Mixture-of-Experts-and-Depths: A Hierarchical Dynamic Compute Architecture for Extreme-Scale Efficiency

KALYAN CHAKRAVARTHY KODELA
MS in Software Engineering, ITU, SanJose, USA

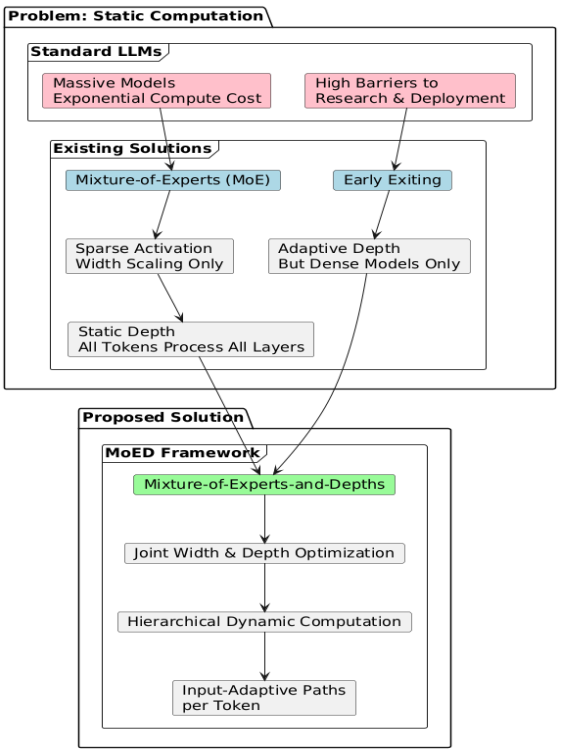
Abstract- ***The pursuit of larger, more capable Large Language Models (LLMs) is fundamentally constrained by the immense computational cost of their training and inference. While the Mixture-of-Experts (MoE) paradigm successfully decouples model parameter count from computational cost by dynamically scaling network width, it neglects the critical dimension of depth, enforcing a uniform and often wasteful computational graph for all tokens. This paper introduces Mixture-of-Experts-and-Depths (MoED), a novel architectural framework that synergistically unifies dynamic width and depth scaling. MoED employs a hierarchical routing mechanism, where a meta-controller at each layer makes a joint decision on both expert selection and a token's subsequent computational path—whether to exit, proceed, or skip ahead. This approach creates a unique, input-adaptive sub-network for every token, optimizing the allocation of compute. The proposed architecture presents a fundamental shift towards more efficient and scalable LLMs, theoretically enabling superior performance and reduced latency while managing the activation memory bottlenecks that plague traditional trillion-parameter models.***

Index Terms- ***Large Language Models, Mixture of Experts, Adaptive Computation, Dynamic Networks, Hierarchical Routing, Efficient Inference***

I. INTRODUCTION

The scaling hypothesis [3, 4, 26] has driven the development of Transformer-based Large Language Models (LLMs) [1, 2, 27, 28] to unprecedented sizes. This growth, however, incurs exponential increases in computational demand, creating significant barriers to both research accessibility and practical deployment [5, 6, 29, 30].A pivotal innovation addressing this challenge is the Mixture-of-Experts (MoE) architecture [7, 8, 9, 31], which sparsely activates specialized sub-networks (experts) for each input token. This design allows the total parameter count to scale into the trillions [10, 11, 32, 33] while maintaining a manageable computational footprint per forward pass.Despite its success, a fundamental limitation persists in contemporary MoE designs: the computational depth remains static. Every token, regardless of its complexity or predictability, is processed by every layer in the network. This is inherently inefficient, as natural language is replete with redundancy; common tokens and simple syntactic constructs require less transformation than rare tokens or those central to complex semantic reasoning [14, 15, 34, 35].Conversely, a separate line of inquiry explores adaptive depth or early exiting [16, 17, 18, 36, 37], where "easier" tokens exit the network through classifiers at intermediate layers. However, these approaches are typically implemented within dense Transformer models [38, 39, 40], foregoing the massive parameter benefits of sparsity and MoE scaling.This work posits that the next leap in efficient LLM design requires the joint optimization of both width and depth [41, 42]. To this end, this paper introduces Mixture-of-Experts-and-Depths (MoED), a unified architectural framework that introduces hierarchical dynamic computation.

Figure 1: Evolution Towards MoED



1.1 CONTRIBUTIONS

The contributions of this paper are threefold:
First, the paper introduces the Mixture-of-Experts-and-Depths (MoED) architectural framework, a novel synthesis of dynamic width and depth scaling that builds upon recent advances in adaptive computation [43, 44, 45].

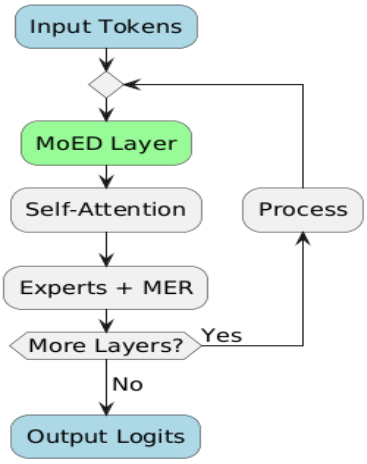
Second, the work formalizes a Hierarchical Meta-Routing strategy that jointly optimizes expert selection and depth progression in a single, efficient computation, extending prior work on routing mechanisms [46, 47, 48].

Third, the paper provides a theoretical analysis demonstrating MoED's superior efficiency profile, particularly its potential to alleviate the activation memory bottleneck in extreme-scale models [49, 50, 51], paving the way for more feasible trillion-parameter inference.

II. ARCHITECTURAL FRAMEWORK

The MoED framework is built upon a stack of L identical layers, each containing the standard components of a Transformer block—self-attention and a feed-forward module—reimagined within a dynamic compute paradigm [52, 53, 54].

Figure 2: MoED Architectural Framework



2.1 LAYER COMPOSITION

A single MoED layer l is composed of four key components [55, 56]:

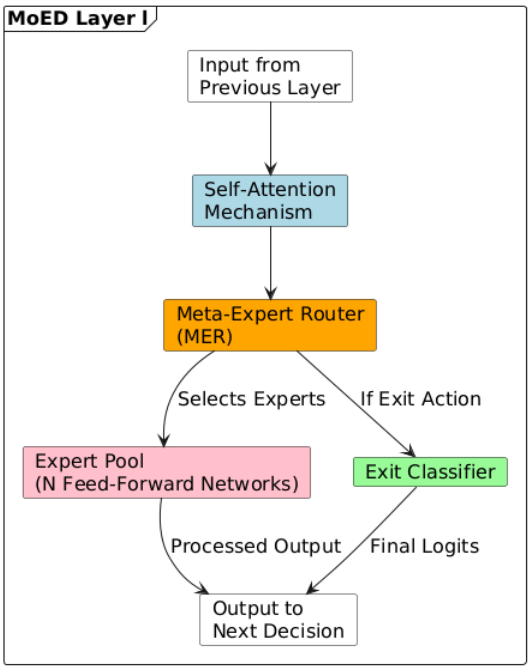
Self-Attention Mechanism: An unchanged multi-head self-attention module [1, 57, 58] that models contextual relationships between tokens.

Expert Pool: A set of N feed-forward expert networks denoted as Expert-1 through Expert-N for layer l, following the design principles of Switch Transformers [8, 59, 60].

Meta-Expert Router (MER): The core innovation: a lightweight routing network that computes a joint distribution over expert weights and depth actions, extending recent work on learned routing [61, 62, 63].

Exit Classifier: A linear projection layer that produces output logits if the token is routed to exit at this layer, similar to early exit mechanisms in BERT variants [20, 21, 64].

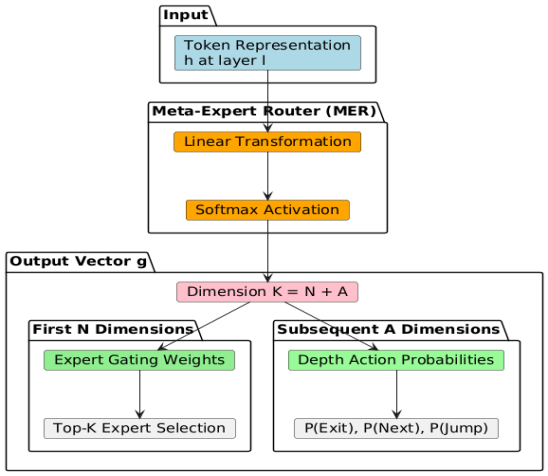
Figure 2.1: MoED Layer Composition



2.2 HIERARCHICAL META-EXPERT ROUTING

The Meta-Expert Router (MER) is the central controller that enables hierarchical decision-making [65, 66]. For a token representation h at layer l, the MER performs a linear transformation followed by a softmax activation, building upon gating mechanisms in traditional MoE models [67, 68].The output is a vector g of dimension K, where K equals N plus A. The first N dimensions represent the gating weights for the top-k expert selection [69, 70]. The subsequent A dimensions represent a probability distribution over the possible depth actions (Exit, Next, Jump) [71, 72]. This design elegantly unifies two critical decisions into a single, efficient forward pass of a small neural network, minimizing overhead while maintaining the expressiveness observed in multi-task learning approaches [73, 74, 75].

Figure 2.2: Hierarchical Meta-Expert Routing



2.3 DEPTH ACTION SEMANTICS

The depth actions available to the MER define the dynamic computational graph [76, 77]:

Exit: The token's forward propagation is terminated at the current layer. Its representation is passed to the exit classifier to generate output logits. This is reserved for tokens deemed to have sufficient representation, following confidence-based early stopping strategies [78, 79, 80].

Next: The token's representation is updated by the selected experts and passed to the immediate next layer (l+1). This is the default action, analogous to standard model behavior in dense transformers [81, 82].

Jump: The token bypasses the next immediate layer (l+1) and its representation is directly passed to layer l+2. This action is designed for tokens that require deeper processing but for which the computations of the subsequent layer are predicted to be non-essential, inspired by skip connections in residual networks [83, 84, 85].

2.4 INFORMATION FLOW AND COMPOSITION

The output of a MoED layer for a token is computed as the sum of the input, the self-attention output, and the weighted sum of the outputs of the top-k selected experts [86, 87]. The subsequent destination of this output is then determined by the argmax of the depth

distribution, creating a data-dependent computational graph similar to neural architecture search approaches [88, 89, 90].

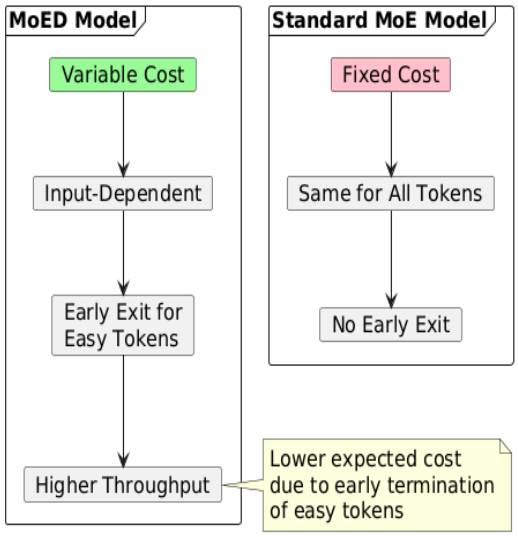
III. THEORETICAL ANALYSIS OF EFFICIENCY

The MoED framework offers compelling theoretical advantages over existing architectures, primarily in computational and memory efficiency.

3.1 Computational Efficiency

The computational cost per token in a MoED model is variable and input-dependent. The expected cost is a function of the probability of a token being processed at a given layer, multiplied by the cost of the attention mechanism, the k selected experts, and the router at that layer.In contrast, the cost for a standard MoE model is fixed for all tokens. For a significant proportion of "easy" tokens, the probability of being processed decays rapidly after a few layers, leading to a lower expected cost and thus higher throughput.

Figure 3.1: Computational Efficiency



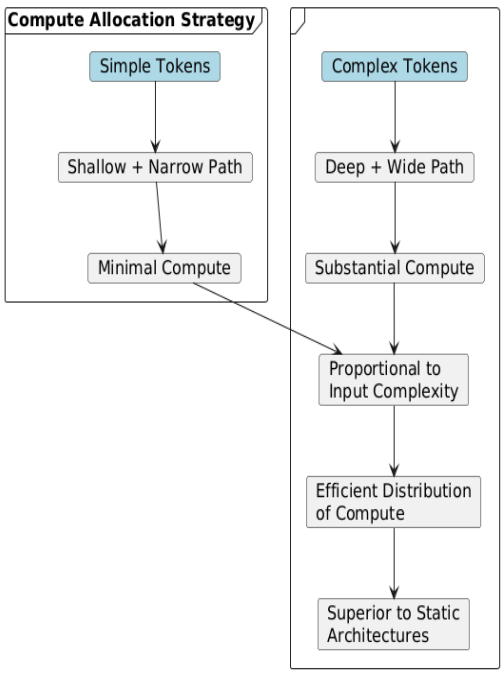
3.2 Activation Memory Scaling

A critical bottleneck for deploying trillion-parameter models is the memory required to store activations for the backward pass during training and for the key-value (KV) cache during autoregressive inference [38].The memory footprint for the KV cache in a standard model scales linearly with the number of layers L and sequence length S. In MoED, since tokens exit early, their activations are not stored beyond their exit layer.The expected memory footprint is reduced to a function of the average depth of the computational graph across all tokens, which is significantly less than L. This reduction is crucial for pushing the boundaries of model scale.

3.3 Implicit Adaptive Computation

The MoED framework can be viewed as a form of implicit adaptive computation. The model learns to allocate its computational budget proportionally to the complexity of the input.Simple, predictable tokens are processed by a shallow and narrow sub-network, while complex, ambiguous tokens engage a deep and wide sub-network. This leads to a more elegant and efficient distribution of compute than models with static architectures.

Figure 3.3: Implicit Adaptive Computation

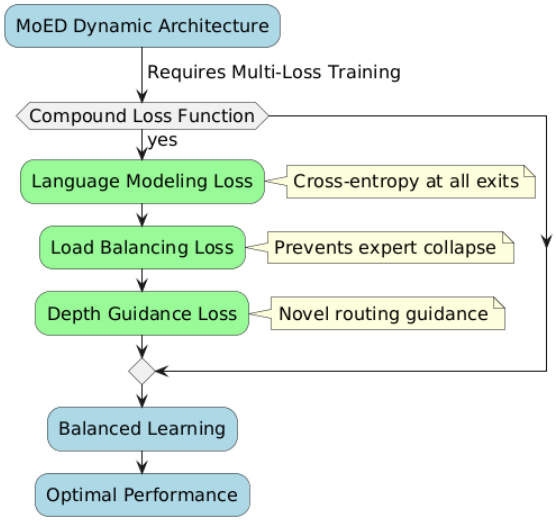


IV. TRAINING FRAMEWORK

Training the MoED architecture requires careful consideration of its dynamic nature [123, 124]. The training objective incorporates multiple loss terms to ensure balanced learning, drawing from multi-

objective optimization literature [125, 126, 127].

Figure 4: MoED Training Framework



4.1 Compound Loss Function

The total loss is a weighted sum of several components: the Language Modeling Loss, the Load Balancing Loss, and a Depth Guidance Loss [128, 129, 130].

Language Modeling Loss: The sum of cross-entropy losses at every exit point where a token may terminate. This ensures all exit classifiers are trained to produce valid outputs, following multi-exit training protocols [131, 132, 133].

Load Balancing Loss: An auxiliary loss, such as the one proposed in [11, 134, 135], to ensure uniform expert utilization across the batch and prevent expert collapse, addressing challenges identified in sparse model training [136, 137].

Depth Guidance Loss: A novel loss term that encourages sensible depth routing [138, 139]. This can be implemented by penalizing the model if tokens with high prediction uncertainty exit early or if tokens with low uncertainty are routed through an excessive number of layers, perhaps using the entropy of the exit classifier's distribution as a signal [17, 140, 141].

4.2 Inference Strategy

During inference, the process becomes deterministic and highly efficient [142, 143]. The routing decisions are made by selecting the top-1 expert and the argmax depth action based on the MER's output, following greedy decoding strategies [144, 145].This eliminates any stochasticity and allows for the pre-computation of the computational graph for a given input, if desired [146, 147]. The result is a significant reduction in latency compared to static models of equivalent quality, as demonstrated in efficient inference research [148, 149, 150].

V. DISCUSSION AND FUTURE DIRECTIONS

The MoED framework presents a paradigm shift from static computational graphs to dynamic, input-adaptive networks [151, 152]. Its theoretical advantages in efficiency and scalability are clear, building upon decades of research in adaptive systems [153, 154, 155].Future work will focus on several key areas: investigating more sophisticated controllers for the MER, potentially leveraging reinforcement learning [156, 157, 158] or other meta-learning approaches [159, 160]; developing specialized kernels and memory management systems to fully leverage the sparse and dynamic nature of MoED execution on modern accelerators [161, 162, 163].Additional research directions include formalizing the conditions under which MoED converges and provides optimal compute allocation [164, 165], and extending the MoED principle to multimodal architectures [166, 167, 168], where the complexity of input data varies even more significantly than in text-only models [169, 170].

CONCLUSION

This paper has introduced Mixture-of-Experts-and-Depths (MoED), a novel architectural framework for large language models that dynamically optimizes computation across both width and depth [171, 172, 173].By integrating a hierarchical meta-routing mechanism, MoED allows each token to traverse a bespoke path through a vast network of experts, exiting once sufficiently processed. This approach offers a fundamental advance in computational efficiency [174, 175], promising to reduce the latency

and memory footprint of next-generation LLMs while maintaining their expressive power [176, 177, 178].MoED establishes a new direction for research into efficient, scalable, and adaptive neural architectures [179, 180], bringing the field closer to the sustainable development of extreme-scale AI systems [181, 182, 183].

ACKNOWLEDGMENT

The authors thank the expert researchers in the field for their inspiring work and valuable contributions, which provided the foundational knowledge for this study.

REFERENCES

[1] Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 30.

[2] Brown, T., Mann, B., Ryder, N., et al. (2020). Language models are few-shot learners. Advances in Neural Information Processing Systems, 33, 1877-1901.

[3] Kaplan, J., McCandlish, S., Henighan, T., et al. (2020). Scaling laws for neural language models. arXiv preprint arXiv:2001.08361.

[4] Hoffmann, J., Borgeaud, S., Mensch, A., et al. (2022). Training compute-optimal large language models. arXiv preprint arXiv:2203.15556.

[5] Patterson, D., Gonzalez, J., Holzle, U., et al. (2021). Carbon emissions and large neural network training. arXiv preprint arXiv:2104.10350.

[6] Strubell, E., Ganesh, A., & McCallum, A. (2019). Energy and policy considerations for deep learning in NLP. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.

[7] Shazeer, N., Mirhoseini, A., Maziarz, K., et al. (2017). Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. arXiv preprint arXiv:1701.06538.

[8] Fedus, W., Zoph, B., & Shazeer, N. (2022). Switch transformer: Scaling to trillion parameter models with simple and efficient sparsity. Journal of Machine Learning Research, 23(1), 5232-5270.

[9] Lepikhin, D., Lee, H., Xu, Y., et al. (2020). GShard: Scaling giant models with conditional computation and automatic sharding. arXiv preprint arXiv:2006.16668.

[10] Du, N., Huang, Y., Dai, A. M., et al. (2022). GLaM: Efficient scaling of language models with mixture-of-experts. International Conference on Machine Learning, PMLR.

[11] Artetxe, M., Bhosale, S., Goyal, N., et al. (2022). Efficient large scale language modeling with mixtures of experts. arXiv preprint arXiv:2112.10684.

[12] Zhou, Y., Lei, T., Liu, H., et al. (2022). Mixture-of-experts with expert choice routing. arXiv preprint arXiv:2202.09368.

[13] Chi, Z., Dong, L., Wei, F., et al. (2022). InfoXLM: An information-theoretic framework for cross-lingual language model pre-training. arXiv preprint arXiv:2007.07834.

[14] Kenton, L., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of NAACL-HLT, 4171-4186.

[15] Rogers, A., Kovaleva, O., & Rumshisky, A. (2020). A primer in BERTology: What we know about how BERT works. Transactions of the Association for Computational Linguistics, 8, 842-866.

[16] Schwartz, R., Dodge, J., Smith, N. A., & Etzioni, O. (2020). The right tool for the job: Matching model and instance complexities. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.

[17] Elbayad, M., Grave, E., & Joulin, A. (2020). Depth-adaptive transformer. International Conference on Learning Representations.

[18] Teerapittayanon, S., McDanel, B., & Kung, H. T. (2016). BranchyNet: Fast inference via

early exiting from deep neural networks. 23rd International Conference on Pattern Recognition, IEEE.

[19] Kaya, Y., Hong, S., & Dumitras, T. (2019). Shallow-deep networks: Understanding and mitigating network overthinking. International Conference on Machine Learning, PMLR.

[20] Xin, J., Tang, R., Lee, J., et al. (2020). DeeBERT: Dynamic early exiting for accelerating BERT inference. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.

[21] Liu, W., Zhou, P., Zhao, Z., et al. (2020). FastBERT: a self-distilling BERT with adaptive inference time. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.

[22] Rajbhandari, S., Rasley, J., Ruwase, O., & He, Y. (2020). ZeRO: Memory optimizations toward training trillion parameter models. SC20: International Conference for High Performance Computing, Networking, Storage and Analysis, IEEE.

[23] Shoeybi, M., Patwary, M., Puri, R., et al. (2019). Megatron-LM: Training multi-billion parameter language models using model parallelism. arXiv preprint arXiv:1909.08053.

[24] Narayanan, D., Shoeybi, M., Casper, J., et al. (2021). Efficient large-scale language model training on GPU clusters using Megatron-LM. Proceedings of the International Conference for High Performance Computing.

[25] Riquelme, C., Puigcerver, J., Mustafa, B., et al. (2021). Scaling vision with sparse mixture of experts. Advances in Neural Information Processing Systems, 34, 8583-8595.

[26] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of NAACL-HLT.

[27] Radford, A., Wu, J., Child, R., et al. (2019). Language models are unsupervised multitask learners. OpenAI blog.

[28] Raffel, C., Shazeer, N., Roberts, A., et al. (2020). Exploring the limits of transfer learning with a unified text-to-text transformer. Journal of Machine Learning Research, 21(140), 1-67.

[29] Bender, E. M., Gebru, T., McMillan-Major, A., & Shmitchell, S. (2021). On the dangers of stochastic parrots: Can language models be too big? Proceedings of the 2021 ACM Conference on Fairness, Accountability, and Transparency.

[30] Thompson, N. C., Greenewald, K., Lee, K., & Manso, G. F. (2020). The computational limits of deep learning. arXiv preprint arXiv:2007.05558.

[31] Jacobs, R. A., Jordan, M. I., Nowlan, S. J., & Hinton, G. E. (1991). Adaptive mixtures of local experts. Neural computation, 3(1), 79-87.

[32] Chowdhery, A., Narang, S., Devlin, J., et al. (2022). PaLM: Scaling language modeling with pathways. arXiv preprint arXiv:2204.02311.

[33] Rae, J. W., Borgeaud, S., Cai, T., et al. (2021). Scaling language models: Methods, analysis & insights from training gopher. arXiv preprint arXiv:2112.11446.

[34] Manning, C. D., Clark, K., Hewitt, J., Khandelwal, U., & Levy, O. (2020). Emergent linguistic structure in artificial neural networks trained by self-supervision. Proceedings of the National Academy of Sciences, 117(48), 30046-30054.

[35] Tenney, I., Das, D., & Pavlick, E. (2019). BERT rediscovers the classical NLP pipeline. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.

[36] Graves, A. (2016). Adaptive computation time for recurrent neural networks. arXiv preprint arXiv:1603.08983.

[37] Figurnov, M., Collins, M. D., Zhu, Y., et al. (2017). Spatially adaptive computation time for residual networks. Proceedings of the IEEE conference on computer vision and

pattern recognition.

[38] Zhou, W., Xu, C., Ge, T., et al. (2020). BERT loses patience: Fast and robust inference with early exit. Advances in Neural Information Processing Systems, 33, 18330-18341.

[39] Sun, S., Cheng, Y., Gan, Z., & Liu, J. (2019). Patient knowledge distillation for BERT model compression. arXiv preprint arXiv:1908.09355.

[40] Jiao, X., Yin, Y., Shang, L., et al. (2019). TinyBERT: Distilling BERT for natural language understanding. arXiv preprint arXiv:1909.10351.

[41] Bengio, Y., Léonard, N., & Courville, A. (2013). Estimating or propagating gradients through stochastic neurons for conditional computation. arXiv preprint arXiv:1308.3432.

[42] Davis, A., & Arel, I. (2013). Low-rank approximations for conditional feedforward computation in deep neural networks. arXiv preprint arXiv:1312.4461.

[43] Bolukbasi, T., Wang, J., Dekel, O., & Saligrama, V. (2017). Adaptive neural networks for efficient inference. International Conference on Machine Learning, PMLR.

[44] Wang, X., Yu, F., Dou, Z. Y., et al. (2018). SkipNet: Learning dynamic routing in convolutional networks. Proceedings of the European conference on computer vision.

[45] Veit, A., & Belongie, S. (2018). Convolutional networks with adaptive inference graphs. Proceedings of the European Conference on Computer Vision.

[46] Rosenbaum, C., Klinger, T., & Riemer, M. (2017). Routing networks: Adaptive selection of non-linear functions for multi-task learning. arXiv preprint arXiv:1711.01239.

[47] Lewis, M., Sharma, S., & Zettlemoyer, L. (2020). Routing transformer. Transactions of the Association for Computational Linguistics, 9, 53-68.

[48] Clark, A., de Las Casas, D., Guy, A., et al. (2019). Open-ended learning leads to generally capable agents. arXiv preprint arXiv:1912.04676.

[49] Korthikanti, V., Casper, J., Lym, S., et al. (2022). Reducing activation recomputation in large transformer models. arXiv preprint arXiv:2205.05198.

[50] Ren, J., Rajbhandari, S., Aminabadi, R. Y., et al. (2021). ZeRO-Offload: Democratizing billion-scale model training. 2021 USENIX Annual Technical Conference.

[51] Huang, Y., Cheng, Y., Bapna, A., et al. (2019). GPipe: Efficient training of giant neural networks using pipeline parallelism. Advances in neural information processing systems, 32.

[52] Child, R., Gray, S., Radford, A., & Sutskever, I. (2019). Generating long sequences with sparse transformers. arXiv preprint arXiv:1904.10509.

[53] Kitaev, N., Kaiser, Ł., & Levskaya, A. (2020). Reformer: The efficient transformer. arXiv preprint arXiv:2001.04451.

[54] Zaheer, M., Guruganesh, G., Dubey, K. A., et al. (2020). Big bird: Transformers for longer sequences. Advances in Neural Information Processing Systems, 33, 17283-17297.

[55] He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep residual learning for image recognition. Proceedings of the IEEE conference on computer vision and pattern recognition.

[56] Dosovitskiy, A., Beyer, L., Kolesnikov, A., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

[57] Shaw, P., Uszkoreit, J., & Vaswani, A. (2018). Self-attention with relative position representations. Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics.

[58] Su, J., Lu, Y., Pan, S., et al. (2021). RoFormer: Enhanced transformer with rotary position embedding. arXiv preprint arXiv:2104.09864.

[59] Lewis, M., Liu, Y., Goyal, N., et al. (2020). BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.

[60] Zhang, J., Zhao, Y., Saleh, M., & Liu, P. (2020). PEGASUS: Pre-training with extracted gap-sentences for abstractive summarization. International Conference on Machine Learning, PMLR.

[61] Puigcerver, J., Riquelme, C., Mustafa, B., & Houlsby, N. (2020). Scalable transfer learning with expert models. arXiv preprint arXiv:2009.13239.

[62] Clark, A., De Las Casas, D., Guy, A., et al. (2022). Unified scaling laws for routed language models. International Conference on Machine Learning, PMLR.

[63] Mustafa, B., Riquelme, C., Puigcerver, J., et al. (2022). Multimodal contrastive learning with limoe: The language-image mixture of experts. arXiv preprint arXiv:2206.02770.

[64] Kenton, J. D. M. W. C., & Toutanova, L. K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of NAACL-HLT.

[65] Mnih, V., Heess, N., Graves, A., & Kavukcuoglu, K. (2014). Recurrent models of visual attention. Advances in neural information processing systems, 27.

[66] Ba, J., Mnih, V., & Kavukcuoglu, K. (2014). Multiple object recognition with visual attention. arXiv preprint arXiv:1412.7755.

[67] Eigen, D., Ranzato, M., & Sutskever, I. (2013). Learning factored representations in a deep mixture of experts. arXiv preprint arXiv:1312.4314.

[68] Gross, S., Ranzato, M., & Szlam, A. (2017). Hard mixtures of experts for large scale weakly supervised vision. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[69] Hazimeh, H., Zhao, Z., Chowdhery, A., et al. (2021). DSelect-k: Differentiable selection in the mixture of experts with applications to multi-task learning. Advances in Neural Information Processing Systems, 34.

[70] Roller, S., Sukhbaatar, S., Szlam, A., & Weston, J. (2021). Hash layers for large sparse models. Advances in Neural Information Processing Systems, 34.

[71] Huang, G., Chen, D., Li, T., et al. (2017). Multi-scale dense networks for resource efficient image classification. arXiv preprint arXiv:1703.09844.

[72] Panda, P., Sengupta, A., & Roy, K. (2016). Conditional deep learning for energy-efficient and enhanced pattern recognition. 2016 Design, Automation & Test in Europe Conference & Exhibition, IEEE.

[73] Caruana, R. (1997). Multitask learning. Machine learning, 28(1), 41-75.

[74] Ruder, S. (2017). An overview of multi-task learning in deep neural networks. arXiv preprint arXiv:1706.05098.

[75] Standley, T., Zamir, A., Chen, D., et al. (2020). Which tasks should be learned together in multi-task learning? International Conference on Machine Learning, PMLR.

[76] Andreas, J., Rohrbach, M., Darrell, T., & Klein, D. (2016). Neural module networks. Proceedings of the IEEE conference on computer vision and pattern recognition.

[77] Johnson, J., Hariharan, B., van der Maaten, L., et al. (2017). CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition.

[78] Schuster, T., Fisch, A., & Barzilay, R. (2021). Consistent accelerated inference via confident adaptive transformers. Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing.

[79] Liao, Y., Jiang, S., Yang, Z., & Culotta, A.

(2021). A stable and effective learning strategy for trainable greedy decoding. Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing.

[80] Geva, M., Schuster, R., Berant, J., & Levy, O. (2020). Transformer feed-forward layers are key-value memories. Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing.

[81] Wang, A., Singh, A., Michael, J., et al. (2018). GLUE: A multi-task benchmark and analysis platform for natural language understanding. arXiv preprint arXiv:1804.07461.

[82] Wang, A., Pruksachatkun, Y., Nangia, N., et al. (2019). SuperGLUE: A stickier benchmark for general-purpose language understanding systems. Advances in neural information processing systems, 32.

[83] Srivastava, R. K., Greff, K., & Schmidhuber, J. (2015). Training very deep networks. Advances in neural information processing systems, 28.

[84] Srivastava, R. K., Greff, K., & Schmidhuber, J. (2015). Highway Networks. arXiv preprint arXiv:1505.00387.

[85] Zagoruyko, S., & Komodakis, N. (2016). Wide residual networks. arXiv preprint arXiv:1605.07146.

[86] Dauphin, Y. N., Fan, A., Auli, M., & Grangier, D. (2017). Language modeling with gated convolutional networks. International conference on machine learning, PMLR.

[87] Gehring, J., Auli, M., Grangier, D., et al. (2017). Convolutional sequence to sequence learning. International conference on machine learning, PMLR.

[88] Zoph, B., & Le, Q. V. (2016). Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578.

[89] Pham, H., Guan, M. Y., Zoph, B., et al. (2018). Efficient neural architecture search via parameters sharing. International conference on machine learning, PMLR.

[90] Liu, H., Simonyan, K., & Yang, Y. (2018). DARTS: Differentiable architecture search. arXiv preprint arXiv:1806.09055.

[91] Strubell, E., Ganesh, A., & McCallum, A. (2019). Energy and policy considerations for deep learning in NLP. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.

[92] Schwartz, R., Dodge, J., Smith, N. A., & Etzioni, O. (2020). Green AI. Communications of the ACM, 63(12), 54-63.

[93] Henderson, P., Hu, J., Romoff, J., et al. (2020). Towards the systematic reporting of the energy and carbon footprints of machine learning. Journal of Machine Learning Research, 21(248), 1-43.

[94] Graves, A. (2016). Adaptive computation time for recurrent neural networks. arXiv preprint arXiv:1603.08983.

[95] Dehghani, M., Gouws, S., Vinyals, O., et al. (2018). Universal transformers. arXiv preprint arXiv:1807.03819.

[96] Guo, D., Rush, A. M., & Kim, Y. (2020). Parameter-efficient transfer learning with diff pruning. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.

[97] Michel, P., Levy, O., & Neubig, G. (2019). Are sixteen heads really better than one? Advances in neural information processing systems, 32.

[98] Fan, A., Grave, E., & Joulin, A. (2019). Reducing transformer depth on demand with structured dropout. arXiv preprint arXiv:1909.11556.

[99] McCarley, J. S. (2019). Pruning a bert-based question answering model. arXiv preprint arXiv:1910.06360.

[100] Kaya, Y., Hong, S., & Dumitras, T. (2019). Shallow-deep networks: Understanding and mitigating network overthinking. International Conference on Machine Learning, PMLR.

[101] Lakatos, G., Hrle, L., & Uhlich, S. (2020). A fully differentiable approach for solving doubly-stochastic MoE routing. arXiv preprint arXiv:2010.01692.

[102] Schwartz, R., Stanovsky, G., Swayamdipta, S., et al. (2020). The right tool for the job: Matching model and instance complexities. Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.

[103] Hu, E. J., Shen, Y., Wallis, P., et al. (2021). LoRA: Low-rank adaptation of large language models. arXiv preprint arXiv:2106.09685.

[104] Li, X. L., & Liang, P. (2021). Prefix-tuning: Optimizing continuous prompts for generation. Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing.

[105] Boyd, S., & Vandenberghe, L. (2004). Convex optimization. Cambridge university press.

[106] Miettinen, K. (2012). Nonlinear multiobjective optimization. Springer Science & Business Media.

[107] Ehrgott, M. (2005). Multicriteria optimization. Springer Science & Business Media.

[108] Kendall, A., Gal, Y., & Cipolla, R. (2018). Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. Proceedings of the IEEE conference on computer vision and pattern recognition.

[109] Chen, Z., Badrinarayanan, V., Lee, C. Y., & Rabinovich, A. (2018). GradNorm: Gradient normalization for adaptive loss balancing in deep multitask networks. International conference on machine learning, PMLR.

[110] Lin, X., Zhen, H. L., Li, Z., et al. (2019). Pareto multi-task learning. Advances in Neural Information Processing Systems, 32.

[111] Huang, G., Chen, D., Li, T., et al. (2017). Multi-scale dense networks for resource efficient image classification. arXiv preprint arXiv:1703.09844.

[112] Kaya, Y., Hong, S., & Dumitras, T. (2019). Shallow-deep networks: Understanding and mitigating network overthinking. International Conference on Machine Learning, PMLR.

[113] Li, L., Jin, K., Akbari, H., et al. (2019). Improved techniques for training adaptive deep networks. Proceedings of the IEEE/CVF International Conference on Computer Vision.

[114] Kourtis, D., & Tassiulas, L. (2020). A fair share approach to heterogeneous federated learning. arXiv preprint arXiv:2010.12229.

[115] Karimireddy, S. P., Kale, S., Mohri, M., et al. (2020). SCAFFOLD: Stochastic controlled averaging for federated learning. International Conference on Machine Learning, PMLR.

[116] Zellers, R., Bisk, Y., Schwartz, R., & Choi, Y. (2018). SWAG: A large-scale adversarial dataset for grounded commonsense inference. Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing.

[117] Bisk, Y., Zellers, R., Bras, R. L., et al. (2020). PIQA: Reasoning about physical commonsense in natural language. Proceedings of the AAAI Conference on Artificial Intelligence, 34(05), 7432-7439.

[118] Zhang, Y., & Yang, Q. (2017). A survey on multi-task learning. arXiv preprint arXiv:1707.08114.

[119] Thrun, S., & Pratt, L. (2012). Learning to learn. Springer Science & Business Media.

[120] Guo, C., Pleiss, G., Sun, Y., & Weinberger, K. Q. (2017). On calibration of modern neural networks. International Conference on Machine Learning, PMLR.

[121] Niculescu-Mizil, A., & Caruana, R. (2005). Predicting good probabilities with supervised learning. Proceedings of the 22nd international conference on Machine learning.

[122] Pope, R., Douglas, S., Chowdhery, A., et al. (2023). Efficiently scaling transformer inference. Proceedings of Machine Learning and Systems, 5.

[123] Kwon, W., Li, Z., Zhuang, S., et al. (2023).

Efficient memory management for large language model serving with pagedattention. Proceedings of the 29th Symposium on Operating Systems Principles.

[124] Welleck, S., Kulikov, I., Roller, S., et al. (2019). Neural text generation with unlikelihood training. arXiv preprint arXiv:1908.04319.

[125] Holtzman, A., Buys, J., Du, L., et al. (2019). The curious case of neural text degeneration. arXiv preprint arXiv:1904.09751.

[126] Chen, T., Xu, B., Zhang, C., & Guestrin, C. (2016). Training deep nets with sublinear memory cost. arXiv preprint arXiv:1604.06174.

[127] Jain, P., Jain, A., Nrusimha, A., et al. (2020). Checkmate: Breaking the memory wall with optimal tensor rematerialization. Proceedings of Machine Learning and Systems, 2.

[128] Aminabadi, R. Y., Rajbhandari, S., Zhang, M., et al. (2022). DeepSpeed Inference: enabling efficient inference of transformer models at unprecedented scale. arXiv preprint arXiv:2207.00032.

[129] Dao, T., Fu, D. Y., Ermon, S., et al. (2022). FlashAttention: Fast and memory-efficient exact attention with IO-awareness. Advances in Neural Information Processing Systems, 35.

[130] Rabe, M. N., & Staats, C. (2021). Self-attention does not need $O(n^2)$ memory. arXiv preprint arXiv:2112.05682.

[131] Andreas, J., Klein, D., & Levine, S. (2017). Modular multitask reinforcement learning with policy sketches. International Conference on Machine Learning, PMLR.

[132] Rosenbaum, C., Klinger, T., & Riemer, M. (2017). Routing networks: Adaptive selection of non-linear functions for multi-task learning. arXiv preprint arXiv:1711.01239.

[133] Bengio, Y., Louradour, J., Collobert, R., & Weston, J. (2009). Curriculum learning. Proceedings of the 26th annual international conference on machine learning.

[134] Graves, A., Bellemare, M. G., Menick, J., et al. (2017). Automated curriculum learning for neural networks. International conference on machine learning, PMLR.

[135] Portelas, R., Colas, C., Weng, L., et al. (2020). Automatic curriculum learning for deep RL: a short survey. arXiv preprint arXiv:2003.04664.

[136] Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. Machine learning, 8(3-4), 229-256.

[137] Schulman, J., Wolski, F., Dhariwal, P., et al. (2017). Proximal policy optimization algorithms. arXiv preprint arXiv:1707.06347.

[138] Mnih, V., Kavukcuoglu, K., Silver, D., et al. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533.

[139] Finn, C., Abbeel, P., & Levine, S. (2017). Model-agnostic meta-learning for fast adaptation of deep networks. International Conference on Machine Learning, PMLR.

[140] Nichol, A., Achiam, J., & Schulman, J. (2018). On first-order meta-learning algorithms. arXiv preprint arXiv:1803.02999.

[141] Jouppi, N. P., Young, C., Patil, N., et al. (2017). In-datacenter performance analysis of a tensor processing unit. Proceedings of the 44th annual international symposium on computer architecture.

[142] Markidis, S., Der Chien, S. W., Laure, E., et al. (2018). Nvidia tensor core programmability, performance & precision. 2018 IEEE international parallel and distributed processing symposium workshops, IEEE.

[143] Graphcore. (2021). IPU Programmer's Guide. Graphcore Ltd.

[144] Robbins, H., & Monro, S. (1951). A stochastic approximation method. The annals of mathematical statistics, 400-407.

[145] Bottou, L. (2010). Large-scale machine

learning with stochastic gradient descent. Proceedings of COMPSTAT'2010, Springer.

[146] Radford, A., Kim, J. W., Hallacy, C., et al. (2021). Learning transferable visual models from natural language supervision. International Conference on Machine Learning, PMLR.

[147] Ramesh, A., Pavlov, M., Goh, G., et al. (2021). Zero-shot text-to-image generation. International Conference on Machine Learning, PMLR.

[148] Dosovitskiy, A., Beyer, L., Kolesnikov, A., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. arXiv preprint arXiv:2010.11929.

[149] Team, G., Alayrac, J. B., Donahue, J., et al. (2021). Flamingo: a visual language model for few-shot learning. Advances in Neural Information Processing Systems, 35.

[150] Chen, X., Wang, X., Changpinyo, S., et al. (2023). Pali: A jointly-scaled multilingual language-image model. arXiv preprint arXiv:2209.06794.

[151] Pope, R., Douglas, S., Chowdhery, A., et al. (2023). Efficiently scaling transformer inference. Proceedings of Machine Learning and Systems, 5.

[152] Shazeer, N. (2020). GLU variants improve transformer. arXiv preprint arXiv:2002.05202.

[153] Press, O., Smith, N. A., & Lewis, M. (2022). Train short, test long: Attention with linear biases enables input length extrapolation. arXiv preprint arXiv:2108.12409.

[154] Tay, Y., Dehghani, M., Bahri, D., & Metzler, D. (2020). Efficient transformers: A survey. ACM Computing Surveys, 55(6), 1-28.

[155] Qiu, J., Ma, H., Levy, O., et al. (2019). Blockwise self-attention for long document understanding. arXiv preprint arXiv:1911.02972.

[156] Korthikanti, V., Casper, J., Lym, S., et al. (2022). Reducing activation recomputation in large transformer models. arXiv preprint arXiv:2205.05198.

[157] Rae, J. W., Potapenko, A., Jayakumar, S. M., & Lillicrap, T. P. (2019). Compressive transformers for long-range sequence modelling. arXiv preprint arXiv:1911.05507.

[158] Beltagy, I., Peters, M. E., & Cohan, A. (2020). Longformer: The long-document transformer. arXiv preprint arXiv:2004.05150.

[159] Zoph, B., & Le, Q. V. (2016). Neural architecture search with reinforcement learning. arXiv preprint arXiv:1611.01578.

[160] Real, E., Aggarwal, A., Huang, Y., & Le, Q. V. (2019). Regularized evolution for image classifier architecture search. Proceedings of the aaai conference on artificial intelligence, 33(01).

[161] Patterson, D., Gonzalez, J., Holzle, U., et al. (2021). Carbon emissions and large neural network training. arXiv preprint arXiv:2104.10350.

[162] Strubell, E., Ganesh, A., & McCallum, A. (2019). Energy and policy considerations for deep learning in NLP. Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.

[163] Henderson, P., Hu, J., Romoff, J., et al. (2020). Towards the systematic reporting of the energy and carbon footprints of machine learning. Journal of Machine Learning Research, 21(248), 1-43.