYY; revised Month DD, YYYY; accepted Month DD, YYYY

Abstract: The Mixture-of-Experts (MoE) layer, a sparsely activated neural architecture controlled by a routing mechanism, has recently achieved remarkable success across large-scale deep learning tasks. In parallel, Neural Architecture Search (NAS) has emerged as a powerful methodology for automatically discovering high-performing neural network. However, the application of NAS methods to MoE architectures remains an underexplored research area. In this work, we propose an architecture search framework for MoE models, which explicitly leverages the underlying cluster structure of the data. We evaluate the proposed approach on computer vision benchmarks and demonstrate that it outperforms baseline MoE architectures trained on the same datasets in terms of accuracy and computational efficiency.

Keywords: Mixture of Experts, Neural Architecture Search

1 Introduction

The Mixture-of-Experts (MoE) architecture, in which input is partitioned through a gating function and subsequently routed as weighted input to specialized experts [1–3], has been well-established in the literature for decades [4]. Scenarios in which the experts are classical statistical models are well characterized in the literature [5], particularly when the experts are modeled as logistic regressors [6] or as Gaussian models [7, 8]. In recent years, concurrent with the growing interest in neural networks, researchers began exploring the application of MoE structures to deep learning architectures [9]. This efficiency was achieved through sparse gating mechanisms, where the input is routed only to a fixed number of experts, which is significantly smaller than the complete set [10–12]. MoE architectures found widespread adoption in the transformer domain, where model size plays a crucial role in achieving superior performance [13–15].

Simultaneously, the field of Neural Architecture Search (NAS) has developed into a well-studied area [16–18]. NAS aims to select optimal architectures from a wide range of candidates, addressing the challenge that architecture selection typically requires substantial domain expertise and manual tuning [16, 19]. The field has diverged into two main paradigms. The first is computationally intensive many-shot NAS, which involves multiple training runs followed by candidate selection using methods such as reinforcement learning [20, 21] and evolutionary algorithms [22, 23]. The second is one-shot NAS, where a single supernet is trained, and the optimal subnet is subsequently selected [19, 24, 25]. This approach effectively transforms the search problem into finding optimal compression strategies for the supernet [19].

Despite the individual successes of both MoE and NAS methodologies, the application of NAS techniques specifically to MoE architectures remains a relatively underexplored research area, presenting significant opportunities for advancing both efficiency and performance in large-scale neural architectures. In contrast to the classical approach where all experts share the same architecture [12], our study allows architectural heterogeneity between experts. We claim that this flexibility enhances the overall efficiency of the resulting architecture by enabling more specialized and adaptable expert networks.

Petr Babkin, Oleg Bakhteev, Moscow Institute of Physics and Technology, Intelligent Systems, Moscow, Russia, e-mail: babkin.pk@phystech.edu

It is important to note that several studies in the literature have addressed the problem of discovering or designing architectures for mixture-of-experts models. In the article [26], the authors employed a comprehensive combination of approaches to discover a fast-operating Mixture-of-Experts (MoE) architecture. The expert weights are derived from a supernet, and the search process is evolutionary in nature. There is a practical work [27] in which the authors develop a framework for efficient MoE inference tailored to specific hardware. The work most closely related to ours is [28], where the authors conduct a search by varying the number of experts in the layers. In contrast to this study, we propose to allow greater flexibility by optimizing the model architectures themselves rather than only the number of experts.

The MoE architecture is inherently well-suited for data with strong clustering properties. There exist theoretical studies, which demonstrate on relatively simple examples that this claim holds [29]. Additionally, practical researches show the effectiveness of MoE on multimodal datasets [30–32]. However, despite these promising results, the underlying mechanisms that enable experts to align with distinct modalities remain insufficiently understood, motivating further investigation into the principles governing modality-specific specialization within MoE frameworks.

Given the foregoing, we formulate the following research question: *how can we design conditions under which expert architectures adapt to specific data clusters?* In this study, we aim to address this question by examining how models evolve during adaptation to specific data modalities.

Contributions

- **SA-NAS framework:** We propose a Structure-Aware Neural Architecture Search (SA-NAS) framework that optimizes expert architectures with respect to the clustering structure of the data.
- **Theoretical understanding:** We provide a theoretical foundation for our method, demonstrating how the data's clustering structure influences the final architectures identified by the search algorithm.

2 Related Work

Neural Architecture Search for Mixture-of-Experts.

BLABLA

Theoretical Understanding.

[]

3 Problem Statement

3.1 Neural Architecture Search

Definition 1. *Let $\mathcal{V} = \{1, 2, \ldots, N\}$ be a set of N vertices representing nodes in a directed acyclic graph (DAG). We define a set of edges as*

$$\mathcal{E} = \{(i, j) \in \mathcal{V} \times \mathcal{V} \mid i < j\}, \tag{1}$$

representing the possible connections between vertices, where the ordering constraint $i < j$ enforces a topological structure.

Definition 2. *The set of admissible operations $\mathcal{O}$ comprises a collection of differentiable or discrete operations that can be applied to edges in the network architecture. Common operations include convolutions of varying kernel sizes (e.g., 1×1, 3×3), pooling operations (max pooling, average pooling), and skip connections.*

For each edge $(i, j) \in \mathcal{E}$, an operation $o^{(i,j)} \in \mathcal{O}$ is assigned to transmit information from vertex i to vertex j. The task of neural architecture search is thus reduced to determining the optimal assignment of operations to edges.

Definition 3. *Let $\boldsymbol{\alpha} \in \mathcal{A}$ be an architecture vector encoding the operations assigned to each edge, where $\mathcal{A}$ is the search space comprising all possible architecture configurations. Each component $\alpha_{i,j}$ specifies the operation on edge (i, j).*

For a given architecture $\boldsymbol{\alpha}_k \in \mathcal{A}$ and its corresponding parameters $\boldsymbol{w}_{\boldsymbol{\alpha}_k} \in \mathcal{W}_{\boldsymbol{\alpha}_k}$, we denote the neural network as $f(\boldsymbol{\alpha}_k, \boldsymbol{w}_{\boldsymbol{\alpha}_k}) : \mathcal{X} \to \mathcal{Y}$, where $\mathcal{X}$ and $\mathcal{Y}$ are the input and output spaces respectively.

Definition 4. *The empirical loss functions on training and validation datasets are defined as follows:*

$$\mathcal{L}_{train}(f) = \sum_{(\boldsymbol{x},y)\in\mathcal{D}_{train}} \ell(f(\boldsymbol{x}), y), \quad \mathcal{L}_{val}(f) = \sum_{(\boldsymbol{x},y)\in\mathcal{D}_{val}} \ell(f(\boldsymbol{x}), y), \tag{2}$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}_{\geq 0}$ is a task-specific loss function, and $\mathcal{D}_{train}$ and $\mathcal{D}_{val}$ denote the training and validation datasets respectively. We use the shorthand notation $\mathcal{L}_{train}(\boldsymbol{\alpha}_k, \boldsymbol{w}_{\boldsymbol{\alpha}_k})$ and $\mathcal{L}_{val}(\boldsymbol{\alpha}_k, \boldsymbol{w}_{\boldsymbol{\alpha}_k})$ interchangeably with $\mathcal{L}_{train}(f(\boldsymbol{\alpha}_k, \boldsymbol{w}_{\boldsymbol{\alpha}_k}))$ and $\mathcal{L}_{val}(f(\boldsymbol{\alpha}_k, \boldsymbol{w}_{\boldsymbol{\alpha}_k}))$ respectively.

The neural architecture search problem is formulated as a bilevel optimization problem:

$$\begin{aligned} &\min_{\boldsymbol{\alpha}\in\mathcal{A}} \mathcal{L}_{\text{val}}(\boldsymbol{\alpha}, \boldsymbol{w}^*_{\boldsymbol{\alpha}}), \\ \text{s.t.} \quad &\boldsymbol{w}^*_{\boldsymbol{\alpha}} = \arg\min_{\boldsymbol{w}\in\mathcal{W}_{\boldsymbol{\alpha}}} \mathcal{L}_{\text{train}}(\boldsymbol{\alpha}, \boldsymbol{w}), \end{aligned} \tag{3}$$

where $\mathcal{W}_{\boldsymbol{\alpha}}$ denotes the space of all learnable parameters for architecture $\boldsymbol{\alpha}$.

Interpretation: The outer minimization seeks an architecture $\boldsymbol{\alpha}$ that minimizes validation loss. The constraint ensures that the network weights $\boldsymbol{w}^*_{\boldsymbol{\alpha}}$ are optimally trained on the training dataset for the selected architecture. This formulation reflects the fundamental trade-off between model expressiveness (architecture choice) and generalization capability (validation performance).

3.2 Mixture-of-Experts Architecture

Definition 5. *A Mixture-of-Experts (MoE) model is a composite model comprising K expert networks, each specialized for a distinct subset of the input space. Given a dataset partitioned into K clusters, $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2 \cup \cdots \cup \mathcal{D}_K$ (forming a partition), where each cluster $\mathcal{D}_k$ corresponds to a specialized domain or data modality.*

Let $\boldsymbol{\alpha} = [\boldsymbol{\alpha}_1^T, \boldsymbol{\alpha}_2^T, \ldots, \boldsymbol{\alpha}_K^T]^T$ be the concatenation of architecture vectors for all K experts. We denote a Mixture-of-Experts model with architecture $\boldsymbol{\alpha}$ and parameters $\boldsymbol{w}_{\boldsymbol{\alpha}}$ as $g(\boldsymbol{\alpha}, \boldsymbol{w}_{\boldsymbol{\alpha}})$.

The MoE prediction is computed as a weighted combination of expert outputs:

$$g(\boldsymbol{\alpha}, \boldsymbol{w}_{\boldsymbol{\alpha}}, \boldsymbol{w}_r)(\boldsymbol{x}) = \sum_{k=1}^{K} r_k(\boldsymbol{x}, \boldsymbol{w}_r) \cdot f(\boldsymbol{\alpha}_k, \boldsymbol{w}^*_{\boldsymbol{\alpha}_k})(\boldsymbol{x}), \tag{4}$$

where $f(\boldsymbol{\alpha}_k, \boldsymbol{w}^*_{\boldsymbol{\alpha}_k})$ denotes the k-th expert network, and $r_k(\boldsymbol{x}, \boldsymbol{w}_r) \geq 0$ are routing gate values learned by a gating network with parameters $\boldsymbol{w}_r$. The routing gates typically satisfy $\sum_{k=1}^{K} r_k(\boldsymbol{x}, \boldsymbol{w}_r) = 1$.

3.3 Neural Architecture Search for Mixture-of-Experts

The problem of discovering optimal architectures for each expert in a Mixture-of-Experts model can be formulated as an extension of Problem 3 to the multi-expert setting.

$$\begin{aligned} &\min_{\boldsymbol{\alpha}\in\mathcal{A}^K} \mathcal{L}_{\text{val}}(\boldsymbol{\alpha}, \boldsymbol{w}^*_{\boldsymbol{\alpha}}, \boldsymbol{w}_r), \\ \text{s.t.} \quad &(\boldsymbol{w}^*_{\boldsymbol{\alpha}}, \boldsymbol{w}^*_r) = \arg\min_{\boldsymbol{w},\boldsymbol{w}_r\in\mathcal{W}_{\boldsymbol{\alpha}}\times\mathcal{W}_r} \mathcal{L}_{\text{train}}(\boldsymbol{\alpha}, \boldsymbol{w}, \boldsymbol{w}_r) \end{aligned} \tag{5}$$

where $\mathcal{A}^K$ denotes the combined search space for K expert architectures.

Remark 1. *Problem 5 extends the search space from $\mathcal{A}$ to $\mathcal{A}^K$, exponentially increasing computational complexity. To mitigate this complexity, we propose the following decomposition.*

We reformulate Problem 5 by decomposing the global optimization objective into cluster-wise sub-problems:

$$\begin{aligned} &\min_{\boldsymbol{\alpha}\in\mathcal{A}^K} \sum_{k=1}^{K} \mathcal{L}^{\mathcal{D}_k}_{\text{val}}(\boldsymbol{\alpha}_k, \boldsymbol{w}^*_{\boldsymbol{\alpha}_k}), \\ \text{s.t.} \quad &\forall k \in \{1, 2, \ldots, K\}, \quad \boldsymbol{w}^*_{\boldsymbol{\alpha}_k} = \arg\min_{\boldsymbol{w}\in\mathcal{W}_{\boldsymbol{\alpha}_k}} \mathcal{L}^{\mathcal{D}_k}_{\text{train}}(\boldsymbol{w}, \boldsymbol{\alpha}_k), \end{aligned} \tag{6}$$

where $\mathcal{L}^{\mathcal{D}_k}_{\text{val}}$ and $\mathcal{L}^{\mathcal{D}_k}_{\text{train}}$ denote validation and training losses computed on cluster $\mathcal{D}_k$ respectively.

Motivation: This decomposition assumes that the optimal expert for cluster $\mathcal{D}_k$ can be found independently by optimizing the architecture and weights on the k-th cluster's data. This assumption is justified when clusters represent sufficiently distinct data distributions and the experts operate independently (no parameter sharing across experts beyond the gating network).

4 Main Part

4.1 Surrogate Function Training

Direct optimization of Problem 6 requires repeatedly training neural networks for different architecture candidates, which is prohibitively expensive. To address this computational bottleneck, we employ a surrogate model approach.

The surrogate model will predict model efficieny on every data cluster. Thus we will be able to choose most efficient network for every cluster. More formally, surrogate function is defined as follows:

Definition 6. *Let us assume that train data split into K distinct clusters. A surrogate function $u : \mathcal{A}\times\mathbb{R}^m \to \mathbb{R}^K$ is a learned meta-model that maps architecture descriptions to predicted performance metrics across clusters. Formally, for an architecture $\boldsymbol{\alpha}$, the surrogate function produces predictions*

$$u(\boldsymbol{\alpha}) = [\hat{\mathcal{L}}^{\mathcal{D}_1}_{val}(\boldsymbol{\alpha}_1), \hat{\mathcal{L}}^{\mathcal{D}_2}_{val}(\boldsymbol{\alpha}_2), \ldots, \hat{\mathcal{L}}^{\mathcal{D}_K}_{val}(\boldsymbol{\alpha}_K)], \tag{7}$$

where $\hat{\mathcal{L}}^{\mathcal{D}_k}_{val}$ approximates the true validation loss on cluster $\mathcal{D}_k$ without full training.

Definition 7. *The surrogate function is trained on a dataset of sampled architectures and their corresponding true performance values:*

$$\mathcal{D}_{surr} = \{(\boldsymbol{\alpha}^{(i)}, \boldsymbol{\ell}^{(i)})\}_{i=1}^{M}, \tag{8}$$

where $\boldsymbol{\ell}^{(i)} = [\mathcal{L}^{\mathcal{D}_1}_{val}(\boldsymbol{\alpha}^{(i)}_1, \boldsymbol{w}^*_{\boldsymbol{\alpha}^{(i)}_1}), \ldots, \mathcal{L}^{\mathcal{D}_K}_{val}(\boldsymbol{\alpha}^{(i)}_K, \boldsymbol{w}^*_{\boldsymbol{\alpha}^{(i)}_K})]^T$ are ground-truth performance vectors obtained through full network training.

The surrogate function is trained by minimizing prediction error:

$$\min_{\boldsymbol{\theta}} \sum_{(\boldsymbol{\alpha},\boldsymbol{\ell})\in\mathcal{D}_{\text{surr}}} \|u_{\boldsymbol{\theta}}(\boldsymbol{\alpha}) - \boldsymbol{\ell}\|_2^2, \tag{9}$$

where $\boldsymbol{\theta}$ denotes the parameters of the surrogate model $u_{\boldsymbol{\theta}}$.

Rationale: The surrogate function enables efficient architecture search by replacing expensive full training with fast surrogate predictions. This is particularly valuable in the MoE setting, where evaluating a single architecture candidate requires training K expert networks.

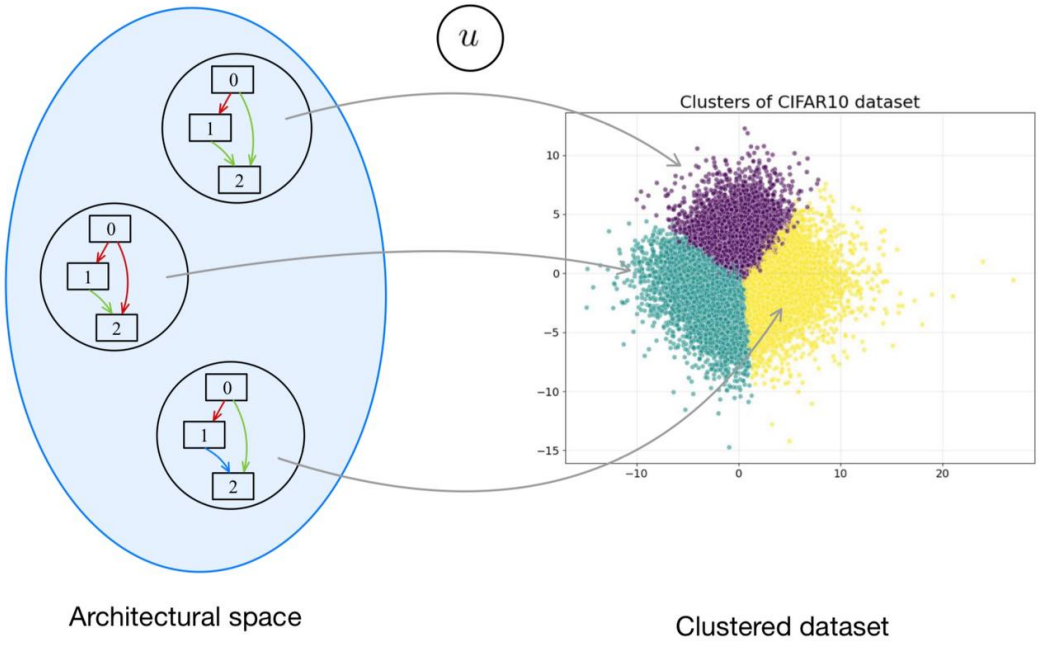

Fig. 1: Method workflow: a surrogate model u is used to select the most suitable models for each cluster.

5 Experiments

6 Conclusion

References

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

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

[3] Robert A Jacobs, Fengchun Peng, and Martin A Tanner. A bayesian approach to model selection in hierarchical mixtures-of-experts architectures. *Neural Networks*, 10(2):231–241, 1997.

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

[5] Jakub Piwko, Jędrzej Ruciński, Dawid Płudowski, Antoni Zajko, Patryzja Żak, Mateusz Zacharecki, Anna Kozak, and Katarzyna Woźnica. Divide, specialize, and route: A new approach to efficient ensemble learning. *arXiv preprint arXiv:2506.20814*, 2025.

[6] Huy Nguyen, Pedram Akbarian, TrungTin Nguyen, and Nhat Ho. A general theory for softmax gating multinomial logistic mixture of experts. *arXiv preprint arXiv:2310.14188*, 2023.

[7] Huy Nguyen, TrungTin Nguyen, and Nhat Ho. Demystifying softmax gating function in gaussian mixture of experts. *Advances in Neural Information Processing Systems*, 36:4624–4652, 2023.

[8] Carl Rasmussen and Zoubin Ghahramani. Infinite mixtures of gaussian process experts. *Advances in neural information processing systems*, 14, 2001.

[9] David Eigen, Marc'Aurelio Ranzato, and Ilya Sutskever. Learning factored representations in a deep mixture of experts. *arXiv preprint arXiv:1312.4314*, 2013.

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

[11] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. *Advances in Neural Information Processing Systems*, 34:8583–8595, 2021.

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

[13] 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. Mixtral of experts. *arXiv preprint arXiv:2401.04088*, 2024.

[14] Zihan Qiu, Zeyu Huang, Bo Zheng, Kaiyue Wen, Zekun Wang, Rui Men, Ivan Titov, Dayiheng Liu, Jingren Zhou, and Junyang Lin. Demons in the detail: On implementing load balancing loss for training specialized mixture-of-expert models. *arXiv preprint arXiv:2501.11873*, 2025.

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

[16] Pengzhen Ren, Yun Xiao, Xiaojun Chang, Po-Yao Huang, Zhihui Li, Xiaojiang Chen, and Xin Wang. A comprehensive survey of neural architecture search: Challenges and solutions. *ACM Computing Surveys (CSUR)*, 54(4):1–34, 2021.

[17] Niv Nayman, Asaf Noy, Tal Ridnik, Itamar Friedman, Rong Jin, and Lihi Zelnik. Xnas: Neural architecture search with expert advice. *Advances in neural information processing systems*, 32, 2019.

[18] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V Le. Learning transferable architectures for scalable image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 8697–8710, 2018.

[19] Hanxiao Liu, Karen Simonyan, and Yiming Yang. Darts: Differentiable architecture search. *arXiv preprint arXiv:1806.09055*, 2018.

[20] Barret Zoph and Quoc V Le. Neural architecture search with reinforcement learning. *arXiv preprint arXiv:1611.01578*, 2016.

[21] Bowen Baker, Otkrist Gupta, Nikhil Naik, and Ramesh Raskar. Designing neural network architectures using reinforcement learning. *arXiv preprint arXiv:1611.02167*, 2016.

[22] Esteban Real, Alok Aggarwal, Yanping Huang, and Quoc V Le. Regularized evolution for image classifier architecture search. In *Proceedings of the aaai conference on artificial intelligence*, volume 33, pages 4780–4789, 2019.

[23] Zhaohui Yang, Yunhe Wang, Xinghao Chen, Boxin Shi, Chao Xu, Chunjing Xu, Qi Tian, and Chang Xu. Cars: Continuous evolution for efficient neural architecture search. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 1829–1838, 2020.

[24] Xiangning Chen and Cho-Jui Hsieh. Stabilizing differentiable architecture search via perturbation-based regularization. In *International conference on machine learning*, pages 1554–1565. PMLR, 2020.

[25] Shan You, Tao Huang, Mingmin Yang, Fei Wang, Chen Qian, and Changshui Zhang. Greedynas: Towards fast one-shot nas with greedy supernet. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1999–2008, 2020.

[26] Ganesh Jawahar, Subhabrata Mukherjee, Xiaodong Liu, Young Jin Kim, Muhammad Abdul-Mageed, Laks VS Lakshmanan, Ahmed Hassan Awadallah, Sebastien Bubeck, and Jianfeng Gao. Automoe: Heterogeneous mixture-of-experts with adaptive computation for efficient neural machine translation. *arXiv preprint arXiv:2210.07535*, 2022.

[27] Shihao Han, Sishuo Liu, Shucheng Du, Mingzi Li, Zijian Ye, Xiaoxin Xu, Yi Li, Zhongrui Wang, and Dashan Shang. Cmn: a co-designed neural architecture search for efficient computing-in-memory-based mixture-of-experts. *Science China Information Sciences*, 67(10):200405, 2024.

[28] Lotfi Abdelkrim Mecharbat, Alberto Marchisio, Muhammad Shafique, Mohammad M Ghassemi, and Tuka Alhanai. Moenas: Mixture-of-expert based neural architecture search for jointly accurate, fair, and robust edge deep neural networks. *arXiv preprint arXiv:2502.07422*, 2025.

[29] Zixiang Chen, Yihe Deng, Yue Wu, Quanquan Gu, and Yuanzhi Li. Towards understanding the mixture-of-experts layer in deep learning. *Advances in neural information processing systems*, 35:23049–23062, 2022.

[30] Tsz Chai Fung and Spark C Tseung. Mixture of experts models for multilevel data: modelling framework and approximation theory. *arXiv e-prints*, pages arXiv–2209, 2022.

[31] Jiang Guo, Darsh J Shah, and Regina Barzilay. Multi-source domain adaptation with mixture of experts. *arXiv preprint arXiv:1809.02256*, 2018.

[32] Zhiwei Hu, Víctor Gutiérrez-Basulto, Zhiliang Xiang, Ru Li, and Jeff Z Pan. Multi-level mixture of experts for multimodal entity linking. In *Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining V. 2*, pages 979–990, 2025.

A Missing Proofs

B Wide Experiments