Why Routers Freeze: Infinite Width Learning Dynamics for Mixture of Experts

Anish Dhir[1] **Volkan Cevher**[2] **Leena Chennuru Vankadara**[1]

[1]Gatsby Computational Neuroscience Unit, University College London

[2]LIONS, École Polytechnique Fédérale de Lausanne (EPFL)

Abstract

Mixture-of-Experts (MoE) architectures are central to large-scale deep learning, relying on sparse execution and expert specialisation. Despite their success, their large-scale training dynamics remain poorly understood, and training is often unstable. While scaling limits such as infinite-width theory have clarified the behavior of dense models, the scaling behavior of MoEs remains largely unexplored. To this end, we derive the infinite-width limits of MoE architectures under fixed number of experts with both soft and Top-K routing, trained with SGD and Adam, using Tensor Programs. Our results show that under the Standard Parameterisation (SP), router updates freeze after one step of training in both soft and Top-K routing MoEs. This provides, at least in part, a principled explanation for the widespread use of auxiliary losses such as load-balancing or z-losses in MoE training. In dense networks, Maximal Update heuristics allow for stable and non-vanishing feature updates. However, in MoEs the Maximal Update heuristic with soft-routing and softmax gating causes the experts to collapse to the same distribution and router gradients to vanish. While sigmoid allows non-rivial router evolution, they still fail to prevent expert collapse. This leads to fixed expert soft-routing MoEs unable to ensure stability, feature learning, and expert specialisation in the infinite-width limit within the common parametrisation framework. We then show that, beyond computational efficiency, Top-K also acts as a symmetry breaking mechanism allowing for both feature learning and specialisation. We empirically validate these predictions, observing close agreement between finite-width training dynamics and the predicted scaling behaviour.

1 Introduction

Mixture of Experts (MoE) architectures partition a parameter space into distinct “experts", that specialise on different areas of the input space. Here, a router maps a given input to gating values that weight the expert outputs, a process that encourages specialisation through training [19, 12]. Sparse execution of the experts (with only non-zero weights) also allows for decoupling parameter count from compute [32]. This conditional computation, in combination with specialisation, has proven a key driver for scaling machine learning systems [33, 21, 9, 38, 17, 13].

Despite their empirical success, MoEs are notoriously difficult to train. A wide range of failure modes have been noted, particularly when scaling the model. The most prominent of these is the observation that router learning fails leading to failure in expert specialisation — with either experts learning similar representations, or most of the tokens being routed to a few experts [46, 5, 28, 14, 11]. Existing mitigation strategies, such as auxiliary losses [46, 14, 36] and different router activations [27, 7, 37], are largely based on empirical heuristics. Consequently, the mechanisms that underlie

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

these failure modes remain poorly understood, with theory providing limited guidance on how to scale MoEs reliably.

Recently, our understanding of the training dynamics of *dense neural networks* has significantly improved through infinite-width analysis [26, 40, 2, 25, 44, 40, 42]. These limits can be used to characterise quantities such as stability, feature learning, and optimisation behavior as the width n grows. For example, these limits were used to show that the Standard Parametrisation (He init [18] and global learning rate) dense network can become unstable or its features stop learning, as the width increases [41]. Further, they have resulted in principled ways of scaling important quantities with width (e.g. learning rate), resulting in the mean-field [25, 30, 6] or Maximum Update parameterisation (μP) [41]. An important empirical consequence of such analysis is the ability for zero-shot hyperparameter transfer across widths for multi-layer perceptrons [43, 10]. Similar infinite-width analyses have been extended to other architectures [31, 2, 35, 15].

Inspired by this line of work, we derive the infinite-width limits ($n \to \infty$) of MoEs at initialisation and during training, characterising their finite-time training behaviour using the Tensor Program framework [40–42]. Importantly, in practice memory overheads lead to architectures where the number of experts m is lower than the width n [21, 9, 38]. To reflect this, and the practical considerations for hyperparameter transfer, where width can be scaled more than number of experts, we focus our analysis on fixed number of experts. Our contributions are as follows:

- We derive the infinite-width limits of soft MoEs under softmax or sigmoid gating, and sparse MoEs with Top-K gating in the regime where network width grows while the number of experts remains fixed.
- For soft-routing MoEs, we show that under Standard Parametrisation (SP) the network becomes unstable and the router probabilities saturate, leading to a frozen router as width goes to ∞. Under a broader class of parametrisations, we then derive μP-MoE which uniquely satisfies stability, feature-learning and faithfulness with Adam, similar to μP for dense networks [41, 42].
- However, with μP-MoE which satisfies stability and feature learning in the soft routing case, the router assigns the same weights across experts, causing the experts to converge to an identical distribution. This leads to a collapse in expert diversity as the width grows. In this case, the commonly used softmax router activation nullifies the gradient, leading to no router updates, whereas other activations such as sigmoid allow the router to update, albeit without expert specialisation. Here, adding a router bias/noise term prevents the above router collapse.
- While problems with SP persist in the Top-K case, we show that the Top-K operator alone prevents the router collapse observed in soft routing and thus could allow for expert specialisation with μP-MoE (along with stability and feature learning), highlighting the utility of Top-K beyond computational considerations.
- Finally, we empirically validate all theoretical predictions, including router saturation and failure of specialisation. We show that the scaling enables hyperparameter transfer in Top-K MoEs in line with Yang et al. [43].

2 Setup and Background

In this work, we are concerned with finding the infinite-width limit of MoE architectures, and understanding the implications that derive from it. We begin by formalising several MoE architectures (softmax/sigmoid, dense/sparse), and specify desirable asymptotic properties.

2.1 Mixture of Experts

We first consider soft routing MoE networks [19]. In this architecture, a standard MLP layer is replaced by a collection of m expert sub-networks. Their outputs are aggregated via a gating mechanism that assigns weights based on a separate router network. Throughout training, these gating weights evolve according to the relative performance of each expert across the input manifold, facilitating functional specialisation.

Most MoE implementations use a router with a softmax activation [19] which encourages competition between experts. However, this competition could harm specialisation [5]. As such, recently there has been interest in alternative routing techniques which might further improve specialisation. Sigmoid gating, for instance, has been shown to achieve better performance in certain tasks [27, 7, 8]. In this work, we provide a rigorous analysis of both softmax and sigmoid gating functions.

Definition 2.1 (Soft MoE Network)**.** Given an input $\boldsymbol{\xi} \in \mathbb{R}^{d_{\text{in}}}$, a Soft MoE network of width n with m experts is defined by the forward pass

$$\begin{aligned} \mathbf{h}^1 &= \mathbf{W}^1\boldsymbol{\xi} \in \mathbb{R}^n, & \mathbf{h}^2 &= \mathbf{W}^2\mathbf{h}^1 + \mathbf{b}^2 \in \mathbb{R}^m, \\ \tilde{\mathbf{h}}^2 &= \mathrm{G}(\mathbf{h}^2) \in \mathbb{R}^m, & \mathbf{h}^{3,k} &= \mathbf{W}^{3,k}\mathbf{h}^1 \in \mathbb{R}^n, \\ \mathbf{h}^3 &= \sum_{k=1}^m \tilde{h}_k^2\, \mathbf{h}^{3,k} \in \mathbb{R}^n, & \mathbf{f} &= \mathbf{W}^4\mathbf{h}^3 \in \mathbb{R}^{d_{\text{out}}}. \end{aligned}$$

where $\mathrm{G} \in \{\mathrm{Softmax},\, \mathrm{Sigmoid}\}$ is the gating function.

We refer to $\mathbf{W}^2$, $\mathbf{h}^2$ as the *router* weights and logits, and $\mathbf{W}^{3,k}$, $\mathbf{h}^{3,k}$ as the *expert* weights and logits for expert k. We also include a 'bias' term for the router logits $\mathbf{b}^2$, which is a generalisation that reflects choices such as a learnt bias, a router noise term [32], or load balancing bias terms [36]. Note that Definition 2.1 contains an MoE block (Definition 2.10) with an input and output layer. This ensures our analysis contains the smallest possible components, and we show how our results extend to MoE blocks in realistic transformers in Section 2.5.

Soft routing activates every expert in a forward pass, even if the router assigns certain experts very low weights. Sparse routing, on the other hand, only activates a subset of experts, leading to a smaller compute budget for each forward pass. Top-K routing MoE networks, for example, only execute experts with the Top-K probabilities [32], decoupling the compute from the number of parameters.

Definition 2.2 (Top-K MoE Networks)**.** A Top-K MoE network replaces the soft routing in Definition 2.1 with a sparse routing,

$$\begin{aligned} \hat{\mathbf{h}}^2 &= \mathrm{T}(\mathbf{h}^2), \text{ where } \mathrm{T}(\mathbf{h}^2)_k = \begin{cases} (\mathbf{h}^2)_k & \text{if } k \in \mathrm{TopK}(\mathbf{h}^2) \\ -\infty & \text{otherwise,} \end{cases} \\ \tilde{\mathbf{h}}^2 &= \mathrm{G}(\hat{\mathbf{h}}^2), \end{aligned}$$

where $\mathrm{G} \in \{\mathrm{Softmax}, \mathrm{Sigmoid}\}$, TopK returns the top K indices, and only experts k with $(\tilde{\mathbf{h}}^2)_k > 0$ are executed. We follow common practice and use the straight-through estimator for the gradients [32].

In modern MoEs, due to memory constraints, the model width n tends to dominate the number of experts m ($m \ll n$) [21, 9, 38, 34]. Hence, we consider the limit where the width increases $n \to \infty$, while the number of experts m remain fixed. This regime is particularly relevant for width-based hyperparameter transfer, where one tunes hyperparameters on a smaller-width model and transfers them to a larger-width model with the same architectural skeleton [43]. It also isolates the effect of increasing expert capacity from the separate effect of increasing the number of experts. A joint limit in which m grows with n would change the router dimension, the Top-K selection problem, and the softmax normalisation, and therefore constitutes a different asymptotic regime. We leave such joint expert-width limits to future work. We also fix other scaling dimensions, commonly done in the Tensor program framework Yang and Hu [41].

Setting 2.3. We study the scaling regime under fixed number of experts m and the common Tensor program setting of fixing other scaling dimensions such as batch size, training time, and depth.

Remark 2.4 (Linear Experts). To reduce cumbersome notation, we focus our theoretical analysis on linear experts. Our analysis can be extended to any networks with pseudo-Lipschitz nonlinearities and does not affect the validity of our conclusions (see Yang and Hu [41, H.15] and Appendix G.2). Furthermore, we validate our theoretical findings through experiments conducted exclusively with non-linear experts to ensure practical relevance.

2.2 Parametrisation and Scaling

To characterise how MoE networks behave with increasing width, we parametrise quantities that govern the learning of the network as a function of the model width n. Motivated by the *abcd* parametrisation for dense networks [42], we define *bcd* parametrisation for MoEs.

Definition 2.5 (bcd-Parametrisation)**.** For each layer ℓ, the weights are initialized as $W_{ij}^{\ell} \sim \mathcal{N}(0, n^{-2b_\ell})$. The optimization updates are defined as:

$$\textbf{SGD: } \mathbf{W}^{\ell} \leftarrow \mathbf{W}^{\ell} - \eta n^{-c_\ell} \nabla_{\mathbf{W}^{\ell}} \mathcal{L}, \qquad \textbf{Adam: } \mathbf{W}^{\ell} \leftarrow \mathbf{W}^{\ell} - \eta n^{-c_\ell} \left(\frac{\widehat{m}_\ell}{\sqrt{\widehat{v}_\ell} + n^{-d_\ell}\epsilon} \right),$$

where $\widehat{m}_\ell, \widehat{v}_\ell$ are the gradient moment estimates (defined in Definition A.5) and η, ϵ are width-independent constants.

For measuring the typical entry scale for vectors $\mathbf{v} \in \mathbb{R}^n$, we use the root-mean-squared norm throughout $n^{-1/2}\|\mathbf{v}\|_2$. We use Bachmann-Landau notation $(\Theta, \mathcal{O}, \Omega)$ to denote scaling of entries with respect to width n, keeping everything else constant.

2.3 Desirable Asymptotic Behaviours

We define four asymptotic properties that a mixture-of-experts (MoE) network should ideally satisfy in the infinite-width limit. Building on the asymptotic framework of Yang and Hu [41], Yang and Littwin [42], which formalises stability, maximal feature learning, and faithfulness, we introduce a fourth desideratum specific to MoE architectures: expert specialisation. We aim to identify the class of *bcd*-parameterizations under which network training dynamics satisfy these asymptotic properties. We denote the value of a quantity at training step t using a subscript, e.g. $\mathbf{h}_t^{\ell}$.

We first require that activations do not diverge with increasing width.

Definition 2.6 (Stability)**.** A network is *stable* if, for every layer ℓ, the activations satisfy $\mathbf{h}^{\ell} = \mathcal{O}(1)$ at initialisation and throughout training.

Next, to ensure feature learning is taking place, we require that the change in activations $\mathbf{h}_t - \mathbf{h}_{t-1}$ at each time step do not vanish with increasing width [41]. Activation updates can be decomposed into two terms

$$\mathbf{h}_t^{\ell} - \mathbf{h}_{t-1}^{\ell} = \underbrace{\mathbf{W}_{t-1}^{\ell}(\mathbf{h}_t^{\ell-1} - \mathbf{h}_{t-1}^{\ell-1})}_{\text{Propagating Updates}} + \underbrace{(\mathbf{W}_t^{\ell} - \mathbf{W}_{t-1}^{\ell})\mathbf{h}_t^{\ell-1}}_{\text{Effective Updates}},$$

where the propagating updates are contributions from the update to the previous layer features, and effective updates are the contribution due to the current weight updates. *Maximal* feature learning requires that both the above terms do not vanish with increasing width. For conciseness we refer to this as simply feature learning throughout.

Definition 2.7 (Maximal Feature Learning)**.** A layer ℓ exhibits *maximal feature learning* if both effective and propagating updates after one step of training scale $\Omega(1)$ [41].

For Adam updates, we require that the gradients have a non-trivial $\Theta(1)$ effect on the updates, a condition referred to as *faithfulness* [42].

Definition 2.8 (Adam Faithfulness)**.** We say a *bcd*-parametrisation is *faithful* if the inputs to the Adam update function have scale $\Theta(1)$ for every layer's parameter updates.

Finally, we introduce a desideratum specific to MoEs. Expert specialisation is a central tenet of MoE architectures; accordingly, we require that such specialisation does not cease at infinite width. A *necessary* condition for expert specialisation is that, for a given input, the router exhibits a preference for certain experts over others. We use this to diagnose regimes where the experts cannot specialise.

Definition 2.9 (Expert Specialisation)**.** We say that a *bcd*-parametrisation exhibits *expert specialisation* only if $\mathbf{h}_t^2 \neq C_t \mathbf{1}$ for any constants C_t for all steps $t > 0$.

2.4 Tensor Programs for Infinite Width Limits

Our goal is to characterise the behavior of MoE architectures as the width n approaches infinity. To address this, we leverage the Tensor Programs (TP) framework [40], which provides a general recipe for computing infinite-width limits for networks and learning dynamics expressible as a restricted algebraic program. Tensor Programs are constructed by specifying a collection of initial (random) matrices $\mathbb{R}^{n \times n}$, vectors $\mathbb{R}^n$, and scalars $\mathbb{R}$. New vectors and scalars are generated through a restricted

set of operations: matrix multiplications, non-linearities, or averages. Under mild regularity of the non-linearities (pseudo-Lipschitz) the Master theorem implies that any vector variable v becomes asymptotically iid and can be associated with a scalar random variable Z^v describing the limiting distribution of the coordinates of v. These distributions can be computed through a symbolic algebra determined by the operations of the program. Tracking correlations between the vectors reduces to tracking correlations between their associated coordinate distribution.

2.5 Extending to MoE blocks in Transformers

For ease of exposition, in the MoE network, we consider one MoE 'block'.

Definition 2.10 (MoE Block)**.** We define an MoE block as,

$$\mathbf{h}^2 = \mathbf{W}^2\mathbf{h}^1 + \mathbf{b}^2 \in \mathbb{R}^m, \qquad \tilde{\mathbf{h}}^2 = \mathrm{G}(\mathbf{h}^2) \in \mathbb{R}^m, \tag{1}$$

$$\mathbf{h}^{3,k} = \mathbf{W}^{3,k}\mathbf{h}^1 \in \mathbb{R}^n, \qquad \mathbf{h}^3 = \sum_{k=1}^{m} \tilde{h}_k^2\, \mathbf{h}^{3,k} \in \mathbb{R}^n, \tag{2}$$

which takes in a width n vector $\mathbf{h}^1$ and outputs a width n vector $\mathbf{h}^3$.

We note that the results in this work hold if the MoE block is part of a standard transformer architecture as long as components of the transformer are parametrised according to μP [43], for example

$$u^\ell = x^\ell + \mathrm{Attn}^\ell\left(\mathrm{Norm}_1^\ell(x^\ell)\right), \tag{3}$$

$$x^{\ell+1} = u^\ell + \mathrm{MoE}^\ell\left(\mathrm{Norm}_2^\ell(u^\ell)\right). \tag{4}$$

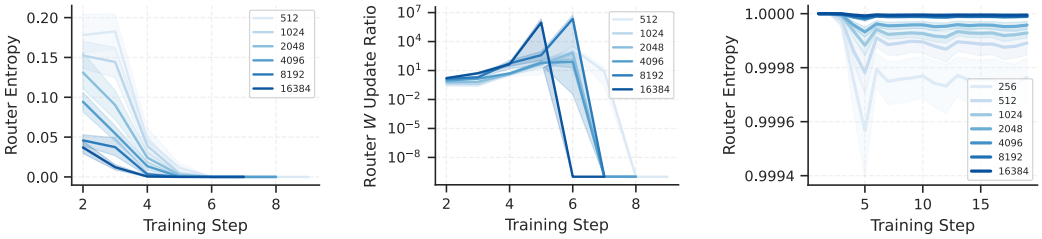
This is because under μP transformer components the scale of the inputs and gradients flowing into the MoE block, and the form of the Tensor Program updates, is the same when MoE block is part of the MoE network (Definition 2.1) or a transformer block. This is also true MoE blocks as a part of L transformer layers. We carry out analysis for the MoE network in Definition 2.1 for ease of exposition but show that our results hold for MoE blocks in transformer architectures in Appendix G.1.

3 Related Work

MoEs were originally introduced to encourage specialisation by decomposing the parameter space into modular experts [19]. Lately, their application is driven by the ability to decouple parameter count from compute cost via sparse routing [32, 21]. Despite their efficiency, MoEs are notoriously difficult to train compared to their dense counterparts. Practitioners frequently encounter exploding router logits [46, 29] and representation collapse — where experts become redundant or underutilised [5, 28]. Mitigation of these issues has largely come from empirical heuristics, including auxiliary losses such as the router z-loss [46] and load-balancing regularisers [14], and alternative gating mechanisms, such as sigmoid and ReLU routing [27, 7, 37]. While effective in practice, these solutions are often ad-hoc, and the underlying theoretical mechanisms driving these instabilities remain poorly understood.

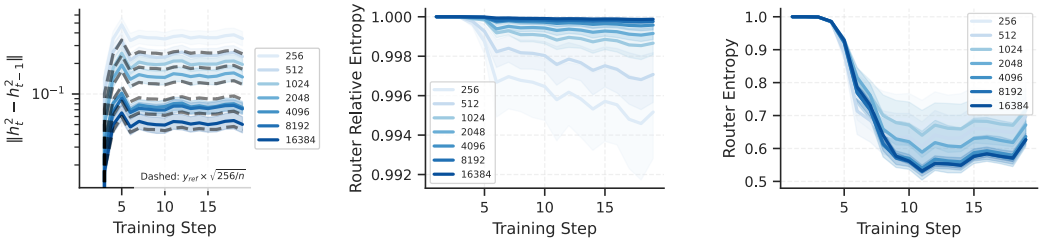
In contrast to the heuristic fixes used in MoEs, the training dynamics of dense neural networks have been rigorously characterised through infinite width limits. Early work focused on the "lazy" regime (e.g. NNNGP, NTK), where weights stay close to initialization and feature learning is suppressed [20, 23, 26]. More recently, the Tensor Programs framework [40, 41] and related mean-field approaches [25, 2] have characterized the "feature learning" (maximal update) limit. This theory led to the μP, which prescribes layer-wise scaling rules to ensure stable feature learning and enables zero-shot hyperparameter transfer across model scales [41, 43]. These limits have been extended to various architectures [3, 15, 35], and for scaling other quantities such as depth [10, 45]. An important empirical consequence of such analysis is the ability for zero shot hyperparameter transfer across width and depth [43, 4].

While independent and concurrent empirical work has attempted to apply μP directly to MoEs [24], they do not derive the limiting dynamics and thus do not characterise the breakdown of expert specialisation without the router bias, or that Top-K acts as a symmetry breaker. Concurrent work by Jiang et al. [22] studies hyperparameter transfer for transformer MoEs and proposes scaling rules for



(a) SP Softmax Soft (b) SP Softmax Soft (c) μP-MoE Softmax Soft

Figure 1: (a) and (b) show router collapse under SP softmax with soft routing. (a) In SP Softmax-Soft MoEs, normalized router entropy decreases toward zero as width increases, indicating collapse to a near one-hot gate. (b) This saturation drives the Softmax Jacobian toward zero and yields a near-zero relative router-weight update norm, $\|\mathbf{W}_t - \mathbf{W}_{t-1}\|_1 / \|\mathbf{W}_{t-1}\|_1$. (c) μP-MoE with softmax routing: normalised router entropy remains at 1.0 (maximum) throughout training as width increases, confirming uniform expert allocation.



(a) μP-MoE Softmax Soft (b) μP-MoE Sigmoid Soft (c) μP-MoE Sigmoid Soft

Figure 2: (a) For μP-MoE with softmax routing the router feature update norm vanishes with width, strictly following the theoretical prediction of $\Theta(n^{-1/2})$, predicted from width 256 (dashed lines). (b) For μP-MoE with sigmoid routing, the normalised entropy of the averaged router probability remains high, indicating experts do not specialise distinctively. (c) Unlike the softmax case, the sigmoid router entropy evolves over time, confirming that gradients are non-zero but identical across experts.

transferring learning rates and initialisation scales as width, depth, expert hidden size, and the number of experts are varied. Our work is complementary. First, the two limits are not interchangeable: Jiang et al. [22] take a joint proportional limit under *gradient flow dynamics* in which width, depth, expert size, and expert count diverge together, whereas we study a width-only Tensor Program limits for both soft and Top-K, sigmoid and softmax routing under SGD and Adam with a fixed number of experts. Since iterated limits do not commute in general, their joint limit does not recover the fixed-expert width limit we study. Second, the two limits are qualitatively different in their predictions: in the proportional limit of Jiang et al. [22], experts become asymptotically independent, whereas in our width limit experts remain strongly correlated. This fixed-expert regime lets us isolate router-specific degeneracies: under SP the router logits saturate and router gradients vanish, while under stable μP-heuristic scaling soft routing remains symmetric across experts unless an explicit symmetry-breaking mechanism is present.

4 Scaling Soft MoEs

We first present insights from the derived infinite width limit of soft MoE networks (Appendix A). Our results here hold for both SGD and Adam.

4.1 Standard Parameterisation: Router Collapse

The Standard Parametrisation (SP), which is commonly used in practice, corresponds to initialising weight matrices with $b_1 = 0$, $b_\ell = 0.5$ for $\ell > 1$ [18], and using a global learning rate scaling $c_\ell = c$ [41]. The initialisation value ensures all activations in the first forward pass scale $\Theta(1)$ [41]. Our first

result states that, with increasing width, the SP MoEs are not stable — resulting in the router weights not learning. We derive this result in Appendix B.1.

Proposition 4.1. *(Router collapse in SP soft MoE networks) Consider a soft MoE network with softmax or sigmoid gating (Definition 2.1) parametrised according to SP. For any choice of c under which $\mathbf{h}^1$ admits maximal feature learning ($\mathbf{h}_1^1 - \mathbf{h}_0^1$) scale $\Theta(1)$. Then as $n \to \infty$:*

1. *The router probability $\tilde{\mathbf{h}}_1^2$ converges to a one-hot vector for Softmax-Soft MoEs and a binary vector for Sigmoid-Soft MoEs after one step.*
2. *The router gradients $\nabla_{W^2}\mathcal{L}$ vanish after one step.*

Mechanism of failure: Under SP initialisation, the router weight entries scale $\Theta(n^{-1/2})$. After one step of SGD (the same holds for Adam), the propagating update of $\mathbf{h}^2$ is $\sum_{j=1}^{n}(\mathbf{W}_0^2)_{:,j}(\mathbf{h}_1^1 - \mathbf{h}_0^1)_{j,:}$. Crucially, as the gradients for the feature updates depend on the forward pass weights, $(\mathbf{h}_1^1 - \mathbf{h}_0^1)_{j,:}$ is *correlated* with $(\mathbf{W}_0^2)_{:,j}$, and their sum scales as n. Thus, as the scale of $(\mathbf{h}_1^1 - \mathbf{h}_0^1)$ is $\Theta(1)$, the propagating updates scale as $\Theta(n^{1/2})$. This results in $\mathbf{h}_1^2$ to diverging with increasing width — softmax activation yields a one-hot gate, whereas the sigmoid activation yields a binary gate. In both cases the Jacobian $\partial\tilde{\mathbf{h}}_1^2/\partial\mathbf{h}_1^2$ vanishes, and thus the router weights cease to update. Thus, the router expresses a strong preference for certain experts but it's weights do not update further, and no meaningful specialisation occurs.

This result also provides a *partial explanation* for the use of additional regularisation terms like the z-loss [46] and load balancing losses [14]. Router instability as the scale is increased was noted by Fedus et al. [14], and the z-loss was introduced to explicitly penalise large logit values. Whereas, router probability saturating can make load imbalances likely necessitating explicit load balancing losses to reduce router saturation and allow for specialisation to occur. As such, these auxiliary losses can be seen as being explicitly designed to prevent the kind of collapse described in Proposition 4.1.

Validation of SP failure (Proposition 4.1): In Figure 1 (a), we show that as the width of the network increases, the router entropy (normalised by the maximum possible entropy) falls to zero — indicating that the router probabilities converge to a deterministic one-hot vector (router collapse). Consequently, in Figure 1 (b), we observe that the relative update norm of the router weights falls to zero, confirming that the router becomes frozen in this collapsed state.

4.2 μP-MoE Scaling: Stability without Specialisation.

By deriving infinite-width limits, we derive the parametrisation values that satisfy stability and faithfulness in MoEs trained with SGD and Adam (Appendix B.2). Whereas only sigmoid routing can exhibit feature learning — for softmax routing there is no *bcd* parametrisation that exhibits feature learning. The parametrisation values match the μP Heuristic for dense networks [43], with the router scaling similar to the output function, and each expert scaling similar to that of dense $n \times n$ layers. The parametrisation values denoted μP-MoE are given in Table 1 for SGD and Table 2 for Adam.

While this scaling successfully resolves the instability issues of SP, it introduces a converse failure mode in the absence of the router logit noise/bias $\mathbf{b}^2$. To allow feature learning in the router, the weights must be initialized to scale as $\Theta(n^{-1})$ (Table 1). In the infinite-width limit, this forces the router logits to converge deterministically, resulting in a router with symmetric scores (across experts) at initialisation. Crucially, without a bias term, we prove that this symmetry cannot be broken during training with either SGD or Adam for any finite training horizon. Consequently, the router assigns identical scores to each expert throughout training, *preventing expert specialisation* rendering the MoE structure redundant.

Proposition 4.2. *(Lack of expert specialisation with μP-heuristic soft MoE networks without router bias) Consider a soft MoE network without a router bias $\mathbf{b}^2$ (Definition 2.1) parametrised according to μP-MoE (Table 1). As $n \to \infty$:*

- *At initialisation ($t = 0$), the router scores become deterministic and identical across experts $\tilde{\mathbf{h}}_0^2 \to \frac{1}{m}\mathbf{1}$ for Softmax and $\tilde{\mathbf{h}}_0^2 \to \frac{1}{2}\mathbf{1}$ for Sigmoid.*
- *For all training steps $t \geq 0$, router scores remain symmetric across all experts: $\tilde{\mathbf{h}}_t^2 = C_t\mathbf{1}$ for some constant C_t, and all experts remain identically distributed.*

The proof for this is in Appendix B.2. Two mechanisms drive this failure:

Symmetric initialisation of router and experts: As stated before, μP-MoE router initialises deterministically to $\tilde{\mathbf{h}}_0^2 \propto \mathbf{1}$. Further, the experts weights are all initialised from the same distribution, resulting in identically distributed experts.

Symmetric gradients: A consequence of identically distributed experts is that the downstream gradient signal $\nabla_{h_k^2}\mathcal{L}$ *converges, almost surely to a deterministic constant vector* (Appendix A.2). Consequently, all router logits receive deterministic, identical updates and evolve synchronously throughout training. As a result, the router scores remain equal across experts at all times. Since the routing distribution is symmetric, the gradients flowing into each expert are also identically distributed, ensuring that the experts remain identically distributed and fail to specialise.

Moreover, when the softmax gating function is used, this results in the the softmax Jacobian nullifying the gradient into the router. Thus, in Softmax-Soft MoEs, router weights have vanishing gradients causing a failure of feature learning in the router.

Proposition 4.3. *(Vanishing router gradient with μP-heuristic Softmax-Soft MoE networks) Consider a Softmax-Soft MoE network without a router bias (Definition 2.1) parametrised according to the μP-MoE. As $n \to \infty$, the gradient with respect to the router weights $\nabla_{W^2}\mathcal{L}$ vanishes.*

In both cases (SP and μP-MoE) the MoE architecture fail to satisfy the asymptotic desiderata. While μP-MoE satisfies stability (and feature learning for sigmoid routing) and faithfulness for Adam, our result, derived using unscaled quantities, holds for any stable parametrisation of soft MoEs (Remark B.8). Thus, under Setting 2.3, soft MoEs do not satisfy our desiderata under any *bcd* parameterisation.

Router bias prevents router stagnation. At initialisation, the router scores are determined by the initialisation of the router bias. If the router bias is initialised from a distribution, then the router scores are not equal. This results in different gradients for each expert and thus the expert distributions differ — preventing router stagnation and potentially helping specisalisation.

Validation of μP-MoE specialisation failure without router bias: We show in Figure 1 (c), that the softmax router entropy converges to one as the width increases — implying that the router probabilities remain uniform throughout training. Whereas Figure 1 (d) confirms that the average norm of the router updates decreases with width according to our predicted scale of $n^{-1/2}$ (dashed lines predicted from width 256). For sigmoid routing, in Figure 2 (a) we plot the router entropy of the averaged (across experts) router probability. A value close to one indicates that the router is assigning the same weight to each expert. Crucially, Figure 2 (b) shows that, unlike the Softmax case, the router entropy *does* change over time. This confirms our theoretical finding: the router receives non-zero gradients, but these gradients are the same for each router logit — the router evolves but maintains identical expert weights throughout training.

5 Scaling Sparse MoEs

We now turn to sparse routing via Top-K gating, which is a common design choice in large-scale MoE models [32, 21]. Unlike soft routing, Top-K gating induces a discrete, input-dependent sparsification of the computation by executing only a fixed number of experts per input. In this section, we study the scaling behaviour of sparse mixture-of-experts (MoE) architectures employing Top-K routing. We first show that under the Standard Parameterisation, Top-K MoEs inherit the instability identified for soft routing. We then show that under the μP-MoE parameterisation, Top-K routing can induce symmetry breaking across experts without the separate symmetry breaking mechanism of the router bias. In particular, for Top-K MoEs we identify μP-MoE as the unique parameterisation (within an equivalence class) under which the resulting training dynamics satisfy all four asymptotic desiderata: stability, maximal feature learning, faithfulness, and the necessary condition for expert specialisation.

5.1 Router collapse under SP scaling

We first note that the unstable dynamics of MoEs under soft routing described in Section 4.1 arise from the scaling behaviour of the router update norms, and are therefore not specific to any particular gating function. While Top-K gating introduces sparsity, it does not rectify the scaling misalignment of SP. Although the gradient is sparse (non-zero only for the top K experts), the *magnitude* of

the non-zero entries still scales as $\Theta(n^{1/2})$ due to strong correlations in the propagating updates. Consequently, Proposition 4.1 continues to hold for Top-K MoEs under SP.

5.2 μP-MoE scaling

In the soft routing case, we showed that μP-MoE parameterisation ensures stability and maximal feature learning but fails to permit expert specialisation. Here, a router bias term or other forms of symmetry breaking are necessary to allow for specialisation. In contrast, we show here that Top-K routing itself acts as a symmetry breaking mechanism. This is as the Top-K index set, even in the infinite-width limit, need not converge deterministically. Instead, it may remain random due to the influence of finite-width fluctuations.

To see this, recall that the router logits are scalars in the Tensor Program and converge to deterministic limits as the width n increases, with finite-width fluctuations vanishing asymptotically (e.g. see Section 4.2). Let $\delta_n = (\mathbf{h}^2(n))_{(K)} - (\mathbf{h}^2(n))_{(K+1)}$ denote the gap between the K-th and $(K+1)$-th largest router logits. If this gap vanishes as $n \to \infty$, then the Top-K index set becomes sensitive to the relative ordering induced by finite-width fluctuations, and therefore fails to concentrate. This phenomenon is formalised in the following proposition.

Proposition 5.1 (Top-K indices remain random under μP-MoE)**.** *Consider the router logits under μP-MoE as a sequence of the model width n, $\mathbf{h}^2(n) = [\mathbf{h}_1^2, \dots, \mathbf{h}_n^2]$ with $\mathbf{h}_k^2 \in \mathbb{R}^m$. In the first forward pass $\mathbf{h}^2(n)$ converges almost surely to $C\mathbf{1}$, where $\mathbf{1}$ is a vector of ones $\in \mathbb{R}^m$. Further we have for finite n that $\mathbf{h}^2(n) = C + \sigma_n \epsilon_i(n)$ for $i \in [m]$, with $\sigma_n \to 0$ and with $\epsilon(n)$ converging to a Gaussian distribution. Then $\mathrm{TopK}(\mathbf{h}^2(n))$ weakly converges to $\mathrm{TopK}(\epsilon)$.*

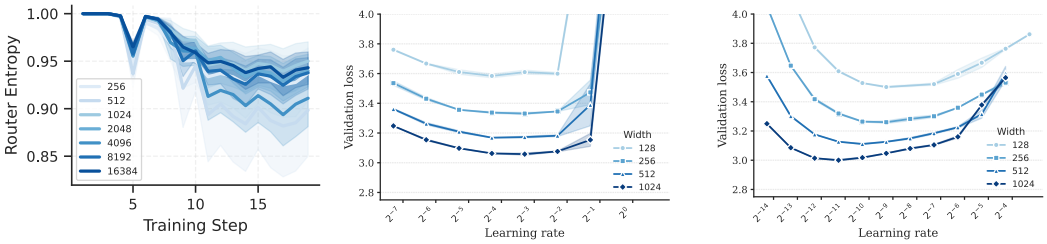
Proof of this result is given in Appendix C. The above result holds when the router logits become symmetric — Top-K acts on the finite-width fluctuations acting as a symmetry breaker. We already see this in the first forward pass, here Proposition 5.1 implies that under μP-MoE, although the router logits converge at initialisation to a deterministic and symmetric limit, the Top-K routing decisions remain stochastic in the infinite-width limit, acting on the Gaussian fluctuations [39]. Consequently, the selected expert indices are entirely determined by the limiting distribution of the rescaled finite-width fluctuations and thus remain stochastic. The resulting gradients for each expert is thus not symmetric leading to different expert distributions.

This highlights a practical benefit of Top-K routing beyond computational efficiency. In contrast to soft routing with μP-MoE, where symmetry of initialisation and gradient updates leads to deterministic and uniform router assignments (Proposition 4.2), the Top-K operator allows the initial randomness to persist in the infinite-width limit. As a result, Top-K routing can induce symmetry breaking across experts already at initialisation thereby preventing router collapse. We verify this in Figure 3a (a), where in contrast to the soft routing case, the router entropy decreases for softmax gating without the symmetry breaking of a router bias.

Hyperparameter Transfer: As hyperparameter transfer is an important empirical consequence of width stable dynamics afforded by our analysis [43, 4], we verify this for the μP-MoE parametrisation. We show that hyperparameters transfer for Top-K MoE with bias with a Mixtral style model [21] trained with Adam in Figure 3b with μP-MoE and lack of transfer in SP-MoE Figure 3c. We show further hyperparameter transfer results (with and without bias terms) in Appendix F.4. We additionally show coordinate checks on Mixtral MoE models in Appendix F showing that our scaling feature learns in transformer MoE models.

6 Discussion

We analyse MoE training in the regime $n \to \infty$ with a fixed number of experts m, and find that the infinite-width limit reveals degeneracies. Under the commonly used Standard Parameterisation (SP), as the width grows the router quickly saturates softmax/sigmoid gates, collapsing the router Jacobian, and causing router gradients to vanish which may, at least in part, explain training instabilities observed in practice [14, 5]. Under μP-MoE scalings, stability (and feature learning for Sigmoid routing) is recovered, but the router dynamics become synchronous across experts in soft MoEs, keeping experts identically distributed so that specialisation disappears.



(a) (b) μP-MoE (c) SP-MoE

Figure 3: (a) Router entropy of the Softmax–Top-K router without bias. The Top-K operation induces symmetry breaking, which prevents router stagnation. (b) (c) Learning rate transfer. Hyperparameters are tuned on the smallest model.

In the Top-K routing case we identify that the Top-K operator on its own allows (without additional interventions) for symmetry breaking and can allow for specialisation with μP-MoE scaling, as the selected expert set can remain stochastic even when logits converge to a symmetric limit. Despite this, it has been observed that Top-K MoEs with Softmax routing often exhibit representation collapse, in which the experts collapse to the same representation [5, 28]. In this case, our results in Proposition 4.3 suggest that softmax routing can stop learning, whereas sigmoid routing can still update features. Overall, large-width MoEs require not only stable scaling, but also explicit, controlled symmetry breaking to avoid router collapse (SP) or failure of expert specialisation (μP soft routing).

Acknowledgments and Disclosure of Funding

This work has been supported by the Gatsby Charitable Foundation (GAT4058). Volkan Cevher acknowledges the financial support of the Swiss National Science Foundation, SNSF grant No 2000-1-240094.

References

[1] Patrick Billingsley. *Convergence of probability measures*. John Wiley & Sons, 2013.

[2] Blake Bordelon and Cengiz Pehlevan. Self-consistent dynamical field theory of kernel evolution in wide neural networks. *Advances in Neural Information Processing Systems*, 35, 2022.

[3] Blake Bordelon, Hamza Chaudhry, and Cengiz Pehlevan. Infinite limits of multi-head transformer dynamics. *Advances in Neural Information Processing Systems*, 37, 2024.

[4] Blake Bordelon, Lorenzo Noci, Mufan Bill Li, Boris Hanin, and Cengiz Pehlevan. Depthwise hyperparameter transfer in residual networks: Dynamics and scaling limit. In *The Twelfth International Conference on Learning Representations*, 2024.

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

[6] Lenaic Chizat, Edouard Oyallon, and Francis Bach. On lazy training in differentiable programming. *Advances in neural information processing systems*, 32, 2019.

[7] Róbert Csordás, Kazuki Irie, and Jürgen Schmidhuber. Approximating two-layer feedforward networks for efficient transformers. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, 2023.

[8] Damai Dai, Li Dong, Shuming Ma, Bo Zheng, Zhifang Sui, Baobao Chang, and Furu Wei. Stablemoe: Stable routing strategy for mixture of experts. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2022.

[9] Damai Dai, Chengqi Deng, Chenggang Zhao, Rx Xu, Huazuo Gao, Deli Chen, Jiashi Li, Wangding Zeng, Xingkai Yu, Y Wu, et al. Deepseekmoe: Towards ultimate expert specialization in mixture-of-experts language models. In *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2024.

[10] Nolan Dey, Bin Claire Zhang, Lorenzo Noci, Mufan Li, Blake Bordelon, Shane Bergsma, Cengiz Pehlevan, Boris Hanin, and Joel Hestness. Don't be lazy: Completep enables compute-efficient deep transformers. *arXiv preprint arXiv:2505.01618*, 2025.

[11] Giang Do, Hung Le, and Truyen Tran. Simsmoe: Toward efficient training mixture of experts via solving representational collapse. In *Findings of the Association for Computational Linguistics: NAACL 2025*, 2025.

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

[13] Zhiwen Fan, Rishov Sarkar, Ziyu Jiang, Tianlong Chen, Kai Zou, Yu Cheng, Cong Hao, Zhangyang Wang, et al. M^{3}vit: Mixture-of-experts vision transformer for efficient multi-task learning with model-accelerator co-design. *Advances in Neural Information Processing Systems*, 35, 2022.

[14] 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), 2022.

[15] Moritz Haas, Jin Xu, Volkan Cevher, and Leena Chennuru Vankadara. $\boldsymbol{\mu}\mathbf{P^2}$: Effective sharpness aware minimization requires layerwise perturbation scaling. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[16] Moritz Haas, Sebastian Bordt, Ulrike von Luxburg, and Leena Chennuru Vankadara. On the surprising effectiveness of large learning rates under standard width scaling. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[17] Xing Han, Huy Nguyen, Carl Harris, Nhat Ho, and Suchi Saria. Fusemoe: Mixture-of-experts transformers for fleximodal fusion. *Advances in Neural Information Processing Systems*, 37, 2024.

[18] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, 2015.

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

[20] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. *Advances in neural information processing systems*, 31, 2018.

[21] 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.

[22] Tianze Jiang, Blake Bordelon, Cengiz Pehlevan, and Boris Hanin. Hyperparameter transfer with mixture-of-expert layers. *arXiv preprint arXiv:2601.20205*, 2026.

[23] Jaehoon Lee, Lechao Xiao, Samuel Schoenholz, Yasaman Bahri, Roman Novak, Jascha Sohl-Dickstein, and Jeffrey Pennington. Wide neural networks of any depth evolve as linear models under gradient descent. *Advances in neural information processing systems*, 32, 2019.

[24] Jan Małaśnicki, Kamil Ciebiera, Mateusz Boruń, Maciej Pióro, Jan Ludziejewski, Maciej Stefaniak, Michał Krutul, Sebastian Jaszczur, Marek Cygan, Kamil Adamczewski, et al. μ-parametrization for mixture of experts. *arXiv preprint arXiv:2508.09752*, 2025.

[25] Song Mei, Andrea Montanari, and Phan-Minh Nguyen. A mean field view of the landscape of two-layer neural networks. *Proceedings of the National Academy of Sciences*, 115(33), 2018.

[26] Radford M Neal. Bayesian learning for neural networks, 1996.

[27] Huy Nguyen, Nhat Ho, and Alessandro Rinaldo. Sigmoid gating is more sample efficient than softmax gating in mixture of experts. *Advances in Neural Information Processing Systems*, 37, 2024.

[28] Quang Pham, Giang Do, Huy Nguyen, TrungTin Nguyen, Chenghao Liu, Mina Sartipi, Binh T Nguyen, Savitha Ramasamy, Xiaoli Li, Steven Hoi, et al. Competesmoe–effective training of sparse mixture of experts via competition. *arXiv preprint arXiv:2402.02526*, 2024.

[29] Joan Puigcerver, Carlos Riquelme Ruiz, Basil Mustafa, and Neil Houlsby. From sparse to soft mixtures of experts. In *The Twelfth International Conference on Learning Representations*, 2024.

[30] Grant M Rotskoff and Eric Vanden-Eijnden. Neural networks as interacting particle systems: Asymptotic convexity of the loss landscape and universal scaling of the approximation error. *stat*, 1050, 2018.

[31] Mana Sakai, Ryo Karakida, and Masaaki Imaizumi. Infinite-width limit of a single attention layer: Analysis via tensor programs. In *The Thirty-ninth Annual Conference on Neural Information Processing Systems*, 2025.

[32] 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. In *International Conference on Learning Representations*, 2017.

[33] Gemini Team, Petko Georgiev, Ving Ian Lei, Ryan Burnell, Libin Bai, Anmol Gulati, Garrett Tanzer, Damien Vincent, Zhufeng Pan, Shibo Wang, et al. Gemini 1.5: Unlocking multimodal understanding across millions of tokens of context. *arXiv preprint arXiv:2403.05530*, 2024.

[34] Jamba Team, Barak Lenz, Alan Arazi, Amir Bergman, Avshalom Manevich, Barak Peleg, Ben Aviram, Chen Almagor, Clara Fridman, Dan Padnos, et al. Jamba-1.5: Hybrid transformer-mamba models at scale. *arXiv preprint arXiv:2408.12570*, 2024.

[35] Leena Chennuru Vankadara, Jin Xu, Moritz Haas, and Volkan Cevher. On feature learning in structured state space models. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[36] Lean Wang, Huazuo Gao, Chenggang Zhao, Xu Sun, and Damai Dai. Auxiliary-loss-free load balancing strategy for mixture-of-experts. *arXiv preprint arXiv:2408.15664*, 2024.

[37] Ziteng Wang, Jun Zhu, and Jianfei Chen. Remoe: Fully differentiable mixture-of-experts with reLU routing. In *The Thirteenth International Conference on Learning Representations*, 2025.

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

[39] Greg Yang. Wide feedforward or recurrent neural networks of any architecture are gaussian processes. *Advances in Neural Information Processing Systems*, 32, 2019.

[40] Greg Yang. Tensor programs iii: Neural matrix laws. *arXiv preprint arXiv:2009.10685*, 2020.

[41] Greg Yang and Edward J Hu. Tensor programs iv: Feature learning in infinite-width neural networks. In *International Conference on Machine Learning*. PMLR, 2021.

[42] Greg Yang and Etai Littwin. Tensor programs ivb: Adaptive optimization in the infinite-width limit. *arXiv preprint arXiv:2308.01814*, 2023.

[43] Greg Yang, Edward J Hu, Igor Babuschkin, Szymon Sidor, Xiaodong Liu, David Farhi, Nick Ryder, Jakub Pachocki, Weizhu Chen, and Jianfeng Gao. Tensor programs v: tuning large neural networks via zero-shot hyperparameter transfer. In *Proceedings of the 35th International Conference on Neural Information Processing Systems*, 2021.

[44] Greg Yang, James B Simon, and Jeremy Bernstein. A spectral condition for feature learning. *arXiv preprint arXiv:2310.17813*, 2023.

[45] Greg Yang, Dingli Yu, Chen Zhu, and Soufiane Hayou. Tensor programs VI: Feature learning in infinite depth neural networks. In *The Twelfth International Conference on Learning Representations*, 2024.

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

Appendix Roadmap: The Logic of Router Dynamics

This appendix provides the rigorous derivations for the infinite-width limits of Mixture-of-Experts (MoE) architectures. To navigate the theoretical contributions, we structure the analysis into a narrative arc: establishing the Tensor Program machinery, demonstrating the inevitable failure of standard soft routing, and deriving the conditional limit for Top-K routing.

- **Tensor Program for Soft MoE Architectures (Appendix A)**
  Develops the tensor-program representation of the forward/backward passes and training dynamics for soft-routed MoEs.
  - **Tensor Program for Soft MoE with SGD (Appendix A.1)**
    Specifies the NETSOR$\top^+$ program for SGD training, including all primitives and scaling conventions.
  - **Corresponding infinite-width limit for Soft MoE with SGD (Appendix A.2)**
    Derives the deterministic and distribution limit objects (Master Theorem recursion) induced by the SGD tensor program.
  - **Tensor Program for Soft MoE with Adam (Appendix A.3)**
    Extends the program to Adam using NE$\otimes$OR$\top$, introducing the OuterNonlin structure needed for adaptive updates.
  - **Corresponding infinite-width limit for Soft MoE with Adam (Appendix A.4)**
    Computes the Adam infinite-width dynamics, including the additional limit rule for OuterNonlin and faithfulness constraints.
- **Deriving Results for Soft MoE (Appendix B)**
  Uses the above limits to prove the main qualitative claims about soft routing: collapse under SP and lack of expert specalisation under μP-style stability.
  - **Router Collapse with Standard Parameterisation (Appendix B.1)**
    Shows that under SP scaling the router logits diverge, causing gating saturation and vanishing router gradients.
  - **Derivation of μP-MoE and Lack of Expert Specialiaation (Appendix B.2)**
    Proves that under stable μP-style scaling the router updates become identical across experts, preventing specialisation in the limit.
- **Top-K Mask Depends on Finite-Width Noise (Appendix C)**
  Establishes that when logits collapse to a common value (in the first forward pass under μP-MoE), the Top-K selection remains random in the limit because it is determined by vanishing-but-ordering noise.
- **Tensor Program for Top-K MoE (Appendix D)**
  Develops the tensor-program representation of the forward/backward passes and training dynamics for Top-K routed MoEs.
  - **Tensor Program for Top-K MoE (Appendix D.1)**
    Defines the Top-K training program.
  - **Corresponding infinite-width limit for Top-K MoE (Appendix D.2)**
    Derives the resulting infinite-width limit.
- **Additional Experiments (Appendix F)**
  Collects supplementary empirical results that validate the theoretical predictions across parametrisations.

A Tensor Program for Soft MoE

Here, we derive the Tensor program and the corresponding infinite width limit for soft MoEs (Definition 2.1) for both softmax and sigmoid gating. We start by deriving the TP for SGD (Appendix A.1) and the infinite width limit (Appendix A.2). Then, we derive the TP for Adam (Appendix A.3) and the infinite width limit (Appendix A.4).

A.1 Tensor Program for Soft MoE with SGD

Matrices, Vectors and Scalars for the program: As we consider the limit of $n \to \infty$, we refer to quantities that have both dimensions scaled ($\in \mathbb{R}^{n \times n}$) as 'matrices' in the program. Similarly, if only one of the dimensions is taken to infinity, with the other being held constant (e.g. $\in \mathbb{R}^{\bullet \times n}$ for some constant dimension $\bullet$), then we refer to the quantity as a 'vector' in the program. Finally, if both dimensions of a quantity are held constant, we refer to it as a 'scalar' in the program. Objects like $\mathbf{W}^2 \in \mathbb{R}^{m \times n}$, are treated as multiple (m) vector objects.

Notation: For all vectors in the program $\mathbf{h}$, we write $\delta \mathbf{h}_t := \theta_{\delta h_t}^{-1}(\mathbf{h}_t - \mathbf{h}_{t-1})$ as the normalised change in the features, where $\theta_{\delta h_t}$ is the scale of $(\mathbf{h}_t - \mathbf{h}_{t-1})$. Thus, $\delta \mathbf{h}_t$ scales $\Theta(1)$. We denote scalars and vectors at training step t as $\mathbf{h}_t = \mathbf{h}_0 + (\mathbf{h}_1 - \mathbf{h}_0) + \ldots + (\mathbf{h}_t - \mathbf{h}_{t-1})$. We define $\boldsymbol{\chi}_t := \frac{\partial \mathcal{L}_t}{\partial f_t} \in \mathbb{R}^{d_{\text{out}} \times 1}$ which we assume is always $\Theta(1)$. This holds for common loss functions (such as mean squared error and cross entropy) as long as the function is stable (does not blow up with width). For a gradient term $\frac{\partial \mathbf{f}}{\partial \mathbf{h}}$ with scale $\theta_{\partial h}$, we define the normalised gradient $\mathrm{d}\mathbf{h} := \theta_{\partial h}^{-1} \frac{\partial \mathbf{f}}{\partial \mathbf{h}}$, such that $\mathrm{d}\mathbf{h}$ scales $\Theta(1)$. We also denote $\theta'_x := n\theta_x$. We reserve subscript for indexing training time $\mathbf{h}_t$, and index a component with brackets, for example $(\mathbf{h}_t)_\alpha$ denotes the α component of $\mathbf{h}$ at step t.

Tensor Program Operations: For the SGD limit, we use NETSOR$\top^+$ [40, 41]. The program contains a sequence of $\mathbb{R}^n$ vectors and $\mathbb{R}$ scalars generated from one of the following ways from initial vectors and scalars

- `MatMul`: Given $\mathbf{W} \in \mathbb{R}^{n \times n}$ and $\mathbf{v} \in \mathbb{R}^n$, new vectors are generated as $\mathbf{W}\mathbf{v}$ or $\mathbf{W}^T\mathbf{v}$.
- `Nonlin`: Given $\phi : \mathbb{R}^k \times \mathbb{R}^l \to \mathbb{R}$, scalars $\theta_1, \ldots, \theta_l \in \mathbb{R}$ and vectors $\mathbf{v}^1, \ldots, \mathbf{v}^k \in \mathbb{R}^n$, a new vector can be generated as

  $$\phi(\mathbf{v}^1, \ldots, \mathbf{v}^k; \theta_1, \ldots, \theta_l) \in \mathbb{R}^n$$

  where ϕ applies coordinate wise on $\mathbf{v}^1_\alpha$.
- `Moment`: Given the same as above, a new scalar can be generated

  $$\frac{1}{n} \sum_{\alpha=1}^{n} \phi(\mathbf{v}^1_\alpha, \ldots, \mathbf{v}^k_\alpha; \theta_1, \ldots, \theta_l) \in \mathbb{R}.$$

The only assumption on ϕ operations are that they are pseudo-Lipschitz [41, Assumption F.4].

We first make the assumption that the parametrisation values are chosen such that the network is stable throughout training — no quantity diverges with increasing width.

Setting A.1 (Stable parametrisation)**.** We only consider values of $\{b_\ell, c_\ell\}_\ell$ for each layer ℓ such that the network is stable throughout training.

This implies that no quantity diverges with increasing width, leading to well-defined limits.

We will write the forward pass, backward pass, and SGD updates as tensor programs [41]. This requires all initial matrices ($n \times n$ shape) to be samples from $\mathcal{N}\left(0, \frac{1}{n}\right)$, and all initial vectors from $\mathcal{N}(0, 1)$. All $\theta_\bullet$ parameters are defined as initial scalars.

Network: We define the network as following,

$$\mathbf{h}^1 = \mathbf{W}^1\boldsymbol{\xi}, \qquad \in \mathbb{R}^{n\times 1} \tag{5}$$
$$\mathbf{h}^2 = \mathbf{W}^2\mathbf{h}^1 + \mathbf{b}^2, \qquad \in \mathbb{R}^{m\times 1} \tag{6}$$
$$\tilde{\mathbf{h}}^2 = \mathrm{G}(\mathbf{h}^2), \qquad \in [0,1]^{m\times 1} \tag{7}$$
$$\mathbf{h}^{3,k} = \mathbf{W}^{3,k}\mathbf{h}^1, \qquad \in \mathbb{R}^{n\times 1} \tag{8}$$
$$\mathbf{h}^3 = \sum_{k=1}^{m} \mathbf{h}^{3,k}\tilde{h}^2_k, \qquad \in \mathbb{R}^{n\times 1} \tag{9}$$
$$\mathbf{f} = \mathbf{W}^4\mathbf{h}^3, \qquad \in \mathbb{R}^{d_{\text{out}}\times 1} \tag{10}$$

where G is the gating function (either softmax or sigmoid), we refer to $\mathbf{h}^2$ as the router, each $\mathbf{h}^{3,k}$ as expert k, and $\mathbf{f}$ as the function output. We follow a simpler variation of the *abc* parametrisation [41]. The initial scalars, vectors and matrices in the program are defined below,

$$\mathbf{W}^1 = n^{-b_1}\overline{\mathbf{W}^1} = \theta_{W_0^1}\overline{\mathbf{W}^1}, \qquad \mathbf{W}^1 \in \mathbb{R}^{n\times d_{\text{in}}} \qquad \overline{W^1_{ij}} \sim \mathcal{N}(0,1), \tag{11}$$
$$\mathbf{W}^2 = n^{-b_2}\overline{\mathbf{W}^2} = \theta_{W_0^2}\overline{\mathbf{W}^2}, \qquad \mathbf{W}^2 \in \mathbb{R}^{m\times n} \qquad \overline{W^2_{ij}} \sim \mathcal{N}(0,1), \tag{12}$$
$$\mathbf{W}^{3,k} = n^{\frac{1}{2}-b_3}\overline{\mathbf{W}^{3,k}} = \theta_{W_0^3}\overline{\mathbf{W}^3_k}, \qquad \mathbf{W}^{3,k} \in \mathbb{R}^{n\times n} \qquad \overline{W^{3,k}}_{ij} \sim \mathcal{N}(0,n^{-1}), \tag{13}$$
$$\mathbf{W}^4 = n^{-b_4}\overline{\mathbf{W}^4} = \theta_{W_0^4}\overline{\mathbf{W}^4}, \qquad \mathbf{W}^4 \in \mathbb{R}^{d_{\text{out}}\times n} \qquad \overline{W^4_{ij}} \sim \mathcal{N}(0,1). \tag{14}$$

Note that we only consider the limit $n \to \infty$ with other factors held fixed, and thus $\{\mathbf{W}^{3,k}\}_{k=1}^m$ are the only initial matrices in the program ($n \times n$).

Initial Matrices: $\overline{\mathbf{W}^{3,k}}$.

Initial Vectors: $\overline{\mathbf{W}^1}, \overline{\mathbf{W}^2}, \overline{\mathbf{W}^4}$, where $\overline{\mathbf{W}^1}$ is a set of d_{in} initial vectors, $\overline{\mathbf{W}^2}$ is a set of m initial vector and $\overline{\mathbf{W}^4}$ is a set of d_{out} initial vectors.

Initial Scalars: All $\theta_\bullet$ parameters. The router bias $\mathbf{b}^2$ is a set of m initial scalars sampled from some density such that $\mathbb{P}(b_i = b_j) = 0$. In practice, this is satisfied by sampling from$\mathcal{N}(0, \sigma_b^2)$ where σ_b^2 is some hyperparameter.

Noise and Learnable Bias: We consider two separate cases here:

1. Case 1: A different noise is added at each time step $\{\mathbf{b}^2_t\}_{t=1}^T$ [32].

2. Case 2: The term $\mathbf{b}^2_0$ is learned, either through gradients of the loss or load balancing considerations [36].

   Note that the updates below are written only for learnable biases from gradient of the loss. For the noise case, a new noise is sampled every forward pass.

Conditioned on router bias/noise: The standard Tensor Programs Master Theorem assumes initial scalars converge almost surely to deterministic limits. A router bias/noise vector sampled from a continuous distribution does not satisfy this assumption unconditionally. Instead, we use a conditional formulation: we first sample the finite collection of router bias/noise scalars and condition on their realised values. Conditional on these values, they are treated as fixed scalar constants in the tensor program. Thus, for any fixed finite time horizon T, we condition on $\{\mathbf{b}^2_t\}_{t=0}^T$ in the noise case, or on the initialisation of $\mathbf{b}^2_0$ in the learnable-bias case. The resulting infinite-width limits are deterministic conditional on the realised bias/noise values, but random unconditionally through their dependence on those values. Practically, this corresponds to coupling widths by using the same sampled router bias/noise values across widths.

First Forward Pass:

$$\mathbf{h}_0^1 = \theta_{W_0^1}\overline{\mathbf{W}_0^1}\boldsymbol{\xi}, \qquad (\mathbf{h}_0^1)_\alpha = \phi\left((\overline{\mathbf{W}_0^1})_{\alpha 1}, \ldots, (\overline{\mathbf{W}_0^1})_{\alpha d_{\text{in}}}; \xi_1, \ldots, \xi_{d_{\text{in}}}, \theta_{W_0^1}\right) = \sum_{j=1}^{d_{\text{in}}} \theta_{W_0^1}(\overline{\mathbf{W}_0^1})_{\alpha j}\xi_j, \qquad \texttt{(Nonlin)}$$

$$\mathbf{h}_0^2 = \theta'_{W_0^2}\frac{1}{n}\overline{\mathbf{W}_0^2}\mathbf{h}_0^1 + \mathbf{b}_0^2, \qquad (\mathbf{h}_0^2)_k = \frac{1}{n}\sum_{\alpha=1}^{n}\phi\left((\overline{\mathbf{W}_0^2})_{k\alpha}, (\mathbf{h}_0^1)_\alpha; \theta'_{W_0^2}, \mathbf{b}_0^2\right) = \frac{1}{n}\theta'_{W_0^2}\sum_{\alpha=1}^{n}(\overline{\mathbf{W}_0^2})_{k\alpha}(\mathbf{h}_0^1)_\alpha + \mathrm{b}_0^2, \quad \texttt{(Moment)}$$

$$\tilde{\mathbf{h}}_0^2 = \mathrm{G}(\mathbf{h}_0^2), \qquad (\texttt{Moment},\ \text{See E})$$

$$\hat{\mathbf{h}}_0^{3,k} = \overline{\mathbf{W}^{3,k}}\mathbf{h}_0^1, \qquad \texttt{(MatMul)}$$

$$\mathbf{h}_0^{3,k} = \theta_{W_0^3}\hat{\mathbf{h}}_0^{3,k}, \qquad (\mathbf{h}_0^{3,k})_\alpha = \phi\left((\hat{\mathbf{h}}_0^{3,k})_\alpha; \theta_{W_0^3}\right) = \theta_{W_0^3}(\hat{\mathbf{h}}_0^{3,k})_\alpha \qquad \texttt{(Nonlin)}$$

$$\mathbf{h}_0^3 = \sum_{k=1}^{m}\mathbf{h}_0^{3,k}\tilde{h}_{0,k}^2, \qquad (\mathbf{h}_0^3)_\alpha = \phi\left((\mathbf{h}_0^{3,1})_\alpha, \ldots, (\mathbf{h}_0^{3,m})_\alpha; \tilde{h}_{0,1}^2, \ldots, \tilde{h}_{0,m}^2\right) = \sum_{k=1}^{m}(\mathbf{h}_0^{3,k})_\alpha\tilde{h}_{0,k}^2 \qquad \texttt{(Nonlin)}$$

$$\mathbf{f}_0 = \theta'_{W_0^4}\frac{1}{n}\overline{\mathbf{W}_0^4}\mathbf{h}_0^3, \qquad (\mathbf{f}_0)_\beta = \frac{1}{n}\sum_{\alpha=1}^{n}\phi\left((\overline{\mathbf{W}_0^4})_{\beta\alpha}, (\mathbf{h}_0^3)_\alpha; \theta'_{W_0^4}\right) = \frac{1}{n}\theta'_{W_0^4}\sum_{\alpha=1}^{n}(\overline{\mathbf{W}_0^4})_{\beta\alpha}(\mathbf{h}_0^3)_\alpha. \qquad \texttt{(Moment)}$$

Stability at Initialisation: Satisfying stability at initialisation (Definition 2.6) fixes the values of b_1, b_2, b_3, b_4, and the corresponding θ parameters. We find the values that satisfy stability at initialisation and fix them for the rest of the program. $\mathbf{h}_0^1$ is trivially a Gaussian with scale $\theta_{W_0^1} = n^{-b_1}\|\boldsymbol{\xi}\|$. Hence, we set $b_1 = 0$ to satisfy stability. Using the Master theorem [41], $\mathbb{E}\left[(Z^{h_0^2})^2\right] = n^{(1-2b_2)}\mathbb{E}\left[(Z^{h_0^1})^2\right] = n^{(1-2b_2)}\|\boldsymbol{\xi}\|^2$. Stability is satisfied for $b_2 \geq \frac{1}{2}$. We can use the Master theorem to find the distribution of $\mathbf{h}_0^3$ as $Z^{h^{3_0}} = \mathcal{N}(0, n^{(1-2b_3)}\mathbb{E}\left[(Z^{h_0^1})^2\right])$. We thus set $b_3 = \frac{1}{2}$. Finally, using similar reasoning as for $\mathbf{h}_0^2$, setting $b_4 \geq \frac{1}{2}$ ensures that $\mathbf{f}_0$ is stable. Note that $\tilde{\mathbf{h}}_0^2$ is always stable due to the bounded softmax operation.

Remark A.2 (Feature Learning considerations to constrain output-like initialisation). Stability constrains the output-like layers to initialise with $b_2 \geq 1/2$ and $b_4 \geq 1/2$. For ease of exposition, we use the Tensor Program derived below to constrain these values. Looking at the first feature updates of the router

$$\mathbf{h}_1^2 - \mathbf{h}_0^2 = \theta_{\delta h_1^1}\theta'_{W_0^2}\frac{\overline{\mathbf{W}_0^2}\delta\mathbf{h}_1^1}{n} - \theta'_{\delta W_1^2}\eta\mathrm{d}\mathbf{h}_0^2\boldsymbol{\chi}_0\frac{(\mathbf{h}_0^1)^T\mathbf{h}_1^1}{n}.$$

By the Master Theorem, $\frac{\overline{\mathbf{W}_0^2}\delta\mathbf{h}_1^1}{n}$ which is a Moment operation converges to an expectation (Appendix A.2). If we have feature learning for $\mathbf{h}_1^1 - \mathbf{h}_0^1$, we require that $\theta_{\delta h_1^1} = 1$. Thus, for the first term to scale $\Theta(1)$, we require $\theta'_{W_0^2} = n \cdot n^{-b_2} = 1$, which implies setting $b_2 = 1$. A similar arguement holds for setting $b_4 = 1$.

The final values for stability at initialisation (and feature learning for 'output-like' layers) are

$$b_1 = 0, \quad b_2 = 1, \quad b_3 = \frac{1}{2}, \quad b_4 = 1, \implies \theta_{W_0^1} = 1, \quad \theta_{W_0^2} = n^{-1}, \quad \theta_{W_0^3} = 1, \quad \theta_{W_0^4} = n^{-1}.$$

Setting A.3 (Stability at initialisation)**.** For stability at initialisation, and feature learning considerations for 'output-like' layers (Remark A.2), we choose

$$b_1 = 0, \quad b_2 = 1, \quad b_3 = \frac{1}{2}, \quad b_4 = 1,$$

which gives

$$\theta_{W_0^1} = 1, \quad \theta_{W_0^2} = n^{-1}, \quad \theta_{W_0^3} = 1, \quad \theta_{W_0^4} = n^{-1}.$$

We set these values in the program for ease of exposition.

Note that under the assumptions made in Setting A.1 and Setting A.3 certain vectors have scale $\Theta(1)$ throughout training.

Remark A.4. Under the assumption of stable parametrisations and the values selected for stability at initialisation, the vectors $\mathbf{h}_t^1, \{\mathbf{h}_t^{3,k}\}_{k=1}^m, \mathbf{h}_t^3$ have scale $\Theta(1)$ throughout training. As the softmax and sigmoid operations are bounded, the softmax output $\tilde{\mathbf{h}}_t^2$ has scale $\Theta(1)$ throughout training.

First Backward Pass:

$$\mathrm{d}\mathbf{h}_0^3 = (\overline{\mathbf{W}_0^4})^T, \quad \in \mathbb{R}^{n \times d_{\text{out}}}. \tag{15}$$

$$n \cdot \mathrm{d}\tilde{h}_{0,k}^2 = (\mathbf{h}_0^{3,k})^T \mathrm{d}\mathbf{h}_0^3, \quad \in \mathbb{R}^{1 \times d_{\text{out}}}, \qquad \forall k \in [m] \tag{16}$$

$$(\mathrm{d}\tilde{h}_{0,k}^2)_\beta = \frac{1}{n} \sum_{\alpha=1}^{n} \phi\left((\mathbf{h}_0^{3,k})_\alpha, (\mathrm{d}\mathbf{h}_0^3)_{\alpha\beta}; -\right), \qquad \texttt{(Moment)} \tag{17}$$

$$\phi(\ldots) := (\mathbf{h}_0^{3,k})_\alpha (\mathrm{d}\mathbf{h}_0^3)_{\alpha\beta}, \tag{18}$$

where we repeat the operation for each expert to give $\mathrm{d}\tilde{h}_{0,1}^2, \ldots, \mathrm{d}\tilde{h}_{0,m}^2$.

$$\mathrm{d}h_{0,k}^2 = \sum_{j=1}^{m} J(\tilde{\mathbf{h}}_0^2)_{jk} \mathrm{d}\tilde{h}_{0,j}^2, \quad \in \mathbb{R}^{1 \times d_{\text{out}}}, \qquad \forall k \in [m] \tag{19}$$

$$(\mathrm{d}h_{0,k}^2)_\beta = \frac{1}{n} \sum_{\alpha=1}^{n} \phi\left(-; \{J(\tilde{\mathbf{h}}_0^2)_{jk}\}_{j=1}^m, \{(\mathrm{d}\tilde{h}_{0,j}^2)_\beta\}_{j=1}^m\right), \qquad \texttt{(Moment)} \tag{20}$$

$$\phi(\ldots) := \sum_{j=1}^{m} J(\tilde{\mathbf{h}}_0^2)_{jk} (\overline{\mathrm{d}\tilde{h}_{0,j}^2})_\beta, \tag{21}$$

where $J(\tilde{\mathbf{h}}_0^2)_{ij} = \frac{\partial(\tilde{\mathbf{h}}_0^2)_i}{\partial(\mathbf{h}_0^2)_j}$.

Again, we repeat this operation for each expert to give $\mathrm{d}h_{0,1}^2, \ldots, \mathrm{d}h_{0,m}^2$. We also define

$$\mathrm{d}\mathbf{h}_0^2 = \begin{pmatrix} \mathrm{d}h_{0,1}^2 \\ \vdots \\ \mathrm{d}h_{0,m}^2 \end{pmatrix} \in \mathbb{R}^{m \times d_{\text{out}}}. \tag{22}$$

$$\mathrm{d}\mathbf{h}_0^{3,k} = \tilde{h}_{0,k}^2 \mathrm{d}\mathbf{h}_0^3, \quad \in \mathbb{R}^{n \times d_{\text{out}}}, \qquad \forall k \in [m] \tag{23}$$

$$(\mathrm{d}\mathbf{h}_0^{3,k})_{\alpha\beta} = \phi\left((\mathrm{d}\mathbf{h}_0^3)_{\alpha\beta}; \tilde{h}_{0,k}^2\right), \qquad \texttt{(Nonlin)} \tag{24}$$

$$\phi(\ldots) := \tilde{h}_{0,k}^2 \overline{\mathrm{d}\mathbf{h}_0^3}, \tag{25}$$

We repeat this for each expert to give $\mathrm{d}\mathbf{h}_0^{3,1}, \ldots, \mathrm{d}\mathbf{h}_0^{3,m}$.

$$\mathrm{d}\mathbf{h}_0^1 = \sum_{k=1}^{m} (\overline{\mathbf{W}_0^2})_{:,k}^T \mathrm{d}h_{0,k}^2 + \sum_{k=1}^{m} (\mathbf{W}_0^{3,k})^T \mathrm{d}\mathbf{h}_0^{3,k}, \quad \in \mathbb{R}^{n \times d_{\text{out}}}, \tag{26}$$

$$(\mathrm{d}\mathbf{h}_0^1)_{\alpha\beta} = \phi\left(\{(\overline{\mathbf{W}_0^2})_{\alpha k}^T, (\mathbf{V}_0^k)_{\alpha\beta}\}_{k=1}^m, ; \{(\mathrm{d}h_{0,k}^2)_\beta\}_{k=1}^m\right), \qquad \texttt{(Nonlin)} \tag{27}$$

$$(\mathbf{V}_0^k)_{:,\beta} := (\mathbf{W}_0^{3,k})^T (\mathrm{d}\mathbf{h}_0^{3,k})_{:,\beta}, \qquad \texttt{(MatMul)} \tag{28}$$

$$\phi(\ldots) := \sum_{k=1}^{m} (\overline{\mathbf{W}_0^2})_{:,k}^T \mathrm{d}h_{0,k}^2 + \sum_{k=1}^{m} \mathbf{V}_0^k. \tag{29}$$

Note that the final gradient has two branches. The scale of this term thus will be the maximum of the the scale of the twp branches. There is a $\theta'_{W_0^4} \theta_{W_0^2}$ scale coming from the router and a $\theta_{W_0^4}$ coming from the experts. The scale from the branches is the same here.

First Weight Updates: Recall that we define the gradient of the loss with respect to the function as $\boldsymbol{\chi}_t$. For the vectors $\mathbf{W} \in \{\mathbf{W}^1, \mathbf{W}^2, \mathbf{W}^4\}$ we write $\delta\mathbf{W}_t := \theta_{\delta W_t}^{-1}(\mathbf{W}_t - \mathbf{W}_{t-1})$, and thus the

weights at step t as $\mathbf{W}_t = \theta_{W_0}\overline{\mathbf{W}_0} + \ldots + \theta_{\delta W_t}\delta W_t$.

$$\mathbf{W}_1^4 - \mathbf{W}_0^4 = -n^{-c_4}\eta\boldsymbol{\chi}_0(\mathbf{h}_0^3)^T \tag{30}$$

$$\theta_{\delta W_1^4} := n^{-c_4}. \tag{31}$$

$$\mathbf{W}_1^{3,k} - \mathbf{W}_0^{3,k} = -n^{-c_3}\theta_{W_0^4}\eta\mathrm{d}\mathbf{h}_0^{3,k}\boldsymbol{\chi}_0(\mathbf{h}_0^1)^T \tag{32}$$

$$\theta_{\delta W_1^3} := n^{-(c_3+1)}. \tag{33}$$

$$\mathbf{W}_1^2 - \mathbf{W}_0^2 = -n^{-c_2}n\theta_{W_0^4}\eta\mathrm{d}\mathbf{h}_0^2\boldsymbol{\chi}_0(\mathbf{h}_0^1)^T \tag{34}$$

$$\theta_{\delta W_1^2} := n^{-c_2}. \tag{35}$$

$$\mathbf{W}_1^1 - \mathbf{W}_0^1 = -n^{-c_1}n\theta_{W_0^4}\theta_{W_0^2}\eta\mathrm{d}\mathbf{h}_0^1\boldsymbol{\chi}_0(\boldsymbol{\xi})^T \tag{36}$$

$$\theta_{\delta W_1^1} := n^{-(c_1+1)}. \tag{37}$$

The router bias in the learnable case updates as

$$\mathbf{b}_1^2 - \mathbf{b}_0^2 = -\eta\theta'_{W_0^4}\mathrm{d}\mathbf{h}_0^2\boldsymbol{\chi}_0, \tag{38}$$

where due to stability at initialisation (Setting A.3) $\theta'_{W_0^4} = 1$.

First Feature Updates: The feature updates are affected by the change in the previous layers features (*propagating updates*) and the update in the weights at the current layer (*effective updates*). We write the feature updates for the vectors $\mathbf{h} \in \{\mathbf{h}^1, \{\mathbf{h}^{3,k}\}_{k=1}^m, \mathbf{h}^3\}$ as $\delta\mathbf{h}_t := \theta_{\delta h_t}^{-1}(\mathbf{h}_t - \mathbf{h}_{t-1})$. This gives each features for the vectors at step t as $\mathbf{h}_t = \mathbf{h}_0 + \theta_{\delta h_1}\delta\mathbf{h}_1 + \ldots + \theta_{\delta h_t}\delta\mathbf{h}_t$. For the scalars $\mathbf{h} \in \{\mathbf{h}^2, \tilde{\mathbf{h}}^2, \mathbf{f}\}$ we simply write $\delta\mathbf{h}_t := (\mathbf{h}_t - \mathbf{h}_{t-1})$ and $\mathbf{h}_t = \mathbf{h}_0 + \ldots + \delta\mathbf{h}_t$.

$$\mathbf{h}_1^1 - \mathbf{h}_0^1 = (\mathbf{W}_1^1 - \mathbf{W}_0^1)\boldsymbol{\xi} \tag{39}$$

$$= -\theta_{\delta W_1^1}\eta\mathrm{d}\mathbf{h}_0^1\boldsymbol{\chi}_0\boldsymbol{\xi}^T\boldsymbol{\xi} \tag{40}$$

$$(\delta\mathbf{h}_1^1)_\alpha = \phi\left(\{(\mathrm{d}\mathbf{h}_0^1)_{\alpha\beta}\}_{\beta=1}^{d_{\text{out}}}; -\eta, \{(\boldsymbol{\chi}_0\boldsymbol{\xi}^T\boldsymbol{\xi})_\beta\}_{\beta=1}^{d_{\text{out}}}\right) \qquad \texttt{(Nonlin)} \tag{41}$$

$$\phi(\ldots) := -\eta\sum_{\beta=1}^{d_{\text{out}}}(\mathrm{d}\mathbf{h}_0^1)_{\alpha\beta}(\boldsymbol{\chi}_0\boldsymbol{\xi}^T\boldsymbol{\xi})_\beta, \tag{42}$$

$$\theta_{\delta h_1^1} := \theta_{\delta W_1^1}, \tag{43}$$

$$\mathbf{h}_1^2 - \mathbf{h}_0^2 = \mathbf{W}_0^2(\mathbf{h}_1^1 - \mathbf{h}_0^1) + (\mathbf{W}_1^2 - \mathbf{W}_0^2)\mathbf{h}_1^1 + (\mathbf{b}_1^2 - \mathbf{b}_0^2) \tag{44}$$

$$= \theta_{\delta h_1^1}\theta'_{W_0^2}\frac{\overline{\mathbf{W}_0^2}\delta\mathbf{h}_1^1}{n} - \theta'_{\delta W_1^2}\eta\mathrm{d}\mathbf{h}_0^2\boldsymbol{\chi}_0\frac{(\mathbf{h}_0^1)^T\mathbf{h}_1^1}{n} - \eta\mathrm{d}\mathbf{h}_0^2\boldsymbol{\chi}_0, \tag{45}$$

$$\theta_{\delta h_1^2}(\delta\mathbf{h}_1^2)_k = \frac{1}{n}\sum_{\alpha=1}^n\phi\left(\overline{\mathbf{W}_0^2}_{k\alpha}, (\delta\mathbf{h}_1^1)_\alpha, (\mathbf{h}_0^1)_\alpha, (\mathbf{h}_1^1)_\alpha; \theta'_{W_0^4}, \theta_{\delta h_1^1}, \theta'_{W_0^2}, \theta'_{\delta W_1^2}, \eta, \{(\mathrm{d}\mathbf{h}_0^2)_{k\beta}, (\boldsymbol{\chi}_0)_\beta\}_\beta^{d_{\text{out}}}\right) \qquad \texttt{(Moment)} \tag{46}$$

$$\phi(\ldots) := \theta_{\delta h_1^1}\theta'_{W_0^2}\overline{\mathbf{W}_0^2}_{k\alpha}(\delta\mathbf{h}_1^1)_\alpha - \theta'_{\delta W_1^2}\eta\sum_{\beta=1}^{d_{\text{out}}}(\mathrm{d}\mathbf{h}_0^2)_{k\beta}(\boldsymbol{\chi}_0)_\beta(\mathbf{h}_0^1)_\alpha(\mathbf{h}_1^1)_\alpha - \eta\theta'_{W_0^4}\sum_{\beta=1}^{d_{\text{out}}}\mathrm{d}(\mathbf{h}_0^2)_{k\beta}(\boldsymbol{\chi}_0)_\beta, \tag{47}$$

$$\theta_{\delta h_1^2} := \max(\theta_{\delta h_1^1}\theta'_{W_0^2}, \theta'_{\delta W_1^2}), \tag{48}$$

$$\tilde{\mathbf{h}}_1^2 - \tilde{\mathbf{h}}_0^2 = \mathrm{G}(\mathbf{h}_1^2) - \mathrm{G}(\mathbf{h}_0^2) \tag{49}$$

$$= \mathrm{G}\left(\mathbf{h}_0^2 + \theta_{\delta h_1^2}\delta\mathbf{h}_1^2\right) - \mathrm{G}(\mathbf{h}_0^2) \tag{50}$$

$$\theta_{\delta\tilde{h}_1^2}(\delta\tilde{\mathbf{h}}_1^2)_k = \frac{1}{n}\sum_{\alpha=1}^n\phi_k\left(-; \{(\mathbf{h}_0^2)_k, (\delta\mathbf{h}_1^2)_k\}_{k=1}^m, \theta_{\delta h_1^2}\right), \qquad \texttt{(Moment)}, \tag{51}$$

$$\theta_{\delta\tilde{h}_1^2} := \theta_{\delta h_1^2}. \tag{52}$$

where G is the router gating function (either softmax or sigmoid). Both softmax and sigmoid operations can be represented as a Moment (see Appendix E).

$$\mathbf{h}_1^{3,k} - \mathbf{h}_0^{3,k} = \mathbf{W}_0^{3,k}(\mathbf{h}_1^1 - \mathbf{h}_0^1) + (\mathbf{W}_1^{3,k} - \mathbf{W}_0^{3,k})\mathbf{h}_1^1, \tag{53}$$

$$= \theta_{\delta h_1^1}\mathbf{W}_0^{3,k}\delta\mathbf{h}_1^1 - \theta'_{\delta W_1^3}\eta \mathrm{d}\mathbf{h}_0^{3,k}\boldsymbol{\chi}_0\frac{(\mathbf{h}_0^1)^T(\mathbf{h}_1^1)}{n}, \tag{54}$$

$$(\delta\mathbf{h}_1^{3,k})_\alpha = \phi\left(\Upsilon_\alpha, \{(\mathrm{d}\mathbf{h}_0^{3,k})_{\alpha l}\}_{l=1}^{d_{\text{out}}}; \theta_{\delta h_1^1/\delta h_1^{3,k}}, \theta'_{\delta W_1^3/\delta h_1^{3,k}}, \{(\boldsymbol{\chi}_0)_l\}_{l=1}^{d_{\text{out}}}, \eta, c_0,\right), \quad \texttt{(Nonlin)} \tag{55}$$

$$\Upsilon := \mathbf{W}_0^{3,k}\delta\mathbf{h}_1^1, \quad \texttt{(MatMul)} \tag{56}$$

$$c_0 := \frac{1}{n}\sum_{\alpha=1}^{n}\psi\left((\mathbf{h}_0^1)_\alpha, (\mathbf{h}_1^1)_\alpha; -\right), \quad \texttt{(Moment)} \tag{57}$$

$$\phi(\dots) := \theta_{\delta h_1^1/\delta h_1^{3,k}}\Upsilon_\alpha - \theta'_{\delta W_1^3/\delta h_1^{3,k}}\eta c_0\sum_{l=1}^{d_{\text{out}}}(\mathrm{d}\mathbf{h}_0^{3,k})_{\alpha l}(\boldsymbol{\chi}_0)_l, \tag{58}$$

$$\theta_{\delta h_1^{3,k}} := \max(\theta_{\delta h_1^1}, \theta'_{\delta W_1^3}), \tag{59}$$

$$\theta_{\delta h_1^1/\delta h_1^{3,k}} := \theta_{\delta h_1^1}/\theta_{\delta h_1^{3,k}}, \tag{60}$$

$$\theta'_{\delta W_1^3/\delta h_1^{3,k}} := \theta'_{\delta W_1^3}/\theta_{\delta h_1^{3,k}} \tag{61}$$

We repeat the above process for each expert k to give $\{\delta\mathbf{h}_1^{3,k}\}_{k=1}^m$.

$$\mathbf{h}_1^3 - \mathbf{h}_0^3 = \sum_{k=1}^{m}\left(\tilde{h}_{0,k}^2(\mathbf{h}_1^{3,k} - \mathbf{h}_0^{3,k}) + (\tilde{h}_{1,k}^2 - \tilde{h}_{0,k}^2)\mathbf{h}_1^{3,k}\right) \tag{62}$$

$$= \sum_{k=1}^{m}\left(\theta_{\delta h_1^{3,k}}\tilde{h}_{0,k}^2\delta\mathbf{h}_1^{3,k} + \theta_{\delta\tilde{h}_1^2}\delta\tilde{h}_{1,k}^2\mathbf{h}_1^{3,k}\right), \tag{63}$$

$$(\delta\mathbf{h}_1^3)_\alpha = \phi\left(\{(\delta\mathbf{h}_1^{3,k})_\alpha\}_{k=1}^m, \{(\mathbf{h}_1^{3,k})_\alpha\}_{k=1}^m; \theta_{\delta h_1^{3,k}/\delta h_1^3}, \theta_{\delta\tilde{h}_1^2/\delta h_1^3}, \{\tilde{h}_{0,k}^2\}_{k=1}^m, \{\delta\tilde{h}_{1,k}^2\}_{k=1}^m\right), \quad \texttt{(Nonlin)} \tag{64}$$

$$\phi(\dots) := \sum_{k=1}^{m}\left(\theta_{\delta h_1^{3,k}/\delta h_1^3}\tilde{h}_{0,k}^2(\delta\mathbf{h}_1^{3,k})_\alpha + \theta_{\delta\tilde{h}_1^2/\delta h_1^3}\delta\tilde{h}_{1,k}^2(\mathbf{h}_1^{3,k})_\alpha\right), \tag{65}$$

$$\theta_{\delta h_1^3} := \max(\theta_{\delta h_1^{3,k}}, \theta_{\delta\tilde{h}_1^2}), \tag{66}$$

$$\theta_{\delta h_1^{3,k}/\delta h_1^3} := \theta_{\delta h_1^{3,k}}/\theta_{\delta h_1^3}, \tag{67}$$

$$\theta_{\delta\tilde{h}_1^2/\delta h_1^3} := \theta_{\delta\tilde{h}_1^2}/\theta_{\delta h_1^3}. \tag{68}$$

$$\mathbf{f}_1 - \mathbf{f}_0 = \mathbf{W}_0^4(\mathbf{h}_1^3 - \mathbf{h}_0^3) + (\mathbf{W}_1^4 - \mathbf{W}_0^4)\mathbf{h}_1^3 \tag{69}$$

$$= \theta_{\delta h_1^3}\frac{\overline{\mathbf{W}_0^4}\delta\mathbf{h}_1^3}{n} - \theta'_{\delta W_1^4}\eta\boldsymbol{\chi}_0\frac{(\mathbf{h}_0^3)^T(\mathbf{h}_1^3)}{n}, \tag{70}$$

$$\theta_{\delta f_1}(\delta\mathbf{f}_1)_\beta = \frac{1}{n}\sum_{\alpha=1}^{n}\phi\left(\overline{\mathbf{W}_0^4}_{\beta\alpha}, (\delta\mathbf{h}_1^3)_\alpha, (\mathbf{h}_0^3)_\alpha, (\mathbf{h}_1^3)_\alpha; \theta_{\delta h_1^3}, \theta'_{\delta W_1^4}, \eta, (\boldsymbol{\chi}_0)_\beta\right), \quad \texttt{(Moment)} \tag{71}$$

$$\phi(\dots) := \theta_{\delta h_1^3}(\overline{\mathbf{W}_0^4})_{\beta\alpha}(\delta\mathbf{h}_1^3)_\alpha - \theta'_{\delta W_1^4}\eta(\boldsymbol{\chi}_0)_\beta(\mathbf{h}_0^3)_\alpha(\mathbf{h}_1^3)_\alpha, \tag{72}$$

$$\theta_{\delta f_1} := \max(\theta_{\delta h_1^3}, \theta'_{\delta W_1^4}). \tag{73}$$

***t*th Backward Pass:** Recall that we write the weights at step t as $\mathbf{W}_t = \theta_{W_0}\overline{\mathbf{W}_0} + \ldots + \theta_{\delta W_t}\delta W_t$.

$$\mathrm{d}\mathbf{h}_t^3 := (\overline{\mathbf{W}_t^4})^T, \quad \in \mathbb{R}^{n\times d_{\text{out}}}. \tag{74}$$

$$n \cdot \mathrm{d}\tilde{h}^2_{t,k} = (\mathbf{h}^{3,k}_t)^T \mathrm{d}\mathbf{h}^3_t, \quad \in \mathbb{R}^{1\times d_{\text{out}}}, \qquad \forall k \in [m] \tag{75}$$

$$(\mathrm{d}\tilde{h}^2_{t,k})_\beta = \frac{1}{n}\sum_{\alpha=1}^{n} \phi\left((\mathbf{h}^{3,k}_t)_\alpha, (\mathrm{d}\mathbf{h}^3_t)_{\alpha\beta}; -\right), \qquad \texttt{(Moment)} \tag{76}$$

$$\phi(\ldots) := (\mathbf{h}^{3,k}_t)_\alpha (\mathrm{d}\mathbf{h}^3_t)_{\alpha\beta}, \tag{77}$$

where we repeat this operation for each expert $\mathrm{d}\tilde{h}^2_{t,1}, \ldots, \mathrm{d}\tilde{h}^2_{t,m}$.

$$\mathrm{d}h^2_{t,k} = \sum_{j=1}^{m} J(\tilde{\mathbf{h}}^2_t)_{jk} \mathrm{d}\tilde{h}^2_{t,j}, \quad \in \mathbb{R}^{1\times d_{\text{out}}}, \qquad \forall k \in [m] \tag{78}$$

$$(\mathrm{d}h^2_{t,k})_\beta = \frac{1}{n}\sum_{\alpha=1}^{n} \phi\left(-; \{J(\tilde{\mathbf{h}}^2_t)_{jk}\}_{j=1}^m, \{(\mathrm{d}\tilde{h}^2_{t,j})_\beta\}_{j=1}^m\right), \qquad \texttt{(Moment)} \tag{79}$$

$$\phi(\ldots) := \sum_{j=1}^{m} J(\tilde{\mathbf{h}}^2_t)_{jk} (\mathrm{d}\tilde{h}^2_{t,j})_\beta, \tag{80}$$

where $J(\tilde{\mathbf{h}}^2_t)_{ij} = \frac{\partial(\tilde{\mathbf{h}}^2_t)_i}{\partial(\mathbf{h}^2_t)_j}$. It is useful to consider this terms for all the experts,

$$(\mathrm{d}\mathbf{h}^2_t)_\beta = \begin{pmatrix} (\mathrm{d}h^2_{t,1})_\beta \\ \vdots \\ (\mathrm{d}h^2_{t,m})_\beta \end{pmatrix} \quad \in \mathbb{R}^{m\times d_{\text{out}}}. \tag{81}$$

$$\mathrm{d}\mathbf{h}^{3,k}_t = \tilde{h}^2_{t,k} \mathrm{d}\mathbf{h}^3_t, \quad \in \mathbb{R}^{n\times d_{\text{out}}}, \qquad \forall k \in [m] \tag{82}$$

$$(\mathrm{d}\mathbf{h}^{3,k}_t)_{\alpha\beta} = \phi\left((\mathrm{d}\mathbf{h}^3_t)_{\alpha\beta}; \tilde{h}^2_{t,k}\right), \qquad \texttt{(Nonlin)} \tag{83}$$

$$\phi(\ldots) := \tilde{h}^2_{t,k} (\overline{\mathrm{d}\mathbf{h}^3_t})_{\alpha\beta}, \tag{84}$$

where we repeat this for each expert to give $\mathrm{d}\mathbf{h}^{3,1}_t, \ldots, \mathrm{d}\mathbf{h}^{3,m}_t$.

$$\mathrm{d}\mathbf{h}^1_t = \sum_{k=1}^{m} (\overline{\mathbf{W}^2_t})^T_{:,k} \mathrm{d}h^2_{t,k} + \sum_{k=1}^{m} (\mathbf{W}^{3,k}_0)^T \mathrm{d}\mathbf{h}^{3,k}_t - \sum_{k=1}^{m}\sum_{s=0}^{t-1} \eta \mathbf{h}^1_s (\boldsymbol{\chi}_s)^T \frac{(\mathrm{d}\mathbf{h}^{3,k}_s)^T \mathrm{d}\mathbf{h}^{3,k}_t}{n}, \quad \in \mathbb{R}^{n\times d_{\text{out}}}, \tag{85}$$

$$(\mathrm{d}\mathbf{h}^1_t)_{\alpha\beta} = \phi\left(\{(\overline{\mathbf{W}^2_t})^T_{\alpha k}, (\mathbf{V}^k_t)_{\alpha\beta}\}_{k=1}^m, \{(\mathbf{h}^1_s)_\alpha\}_{s=0}^{t-1}; \{(\mathrm{d}h^2_{t,k})_\beta\}_{k=1}^m, \{(\boldsymbol{\chi}_s)_\beta, \{C^k_{s;\gamma\beta}\}_{k=1}^m\}_{s=0}^{t-1}\right), \qquad \texttt{(Nonlin)} \tag{86}$$

$$(\mathbf{V}^k_t)_{:,\beta} := (\mathbf{W}^{3,k}_0)^T (\mathrm{d}\mathbf{h}^{3,k}_t)_{:,\beta}, \qquad \texttt{(MatMul)} \tag{87}$$

$$C^k_{s;\gamma\beta} := \frac{1}{n}\sum_{\alpha=1}^{n} \psi\left((\mathrm{d}\mathbf{h}^{3,k}_s)_{\alpha\gamma}, (\mathrm{d}\mathbf{h}^{3,k}_t)_{\alpha\beta}; -\right), \qquad \texttt{(Moment)} \tag{88}$$

$$\phi(\ldots) := \sum_{k=1}^{m} (\overline{\mathbf{W}^2_t})^T_{\alpha k} (\mathrm{d}h^2_{t,k})_\beta + \sum_{k=1}^{m} (\mathbf{V}^k_t)_{\alpha\beta} - \sum_{k=1}^{m}\sum_{s=0}^{t-1}\sum_{\gamma=1}^{d_{\text{out}}} \eta (\mathbf{h}^1_s)_\alpha (\boldsymbol{\chi}_s)_\gamma C^k_{s;\gamma\beta}. \tag{89}$$

tth Weight Updates: Recall that we define the gradient of the loss with respect to the function as $\boldsymbol{\chi}_t \in \mathbb{R}^{d_{\text{out}} \times 1}$.

$$\mathbf{W}_t^4 - \mathbf{W}_{t-1}^4 = -n^{-c_4}\eta \boldsymbol{\chi}_{t-1}(\mathbf{h}_{t-1}^3)^T \tag{90}$$

$$\theta_{\delta W_t^4} := n^{-c_4}. \tag{91}$$

$$\mathbf{W}_t^{3,k} - \mathbf{W}_{t-1}^{3,k} = -n^{-c_3}\theta_{W_{t-1}^4}\eta \mathrm{d}\mathbf{h}_{t-1}^{3,k}\boldsymbol{\chi}_{t-1}(\mathbf{h}_{t-1}^1)^T \tag{92}$$

$$\theta_{\delta W_t^3} := n^{-c_3}\theta_{W_{t-1}^4} \tag{93}$$

$$\mathbf{W}_t^2 - \mathbf{W}_{t-1}^2 = -n^{-c_2}\theta'_{W_{t-1}^4}\eta \mathrm{d}\mathbf{h}_{t-1}^2\boldsymbol{\chi}_{t-1}(\mathbf{h}_{t-1}^1)^T \tag{94}$$

$$\theta_{\delta W_t^2} := n^{-c_2}\theta'_{W_{t-1}^4}. \tag{95}$$

$$\mathbf{W}_t^1 - \mathbf{W}_{t-1}^1 = -n^{-c_1}\theta'_{W_{t-1}^2}\theta_{W_{t-1}^4}\eta \mathrm{d}\mathbf{h}_{t-1}^1\boldsymbol{\chi}_{t-1}(\boldsymbol{\xi})^T \tag{96}$$

$$\theta_{\delta W_t^1} := n^{-c_1}\theta'_{W_{t-1}^2}\theta_{W_{t-1}^4}. \tag{97}$$

For the learnable bias we have,

$$\mathbf{b}_t^2 - \mathbf{b}_{t-1}^2 = -\eta\theta'_{W_{t-1}^4}\mathrm{d}\mathbf{h}_{t-1}^2\boldsymbol{\chi}_{t-1}, \tag{98}$$

tth Feature Updates:

$$\mathbf{h}_t^1 - \mathbf{h}_{t-1}^1 = (\mathbf{W}_t^1 - \mathbf{W}_{t-1}^1)\boldsymbol{\xi} \tag{99}$$

$$= -\theta_{\delta W_t^1}\eta \mathrm{d}\mathbf{h}_{t-1}^1\boldsymbol{\chi}_{t-1}\boldsymbol{\xi}^T\boldsymbol{\xi} \tag{100}$$

$$(\delta\mathbf{h}_t^1)_\alpha = \phi\left(\{(\mathrm{d}\mathbf{h}_{t-1}^1)_{\alpha j}\}_{j=1}^{d_{\text{out}}}; -\eta, \{(\boldsymbol{\chi}_{t-1}\boldsymbol{\xi}^T\boldsymbol{\xi})_j\}_{j=1}^{d_{\text{out}}}\right) \qquad \texttt{(Nonlin)} \tag{101}$$

$$\phi(\ldots) := -\eta\sum_{j=1}^{d_{\text{out}}}(\mathrm{d}\mathbf{h}_{t-1}^1)_{\alpha j}(\boldsymbol{\chi}_{t-1}\boldsymbol{\xi}^T\boldsymbol{\xi})_j \tag{102}$$

$$\theta_{\delta h_t^1} := \theta_{\delta W_t^1}. \tag{103}$$

$$\mathbf{h}_t^2 - \mathbf{h}_{t-1}^2 = \mathbf{W}_{t-1}^2(\mathbf{h}_t^1 - \mathbf{h}_{t-1}^1) + (\mathbf{W}_t^2 - \mathbf{W}_{t-1}^2)\mathbf{h}_t^1 + (\mathbf{b}_t^2 - \mathbf{b}_{t-1}^2) \tag{104}$$

$$= \theta'_{W_{t-1}^2}\theta_{\delta h_t^1}\frac{\overline{\mathbf{W}_{t-1}^2}\delta\mathbf{h}_t^1}{n} - \theta'_{\delta W_t^2}\eta \mathrm{d}\mathbf{h}_{t-1}^2\boldsymbol{\chi}_{t-1}\frac{(\mathbf{h}_{t-1}^1)^T\mathbf{h}_t^1}{n} - \eta\theta'_{W_{t-1}^4}\mathrm{d}\mathbf{h}_{t-1}^2\boldsymbol{\chi}_{t-1}, \tag{105}$$

$$\theta_{\delta h_t^2}(\delta\mathbf{h}_t^2)_k = \frac{1}{n}\sum_{\alpha=1}^{n}\phi\Bigg((\overline{\mathbf{W}_{t-1}^2})_{k\alpha}, (\delta\mathbf{h}_t^1)_\alpha, (\mathbf{h}_{t-1}^1)_\alpha, (\mathbf{h}_t^1)_\alpha;$$

$$\theta'_{W_{t-1}^4}, \theta'_{W_{t-1}^2}, \theta_{\delta h_t^1}, \theta'_{\delta W_t^2}, \eta, \{(\mathrm{d}h_{t-1,k}^2)_\beta, (\boldsymbol{\chi}_{t-1})_\beta\}_{\beta=1}^{d_{\text{out}}}\Bigg), \qquad \texttt{(Moment)} \tag{106}$$

$$\phi(\ldots) := \theta'_{W_{t-1}^2}\theta_{\delta h_t^1}(\overline{\mathbf{W}_{t-1}^2})_{k\alpha}(\delta\mathbf{h}_t^1)_\alpha - \theta'_{\delta W_t^2}\eta\sum_{\beta=1}^{d_{\text{out}}}(\mathrm{d}h_{t-1,k}^2)_\beta(\boldsymbol{\chi}_{t-1})_\beta(\mathbf{h}_{t-1}^1)_\alpha(\mathbf{h}_t^1)_\alpha$$

$$- \eta\theta'_{W_{t-1}^4}\sum_{\beta=1}^{d_{\text{out}}}(\mathrm{d}\mathbf{h}_{t-1}^2)_{k\beta}(\boldsymbol{\chi}_{t-1})_\beta. \tag{107}$$

$$\theta_{\delta h_t^2} := \max(\theta'_{W_{t-1}^2}\theta_{\delta h_t^1}, \theta'_{\delta W_t^2}, \theta'_{W_{t-1}^4}). \tag{108}$$

$$\tilde{\mathbf{h}}_t^2 - \tilde{\mathbf{h}}_{t-1}^2 = \mathrm{G}(\mathbf{h}_t^2) - \mathrm{G}(\mathbf{h}_{t-1}^2) \tag{109}$$

$$= \mathrm{G}\left(\mathbf{h}_{t-1}^2 + \theta_{\delta h_t^2}\delta\mathbf{h}_t^2\right) - \mathrm{G}(\mathbf{h}_{t-1}^2) \tag{110}$$

$$\theta_{\delta\tilde{h}_t^2}(\delta\tilde{\mathbf{h}}_t^2)_k = \frac{1}{n}\sum_{\alpha=1}^{n}\phi_k\left(-; \{(\mathbf{h}_{t-1}^2)_k, (\delta\mathbf{h}_t^2)_k\}_{k=1}^m, \theta_{\delta h_t^2}\right), \qquad \texttt{(Moment)}, \tag{111}$$

$$\theta_{\delta\tilde{h}_t^2} := \theta_{\delta h_t^2}, \tag{112}$$

where G is the gating function (either softmax or sigmoid) (see Appendix E) [41].

$$\mathbf{h}_t^{3,k} - \mathbf{h}_{t-1}^{3,k} = \mathbf{W}_{t-1}^{3,k}(\mathbf{h}_t^1 - \mathbf{h}_{t-1}^1) + (\mathbf{W}_t^{3,k} - \mathbf{W}_{t-1}^{3,k})\mathbf{h}_t^1, \tag{113}$$

$$= \theta_{\delta h_t^1}\mathbf{W}_0^{3,k}\delta\mathbf{h}_t^1 - \sum_{s=0}^{t-2}\theta'_{\delta W_{s+1}^3}\theta_{\delta h_t^1}\eta \mathrm{d}\mathbf{h}_s^{3,k}\boldsymbol{\chi}_s\frac{(\mathbf{h}_s^1)^T(\delta\mathbf{h}_t^1)}{n} - \theta'_{\delta W_t^3}\eta\mathrm{d}\mathbf{h}_{t-1}^{3,k}\boldsymbol{\chi}_{t-1}\frac{(\mathbf{h}_{t-1}^1)^T(\mathbf{h}_t^1)}{n}, \tag{114}$$

$$(\delta\mathbf{h}_t^{3,k})_\alpha = \phi\left(\Upsilon_\alpha, \{\Lambda_{s,\alpha}\}_{s=0}^{t-2}, \mathbf{V}_\alpha; \theta_{\delta h_t^1/\delta h_t^{3,k}}, \{\theta'_{\delta W_{s+1}^3\delta h_t^1/\delta h_t^{3,k}}\}_{s=0}^{t-2}, \theta'_{\delta W_t^3/\delta h_t^{3,k}}\right) \tag{Nonlin}$$

$$\tag{115}$$

$$\phi(\ldots) := \theta_{\delta h_t^1/\delta h_t^{3,k}}\Upsilon_\alpha - \sum_{s=0}^{t-2}\theta'_{\delta W_{s+1}^3\delta h_t^1/\delta h_t^{3,k}}\Lambda_{s,\alpha} - \theta'_{\delta W_t^3/\delta h_t^{3,k}}\mathbf{V}_\alpha, \tag{116}$$

$$\Upsilon := \mathbf{W}_0^{3,k}\delta\mathbf{h}_t^1, \tag{MatMul}$$

$$\tag{117}$$

$$\Lambda_{s,\alpha} := \psi\left(\{(\mathrm{d}\mathbf{h}_s^{3,k})_{\alpha l}\}_{l=1}^{d_{\text{out}}}; \eta, \{(\boldsymbol{\chi}_s)_l\}_{l=1}^{d_{\text{out}}}, d_s\right) = \eta d_s\sum_{l=1}^{d_{\text{out}}}(\mathrm{d}\mathbf{h}_s^{3,k})_{\alpha l}(\boldsymbol{\chi}_s)_l, \tag{Nonlin}$$

$$\tag{118}$$

$$d_s := \frac{1}{n}\sum_{\alpha=1}^{n}\nu\left((\mathbf{h}_s^1)_\alpha, (\delta\mathbf{h}_t^1)_\alpha; -\right), \tag{Moment}$$

$$\tag{119}$$

$$\mathbf{V}_\alpha := \psi\left(\{(\mathrm{d}\mathbf{h}_{t-1}^{3,k})_{\alpha l}\}_{l=1}^{d_{\text{out}}}; \{(\boldsymbol{\chi}_{t-1})_l\}_{l=1}^{d_{\text{out}}}, \eta, c_{t-1}\right) = \eta c_{t-1}\sum_{l=1}^{d_{\text{out}}}(\mathrm{d}\mathbf{h}_{t-1}^{3,k})_{\alpha l}(\boldsymbol{\chi}_{t-1})_l, \tag{Nonlin}$$

$$\tag{120}$$

$$c_{t-1} := \frac{1}{n}\sum_{\alpha=1}^{n}\psi\left((\mathbf{h}_{t-1}^1)_\alpha, (\mathbf{h}_t^1)_\alpha; -\right), \tag{Moment}$$

$$\tag{121}$$

$$\theta_{\delta h_t^{3,k}} = \max\left(\theta_{\delta h_t^1}, \max_{0\le s\le t-2}\left(\theta'_{\delta W_{s+1}^3}\theta_{\delta h_t^1}\right), \theta'_{\delta W_t^3}\right), \tag{122}$$

$$\theta_{\delta h_t^1/\delta h_t^{3,k}} := \theta_{\delta h_t^1}/\theta_{\delta h_t^{3,k}}, \tag{123}$$

$$\theta'_{\delta W_s^3\delta h_t^1/\delta h_t^{3,k}} := \theta'_{\delta W_s^3}\theta_{\delta h_t^1}/\theta_{\delta h_t^{3,k}}, \tag{124}$$

$$\theta'_{\delta W_t^3/\delta h_t^{3,k}} := \theta'_{\delta W_t^3}/\theta_{\delta h_t^{3,k}}. \tag{125}$$

We repeat the above process for each expert k.

$$\mathbf{h}_t^3 - \mathbf{h}_{t-1}^3 = \sum_{k=1}^{m}\left(\tilde{h}_{t-1,k}^2(\mathbf{h}_t^{3,k} - \mathbf{h}_{t-1}^{3,k}) + (\tilde{h}_{t,k}^2 - \tilde{h}_{t-1,k}^2)\mathbf{h}_t^{3,k}\right), \tag{126}$$

$$= \sum_{k=1}^{m}\theta_{\delta h_t^{3,k}}\tilde{h}_{t-1,k}^2\delta\mathbf{h}_t^{3,k} + \theta_{\delta\tilde{h}_t^2}\delta\tilde{h}_{t,k}^2\mathbf{h}_t^{3,k}, \tag{127}$$

$$(\delta\mathbf{h}_t^3)_\alpha = \phi\left(\{(\delta\mathbf{h}_t^{3,k})_\alpha\}_{k=1}^m, \{(\mathbf{h}_t^{3,k})_\alpha\}_{k=1}^m; \theta_{\delta h_t^{3,k}/\delta h_t^3}, \theta_{\delta\tilde{h}_t^2/\delta h_t^3}, \{\tilde{h}_{t-1,k}^2\}_{k=1}^m, \{\delta\tilde{h}_{t,k}^2\}_{k=1}^m\right), \tag{Nonlin}$$

$$\tag{128}$$

$$\theta_{\delta h_t^3} := \max(\theta_{\delta h_t^{3,k}}, \theta_{\delta\tilde{h}_t^2}), \tag{129}$$

$$\theta_{\delta h_t^{3,k}/\delta h_t^3} := \theta_{\delta h_t^{3,k}}/\theta_{\delta h_t^3}, \tag{130}$$

$$\theta_{\delta\tilde{h}_t^2/\delta h_t^3} := \theta_{\delta\tilde{h}_t^2}/\theta_{\delta h_t^3}. \tag{131}$$

Note that in the above, all experts have the same scale by symmetry so we do not write a maximum over experts.

$$\mathbf{f}_t - \mathbf{f}_{t-1} = \mathbf{W}^4_{t-1}(\mathbf{h}^3_t - \mathbf{h}^3_{t-1}) + (\mathbf{W}^4_t - \mathbf{W}^4_{t-1})\mathbf{h}^3_t, \tag{132}$$

$$= \theta'_{W^4_{t-1}}\theta_{\delta h^3_t}\frac{\overline{\mathbf{W}^4_{t-1}}\delta\mathbf{h}^3_t}{n} - \theta'_{\delta W^4_t}\eta\boldsymbol{\chi}_{t-1}\frac{(\mathbf{h}^3_{t-1})^T(\mathbf{h}^3_t)}{n}, \tag{133}$$

$$\theta_{\delta f_t}(\delta\mathbf{f}_t)_\beta = \frac{1}{n}\sum_{\alpha=1}^{n}\phi\left((\overline{\mathbf{W}^4_{t-1}})_{\beta\alpha}, (\delta\mathbf{h}^3_t)_\alpha, (\mathbf{h}^3_{t-1})_\alpha, (\mathbf{h}^3_t)_\alpha; \theta'_{W^4_{t-1}}, \theta_{\delta h^3_t}, \theta'_{\delta W^4_t}, \eta, (\boldsymbol{\chi}_{t-1})_\beta\right) \quad \texttt{(Moment)} \tag{134}$$

$$\phi(\ldots) := \theta'_{W^4_{t-1}}\theta_{\delta h^3_t}(\overline{\mathbf{W}^4_{t-1}})_{\beta\alpha}(\delta\mathbf{h}^3_t)_\alpha - \theta'_{\delta W^4_t}\eta(\boldsymbol{\chi}_{t-1})_\beta(\mathbf{h}^3_{t-1})_\alpha(\mathbf{h}^3_t)_\alpha \tag{135}$$

$$\theta_{\delta f_t} := \max(\theta'_{W^4_{t-1}}\theta_{\delta h^3_t}, \theta'_{\delta W^4_t}). \tag{136}$$

A.2 Infinite Width Limit for Soft MoE with SGD

Under stable parametrisations (Setting A.1), all the θ parameters in the program will converge to either zero or one.

Z random variables: According to the Master Theorem [41], the distribution of the coordinates of each vector z in the program will follow an iid distribution which we refer to as Z^z. Each scalar in the program $\bullet$ converges to a deterministic limit that we label $\mathring{\bullet}$. We recursively define each Z distribution based on the operation in the program used to create it.

Z Init: Let $\mathcal{V}_I$ be the set of initial vectors. For each $z \in \mathcal{V}_I$, we set $\hat{Z}^z := Z^z$ and $\dot{Z}^z := 0$ (defined below).

Z Nonlin: Given $\phi : \mathbb{R}^k \times \mathbb{R}^l \to \mathbb{R}$, scalars $\theta_1, \ldots, \theta_l \in \mathbb{R}$, and vectors $x^1, \ldots, x^k \in \mathbb{R}^n$, we have

$$Z^{\phi(x^1,\ldots,x^k;\theta_1,\ldots,\theta_l)} := \phi(Z^{z^1}, \ldots, Z^{x^k}; \mathring{\theta}_1, \ldots, \mathring{\theta}_l). \tag{137}$$

Z Moment: Given a scalar $\theta = \frac{1}{n}\sum_{\alpha=1}^{n}\phi(x^1_\alpha, \ldots, x^k_\alpha; \theta_1, \ldots, \theta_l)$, we have

$$\mathring{\theta} := \mathbb{E}[\phi(Z^{z^1}, \ldots, Z^{x^k}; \mathring{\theta}_1, \ldots, \mathring{\theta}_l)], \tag{138}$$

where the expectation is over $Z^{z^1}, \ldots, Z^{x^k}$.

Z MatMul: $Z^{Wx} := \hat{Z}^{Wx} + \dot{Z}^{Wx}$ for every matrix $W \sim \mathcal{N}(0, \sigma^2_W/n)$ and vector x.

- $\hat{Z}^{Wx}$ is Gaussian with zero mean. Let $\mathcal{V}_W := \{Wy \text{ for some vector } y\}$. Then $\{\hat{Z}^{Wy} : Wy \in \mathcal{V}_W\}$ is jointly Gaussian with zero mean and covariance,

$$\text{Cov}\left(\hat{Z}^{Wx}, \hat{Z}^{Wy}\right) := \sigma^2_W\mathbb{E}[Z^xZ^y] \text{ for any } Wx, Wy \in \mathcal{V}_W. \tag{139}$$

- Z^x has been computed by some operation that we unwind $Z^x = \phi(\{\hat{Z}^{W^Ty^i}\}_{i=1}^k), \{\hat{Z}^{z^i}\}_{i=1}^j; \{\mathring{\theta}_i\}_{i=1}^l)$ with $z^i \notin \mathcal{V}_{W^T}$[1]. Define $\partial Z^x/\partial\hat{Z}^{W^Ty^i} := \partial_i\phi(\ldots)$. Then

$$\dot{Z}^{Wx} := \sigma^2_W\sum_{i=1}^{k} Z^{y^i}\mathbb{E}\left[\frac{\partial Z^x}{\partial\hat{Z}^{W^Ty^i}}\right]. \tag{140}$$

Initial vectors: All our initial vectors have coordinates distributed according to $\mathcal{N}(0,1)$ by construction. The set of initial vectors are $\mathcal{V} := \{\overline{\mathbf{W}^1}, \overline{\mathbf{W}^2}, \overline{\mathbf{W}^4}\}$. Thus, $Z^W = \mathcal{N}(0,1)$ for $W \in \mathcal{V}$.

[1] z^i is not a function of W^T.

First Forward Pass:

$$Z^{h_0^1} = \sum_{j=1}^{d_{\text{in}}} \xi_j Z^{W_{0,j}^1}, \tag{141}$$

$$\mathring{\mathbf{h}}_0^2 = \mathbf{b}_0^2, \tag{142}$$

$$\mathring{\tilde{\mathbf{h}}}_0^2 = G(\mathbf{b}_0^2), \tag{143}$$

$$Z^{h_0^{3,k}} = Z^{W_0^{3,k} h_0^1} \text{ for every expert } k, \tag{144}$$

$$Z^{h_0^3} = \sum_{k=1}^{m} \mathring{\tilde{h}}_{0,k}^2 Z^{h_0^{3,k}} \tag{145}$$

$$\mathring{\mathbf{f}}_0 = \mathbf{0}. \tag{146}$$

Here, $Z^{W_0^1}$ is standard Gaussian by construction. $\mathring{\mathbf{h}}_0^2 = \mathbf{b}_0^2$ as the non bias router logits tend to zero $\mathbb{E}[Z^{W_0^2}]\mathbb{E}[Z^{h_0^1}] = 0$. $Z^{W_0^{3,k} h_0^1} = \hat{Z}^{W_0^{3,k} h_0^1}$ from the Z MatMul rule. Finally $\mathring{\mathbf{f}}_0 = \mathbf{0}$ as $\mathring{f}_0 = \mathbb{E}[Z^{W_0^4}]\mathbb{E}[Z^{h_0^3}] = 0$.

First Backward Pass: For $d_{\text{out}} > 1$, we index the distributions by β,

$$Z^{(\mathrm{d}h_0^3)_\beta} = Z^{(\overline{W_0^4})_\beta}, \tag{147}$$

$$\mathrm{d}\mathring{\tilde{h}}_{0,k}^2 = 0 \text{ for every expert } k, \tag{148}$$

$$\mathrm{d}\mathring{h}^2{}_{0,k} = 0 \text{ for every expert } k, \tag{149}$$

$$Z^{(\mathrm{d}h_0^{3,k})_\beta} = \mathring{\tilde{h}}_{0,k}^2 Z^{(\mathrm{d}h_0^3)_\beta} \text{ for every expert } k, \tag{150}$$

$$Z^{(\mathrm{d}h_0^1)_\beta} = \sum_{k=1}^{m} Z^{(W_0^{3,k})^T (\mathrm{d}h_0^{3,k})_\beta} \tag{151}$$

$$\tag{152}$$

$Z^{\overline{W_0^4}}$ is a standard Gaussian by construction. $\mathrm{d}\mathring{\tilde{h}}_{0,k}^2 = \mathbb{E}[Z^{h_0^{3,k}}]\mathbb{E}[\overline{W_0^4}] = 0$. From this, $\mathrm{d}\mathring{h}^2{}_{0,k} = 0$ follows. $Z^{(W_0^{3,k})^T \mathrm{d}h_0^3} = \hat{Z}^{(W_0^{3,k})^T \mathrm{d}h_0^3}$ from the Z MatMul rule.

First Feature Update:

$$Z^{\delta h_1^1} = -\eta \sum_{\beta=1}^{d_{\text{out}}} Z^{(\mathrm{d}h_0^1)_\beta} (\mathring{\boldsymbol{\chi}}_0)_\beta \boldsymbol{\xi}^T \boldsymbol{\xi}, \tag{153}$$

$$\delta \mathring{h}_{1,k}^2 = \mathring{\theta}_{\delta h_1^1} \mathbb{E}[Z^{(\overline{W_0^2})_{k,:}} Z^{\delta h_1^1}] - \mathring{\theta}'_{\delta W_1^2} \eta \sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathring{h}_{0,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_0)_\beta \mathbb{E}[Z^{h_0^1} Z^{h_1^1}] - \eta \mathrm{d}\mathring{\mathbf{h}}_0^2 \mathring{\boldsymbol{\chi}}_0, , \tag{154}$$

$$\delta \mathring{\tilde{\mathbf{h}}}_1^2 = \begin{cases} \mathrm{G}(\mathring{\mathbf{h}}_1^2) - \mathrm{G}(\mathring{\mathbf{h}}_0^2) \text{ if } \mathring{\theta}_{\delta h_1^2} = 1 \\ \mathrm{G}'(\mathring{\mathbf{h}}_0^2)(\delta \mathring{\mathbf{h}}_1^2) \text{ if } \mathring{\theta}_{\delta h_1^2} = 0 \end{cases}, \tag{155}$$

$$Z^{\delta h_1^{3,k}} = \mathring{\theta}_{\delta h_1^1 / \delta h_1^{3,k}} Z^{W_0^{3,k} \delta h_1^1} - \mathring{\theta}'_{\delta W_1^3 / \delta h_1^{3,k}} \eta \sum_{\beta=1}^{d_{\text{out}}} Z^{(\mathrm{d}h^{3,k})_\beta} (\mathring{\boldsymbol{\chi}}_0)_\beta \mathbb{E}[Z^{h_0^1} Z^{h_1^1}] \text{ for every expert } k, \tag{156}$$

$$Z^{\delta h_1^3} = \sum_{k=1}^{m} \mathring{\theta}_{\delta h_1^{3,k} / \delta h_1^3} \mathring{\tilde{h}}_{0,k}^2 Z^{\delta h_1^{3,k}} + \mathring{\theta}_{\delta \tilde{h}_1^2} \mathring{\theta}_{h_1^{3,k} / \delta h^3} \delta \mathring{\tilde{h}}_{1,k}^2 Z^{h_1^{3,k}}, \tag{157}$$

$$\delta \mathring{f}_1 = \mathring{\theta}_{\delta h_1^3} \mathbb{E}[Z^{\overline{W_0^4}} Z^{\delta h_1^3}] - \mathring{\theta}'_{\delta W_1^4} \eta \mathring{\chi}_0 \mathbb{E}[Z^{h_0^3} Z^{h_1^3}], \tag{158}$$

where G is either softmax or sigmoid and G′ is the derivative. Further, $Z^{W_0^{3,k}\delta h_1^1}$ is given by the Z MatMul rule,

$$Z^{W_0^{3,k}\delta h_1^1} = \hat{Z}^{W_0^{3,k}\delta h_1^1} + \dot{Z}^{W_0^{3,k}\delta h_1^1} \tag{159}$$

$$= \hat{Z}^{W_0^{3,k}\delta h_1^1} + \sum_{\beta=1}^{d_{\text{out}}} Z^{(\mathrm{d}h_0^{3,k})_\beta} \mathbb{E}\left[\frac{\partial Z^{\delta h_1^1}}{\partial \hat{Z}^{(W_0^{3,k})^T(\mathrm{d}h_0^{3,k})_\beta}}\right]. \tag{160}$$

We write the limit of each vector in the program as

$$Z^{h_t} = Z^{h_0} + \mathring{\theta}_{\delta h_1} Z^{\delta h_1} + \ldots + \mathring{\theta}_{\delta h_t} Z^{\delta h_t}. \tag{161}$$

Weight updates: The distribution of certain weight updates are required in the backward pass and thus the forward passes.

$$Z^{\overline{\delta W_t^4}} = -\eta \mathring{\chi}_{t-1} Z^{h_{t-1}^3}, \tag{162}$$

$$Z^{(\overline{\delta W_t^2})_{k,:}} = -\eta \sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathring{h}_{t-1,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_{t-1})_\beta Z^{h_{t-1}^1}, \tag{163}$$

$$\delta \mathring{\mathbf{b}}^2{}_t = -\eta \mathring{\theta}'_{W_{t-1}^4} \mathrm{d}\mathring{\mathbf{h}}_{t-1}^2 \mathring{\boldsymbol{\chi}}_{t-1}. \tag{164}$$

tth Backward Pass:

$$Z^{(\mathrm{d}h_t^3)_\beta} = Z^{(\overline{W_t^4})_\beta^T}, \tag{165}$$

$$(\mathrm{d}\mathring{\tilde{h}}_{t,k}^2)_\beta = \mathbb{E}[Z^{h_t^{3,k}} Z^{(\mathrm{d}h_t^3)_\beta}], \tag{166}$$

$$(\mathrm{d}\mathring{h}_{t,k}^2)_\beta = \sum_{j=1}^{m} J(\mathring{h}_t^2)_{jk} (\mathrm{d}\mathring{\tilde{h}}_{t,j}^2)_\beta, \tag{167}$$

$$Z^{(\mathrm{d}h_t^{3,k})_\beta} = \mathring{\tilde{h}}_{t,k}^2 Z^{(\mathrm{d}h_t^3)_\beta}, \tag{168}$$

$$Z^{(\mathrm{d}h_t^1)_\beta} = \sum_{k=1}^{m} \left(Z^{(\overline{W_t^2})_{k,:}} (\mathrm{d}\mathring{h}_{t,k}^2)_\beta + Z^{(W_0^{3,k})^T (\mathrm{d}h_t^{3,k})_\beta} - \eta \sum_{s=0}^{t-1} Z^{h_s^1} \sum_{\gamma=1}^{d_{\text{out}}} (\mathring{\chi}_s)_\gamma \mathbb{E}[Z^{(\mathrm{d}h_s^{3,k})_\gamma} Z^{(\mathrm{d}h_t^{3,k})_\beta}] \right), \tag{169}$$

where $Z^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}}$ is given by the Z MatMul rule,

$$Z^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} = \hat{Z}^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} + \dot{Z}^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} \tag{170}$$

$$= \hat{Z}^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} + \sum_{v \in \mathcal{V}: W_0^{3,k} v \in \mathcal{V}} Z^v \mathbb{E}\left[\frac{\partial Z^{\mathrm{d}h_t^{3,k}}}{\partial \hat{Z}^{W_0^{3,k} v}}\right], \tag{171}$$

where $\mathcal{V} := \{v \text{ for some vector in the program } v\}$ is the set of all vectors in the program.

tth Forward Pass:

$$Z^{\delta h_t^1} = -\eta \sum_{\beta=1}^{d_{\text{out}}} Z^{(\mathrm{d}\mathbf{h}_{t-1}^1)_\beta} (\mathring{\boldsymbol{\chi}}_{t-1})_\beta \boldsymbol{\xi}^T \boldsymbol{\xi}, \tag{172}$$

$$\delta \mathring{h}_{t,k}^2 = \mathring{\theta}'_{W_{t-1}^2} \mathring{\theta}_{\delta h_t^1} \mathbb{E}[Z^{(\overline{W_{t-1}^2})_{k,:}} Z^{\delta h_t^1}] - \mathring{\theta}'_{\delta W_t^2} \eta \sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathring{h}_{t-1,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_{t-1})_\beta \mathbb{E}[Z^{h_{t-1}^1} Z^{h_t^1}] + \delta \mathring{\mathbf{b}}^2 t, \tag{173}$$

$$\delta \mathring{\tilde{\mathbf{h}}}_t^2 = \begin{cases} \mathrm{G}(\mathring{\mathbf{h}}_t^2) - \mathrm{G}(\mathring{\mathbf{h}}_{t-1}^2) \text{ if } \mathring{\theta}_{\delta h_t^2} = 1 \\ \mathrm{G}'(\mathring{\mathbf{h}}_{t-1}^2)(\delta \mathring{\mathbf{h}}_t^2) \text{ if } \mathring{\theta}_{\delta h_t^2} = 0 \end{cases}, \tag{174}$$

$$Z^{\delta h_t^{3,k}} = \mathring{\theta}_{\delta h_t^1 / \delta h_t^{3,k}} Z^{W_0^{3,k} \delta h_t^1} - \sum_{s=0}^{t-2} \mathring{\theta}'_{\delta W_{s+1}^{3,k} \delta h_t^1 / \delta h_t^{3,k}} \eta \sum_{\beta}^{d_{\text{out}}} Z^{(\mathrm{d}h_s^{3,k})_\beta} (\mathring{\boldsymbol{\chi}}_s)_\beta \mathbb{E}[Z^{h_s^1} Z^{\delta h_t^1}] \tag{175}$$

$$- \mathring{\theta}'_{\delta W_t^{3,k} / \delta h_t^{3,k}} \eta \sum_{\beta}^{d_{\text{out}}} Z^{(\mathrm{d}h_{t-1}^{3,k})_\beta} (\mathring{\boldsymbol{\chi}}_{t-1})_\beta \mathbb{E}[Z^{h_{t-1}^1} Z^{h_t^1}], \tag{176}$$

$$Z^{\delta h_t^3} = \sum_{k=1}^{m} \mathring{\theta}_{\delta h_t^{3,k} / \delta h_t^3} \mathring{\tilde{h}}_{k,t-1}^2 Z^{\delta h_t^{3,k}} + \mathring{\theta}_{\delta \tilde{h}_t^2 / \delta h_t^3} \delta \mathring{\tilde{h}}_{t,k}^2 Z^{h_t^{3,k}}, \tag{177}$$

$$(\delta \mathring{f}_t)_\beta = \mathring{\theta}'_{W_{t-1}^4} \mathring{\theta}_{\delta h_t^3} \mathbb{E}[Z^{(\overline{W_{t-1}^4})_\beta} Z^{\delta h_t^3}] - \mathring{\theta}'_{\delta W_t^4} \eta (\mathring{\chi}_{t-1})_\beta \mathbb{E}[Z^{h_{t-1}^3} Z^{h_t^3}], \tag{178}$$

where G is either softmax or sigmoid and G' is the derivative, and

$$Z^{W_0^{3,k} \delta h_t^1} = \hat{Z}^{W_0^{3,k} \delta h_t^1} + \dot{Z}^{W_0^{3,k} \delta h_t^1} \tag{179}$$

$$= \hat{Z}^{W_0^{3,k} \delta h_t^1} + \sum_{s=0}^{t-1} \sum_{\beta=1}^{d_{\text{out}}} Z^{(\mathrm{d}h_s^{3,k})_\beta} \mathbb{E}\left[\frac{\partial Z^{\delta h_t^1}}{\partial \hat{Z}^{(W_0^{3,k})^T (\mathrm{d}h_s^{3,k})_\beta}} \right] \tag{180}$$

A.3 Tensor Program for Soft MoE with Adam

We consider Adam optimiser updates here.

Tensor Program: We use NE⊗OR$\top$ for the Adam limit [42]. This has the same operations as NETSOR$\top^+$ (Appendix A.1), but with a generalisation of the `Nonlin`.

- `OuterNonlin`: Given $r \geq 0$, vectors $\mathbf{x}^0, \ldots, \mathbf{x}^r$, and function $\psi : \mathbb{R}^{(r+1)+\ell} \to \mathbb{R}$, generate a new vector ($\mathbb{R}^n$) as

$$y_\alpha = \frac{1}{n^r} \sum_{\beta_1, \ldots, \beta_r = 1}^{n} \psi(x_\alpha^0; x_{\beta_1}^1; \ldots; x_{\beta_r}^r; \theta_1, \ldots, \theta_l)$$

Definition A.5 (Adam update)**.** Given gradients $g_0, \ldots, g_{t-1}$ evaluated at $\mathbf{W}_0, \ldots, \mathbf{W}_{t-1}$, respectively, the tth Adam update, for $t \geq 1$, with learning rate η is

$$\hat{m}_t := \frac{1}{1 - \beta_1^t} \sum_{s=0}^{t-1} (1 - \beta_1) \beta_1^{t-1-s} g_s,$$

$$\hat{v}_t := \frac{1}{1 - \beta_2^t} \sum_{s=0}^{t-1} (1 - \beta_2) \beta_2^{t-1-s} g_s^2,$$

$$\mathbf{W}_t := \mathbf{W}_{t-1} - \eta \frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon},$$

where (β_1, β_2) are momentum parameters, $\hat{v}_t$ is the bias-corrected second moment, and $\epsilon > 0$ is a constant added outside the square root. All operations are entrywise.

bcd parametrisation for Adam updates: The parameters b and c play the same role as in Appendix A.1. We parametrise the weights for layer l as $\mathbf{W}^l = n^{-b_l}\overline{\mathbf{W}}^l$. The learning rate for the parameter $\mathbf{W}^l$ is ηn^{-c_l}, where η is some width independent constant. The function Q_t^l used in the update $\mathbf{W}_t^l - \mathbf{W}_{t-1}^l$ takes the scaled gradient history through time $t-1$. For Adam, for $t \geq 1$,

$$Q_t^l(n^{d_l}g_0, \ldots, n^{d_l}g_{t-1}) = \frac{\frac{1}{1-\beta_1^t}\sum_{s=0}^{t-1}(1-\beta_1)\beta_1^{t-1-s}n^{d_l}g_s}{\sqrt{\frac{1}{1-\beta_2^t}\sum_{s=0}^{t-1}(1-\beta_2)\beta_2^{t-1-s}n^{2d_l}g_s^2} + \epsilon}. \tag{181}$$

Note that instead of scaling the gradients g by n^{d_l}, we can equivalently scale $\epsilon \to n^{-d_l}\epsilon$. Below, when only the latest gradient argument is displayed, $Q_t^l(n^{d_l}g_{t-1})$ abbreviates $Q_t^l(n^{d_l}g_0, \ldots, n^{d_l}g_{t-1})$ for the same parameter coordinate; the corresponding Tensor Program operations include the full gradient history. In particular, the historical update from g_s uses Q_{s+1}^l, and its associated weight increment is $\delta\mathbf{W}_{s+1}^l$.

Faithfulness (Definition 2.8): To ensure that the inputs to the update function Q are neither too large (causing diverging behavior) or too small (trivialising the update function), we require that the inputs to the update function scale $\Theta(1)$. Equivalently, for our Adam update function, we require that the scale of $n^{-d_l}\epsilon$ is the same as the scale of the gradients supplied to Q_t^l.

We assume that the parametrisation is stable (Setting A.1), and parameter values are chosen such that the network is stable at initialisation (Setting A.3).

First Backward Pass: Same as Appendix A.1.

First Weight Updates: Recall that we define the gradient of the loss with respect to the function as $\boldsymbol{\chi}_t$. For the vectors $\mathbf{W} \in \{\mathbf{W}^1, \mathbf{W}^2, \mathbf{W}^4\}$ we write $\delta\mathbf{W}_t := \theta_{\delta W_t}^{-1}(\mathbf{W}_t - \mathbf{W}_{t-1})$, and thus the weights at step t as $\mathbf{W}_t = \theta_{W_0}\overline{\mathbf{W}_0} + \ldots + \theta_{\delta W_t}\delta W_t$.

$$\mathbf{W}_1^4 - \mathbf{W}_0^4 = -n^{-c_4}\eta Q_1^4\left(n^{d_4}\boldsymbol{\chi}_0(\mathbf{h}_0^3)^T\right) \tag{182}$$

$$\theta_{\delta W_1^4} := n^{-c_4}. \tag{183}$$

$$\mathbf{W}_1^{3,k} - \mathbf{W}_0^{3,k} = -n^{-c_3}\eta Q_1^3\left(n^{(d_3-1)}\mathrm{d}\mathbf{h}_0^{3,k}\boldsymbol{\chi}_0(\mathbf{h}_0^1)^T\right), \tag{184}$$

$$\theta_{\delta W_1^3} := n^{-c_3}. \tag{185}$$

$$\mathbf{W}_1^2 - \mathbf{W}_0^2 = -n^{-c_2}\eta Q_1^2\left(n^{d_2}n\theta_{W_0^4}\mathrm{d}\mathbf{h}_0^2\boldsymbol{\chi}_0(\mathbf{h}_0^1)^T\right) \tag{186}$$

$$= -n^{-c_2}\eta Q_1^2\left(n^{d_2}\mathrm{d}\mathbf{h}_0^2\boldsymbol{\chi}_0(\mathbf{h}_0^1)^T\right) \tag{187}$$

$$\theta_{\delta W_1^2} := n^{-c_2}. \tag{188}$$

$$\mathbf{W}_1^1 - \mathbf{W}_0^1 = -n^{-c_1}\eta Q_1^1\left(n^{d_1}n\theta_{W_0^2}\theta_{W_0^4}\mathrm{d}\mathbf{h}_0^1\boldsymbol{\chi}_0(\boldsymbol{\xi})^T\right) \tag{189}$$

$$= -n^{-c_1}\eta Q_1^1\left(n^{(d_1-1)}\mathrm{d}\mathbf{h}_0^1\boldsymbol{\chi}_0(\boldsymbol{\xi})^T\right) \tag{190}$$

$$\theta_{\delta W_1^1} := n^{-c_1}. \tag{191}$$

The router bias in the learnable case updates as

$$\mathbf{b}_1^2 - \mathbf{b}_0^2 = -\eta\theta'_{W_0^4}\mathrm{d}\mathbf{h}_0^2\boldsymbol{\chi}_0, \tag{192}$$

where due to stability at initialisation (Setting A.3) $\theta'_{W_0^4} = 1$.

First Feature Updates: The feature updates are affected by the change in the previous layers features (*propagating updates*) and the update in the weights at the current layer (*effective updates*). We write the feature updates for the vectors $\mathbf{h} \in \{\mathbf{h}^1, \{\mathbf{h}^{3,k}\}_{k=1}^m, \mathbf{h}^3\}$ as $\delta\mathbf{h}_t := \theta_{\delta h_t}^{-1}(\mathbf{h}_t - \mathbf{h}_{t-1})$. This gives each features for the vectors at step t as $\mathbf{h}_t = \mathbf{h}_0 + \theta_{\delta h_1}\delta\mathbf{h}_1 + \ldots + \theta_{\delta h_t}\delta\mathbf{h}_t$. For the scalars $\mathbf{h} \in \{\mathbf{h}^2, \tilde{\mathbf{h}}^2, \mathbf{f}\}$ we simply write $\delta\mathbf{h}_t := (\mathbf{h}_t - \mathbf{h}_{t-1})$ and $\mathbf{h}_t = \mathbf{h}_0 + \ldots + \delta\mathbf{h}_t$.

$$\mathbf{h}_1^1 - \mathbf{h}_0^1 = (\mathbf{W}_1^1 - \mathbf{W}_0^1)\boldsymbol{\xi} \tag{193}$$

$$= -\theta_{\delta W_1^1}\eta Q_1^1 \left(n^{d_1} n^{-1} \mathrm{d}\mathbf{h}_0^1 \boldsymbol{\chi}_0 \boldsymbol{\xi}^T\right) \boldsymbol{\xi} \tag{194}$$

$$(\delta \mathbf{h}_1^1)_\alpha = \phi\left(\{(\mathrm{d}\mathbf{h}_0^1)_{\alpha\beta}\}_{\beta=1}^{d_{\text{out}}}; -\eta, n^{(d_1-1)}, \{(\boldsymbol{\chi}_0)_\beta\}_{\beta=1}^{d_{\text{out}}}, \boldsymbol{\xi}\right) \qquad \texttt{(Nonlin)} \tag{195}$$

$$\phi(\ldots) := -\eta Q_1^1 \left(n^{(d_1-1)} \sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathbf{h}_0^1)_{\alpha\beta} (\boldsymbol{\chi}_0)_\beta \boldsymbol{\xi}^T\right) \boldsymbol{\xi}, \tag{196}$$

$$\theta_{\delta h_1^1} := \theta_{\delta W_1^1}, \tag{197}$$

$$\mathbf{h}_1^2 - \mathbf{h}_0^2 = \mathbf{W}_0^2(\mathbf{h}_1^1 - \mathbf{h}_0^1) + (\mathbf{W}_1^2 - \mathbf{W}_0^2)\mathbf{h}_1^1 + (\mathbf{b}_1^2 - \mathbf{b}_0^2) \tag{198}$$

$$= \theta_{\delta h_1^1}\theta'_{W_0^2} \frac{\overline{\mathbf{W}_0^2}\delta \mathbf{h}_1^1}{n} - \theta'_{\delta W_1^2}\eta \frac{1}{n} Q_1^2 \left(n^{d_2} \mathrm{d}\mathbf{h}_0^2 \boldsymbol{\chi}_0 (\mathbf{h}_0^1)^T\right) \mathbf{h}_1^1 - \eta \mathrm{d}\mathbf{h}_0^2 \boldsymbol{\chi}_0, \tag{199}$$

$$\theta_{\delta h_1^2}(\delta \mathbf{h}_1^2)_k = \frac{1}{n}\sum_{\alpha=1}^{n} \phi\left(\overline{\mathbf{W}_0^2}_{k\alpha}, (\delta \mathbf{h}_1^1)_\alpha, (\mathbf{h}_0^1)_\alpha, (\mathbf{h}_1^1)_\alpha; \theta_{\delta h_1^1}, \theta'_{W_0^2}, \theta'_{\delta W_1^2}, n^{d_2}, \eta, \{(\mathrm{d}\mathbf{h}_0^2)_{k\beta}, (\boldsymbol{\chi}_0)_\beta\}_\beta^{d_{\text{out}}}\right) \quad \texttt{(Moment)} \tag{200}$$

$$\phi(\ldots) := \theta_{\delta h_1^1}\theta'_{W_0^2} \overline{\mathbf{W}_0^2}_{k\alpha} (\delta \mathbf{h}_1^1)_\alpha - \theta'_{\delta W_1^2}\eta \frac{1}{n} Q_1^2 \left(n^{d_2} \sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathbf{h}_0^2)_{k\beta} (\boldsymbol{\chi}_0)_\beta (\mathbf{h}_0^1)_\alpha\right) (\mathbf{h}_1^1)_\alpha \tag{201}$$

$$- \eta\theta'_{W_0^4} \mathrm{d}\mathbf{h}_0^2 \boldsymbol{\chi}_0, \tag{202}$$

$$\theta_{\delta h_1^2} := \max(\theta_{\delta h_1^1}\theta'_{W_0^2}, \theta'_{\delta W_1^2}), \tag{203}$$

$$\tilde{\mathbf{h}}_1^2 - \tilde{\mathbf{h}}_0^2 = \mathrm{G}(\mathbf{h}_1^2) - \mathrm{G}(\mathbf{h}_0^2) \tag{204}$$

$$= \mathrm{G}\left(\mathbf{h}_0^2 + \theta_{\delta h_1^2}\delta \mathbf{h}_1^2\right) - \mathrm{G}(\mathbf{h}_0^2) \tag{205}$$

$$\theta_{\delta \tilde{h}_1^2}(\delta \tilde{\mathbf{h}}_1^2)_k = \frac{1}{n}\sum_{\alpha=1}^{n} \phi_k\left(-; \{(\mathbf{h}_0^2)_k, (\delta \mathbf{h}_1^2)_k\}_{k=1}^m\right), \qquad \texttt{(Moment)}, \tag{206}$$

$$\theta_{\delta \tilde{h}_1^2} := \theta_{\delta h_1^2}. \tag{207}$$

where G is either softmax or sigmoid and can be represented as a Moment (see Appendix E) [41].

$$\mathbf{h}_1^{3,k} - \mathbf{h}_0^{3,k} = \mathbf{W}_0^{3,k}(\mathbf{h}_1^1 - \mathbf{h}_0^1) + (\mathbf{W}_1^{3,k} - \mathbf{W}_0^{3,k})\mathbf{h}_1^1, \tag{208}$$

$$= \theta_{\delta h_1^1}\mathbf{W}_0^{3,k}\delta\mathbf{h}_1^1 - \theta'_{\delta W_1^3}\eta\frac{1}{n}Q_1^3\left(n^{(d_3-1)}\mathrm{d}\mathbf{h}_0^{3,k}\boldsymbol{\chi}_0(\mathbf{h}_0^1)^T\right)(\mathbf{h}_1^1), \tag{209}$$

$$(\delta\mathbf{h}_1^{3,k})_\alpha = \frac{1}{n}\sum_{\gamma=1}^{n}\phi\Bigg(\Upsilon_\alpha, \{(\mathrm{d}\mathbf{h}_0^{3,k})_{\alpha l}\}_{l=1}^{d_{\text{out}}}; (\mathbf{h}_0^1)_\gamma, (\mathbf{h}_1^1)_\gamma \tag{OuterNonlin}$$

$$; \theta_{\delta h_1^1/\delta h_1^{3,k}}, \theta'_{\delta W_1^3/\delta h_1^{3,k}}, n^{(d_3-1)}, \{(\boldsymbol{\chi}_0)_l\}_{l=1}^{d_{\text{out}}}, \eta\Bigg), \tag{210}$$

$$\Upsilon := \mathbf{W}_0^{3,k}\delta\mathbf{h}_1^1, \tag{MatMul}$$

$$\phi(\ldots) := \theta_{\delta h_1^1/\delta h_1^{3,k}}\Upsilon_\alpha - \theta'_{\delta W_1^3/\delta h_1^{3,k}}\eta Q_1^3\left(n^{(d_3-1)}\sum_{l=1}^{d_{\text{out}}}(\mathrm{d}\mathbf{h}_0^{3,k})_{\alpha l}(\boldsymbol{\chi}_0)_l(\mathbf{h}_0^1)_\gamma\right)(\mathbf{h}_1^1)_\gamma, \tag{212}$$

$$\theta_{\delta h_1^{3,k}} = \max(\theta_{\delta h_1^1}, \theta'_{\delta W_1^3}), \tag{213}$$

$$\theta_{\delta h_1^1/\delta h_1^{3,k}} := \theta_{\delta h_1^1}/\theta_{\delta h_1^{3,k}}, \tag{214}$$

$$\theta'_{\delta W_1^3/\delta h_1^{3,k}} := \theta'_{\delta W_1^3}/\theta_{\delta h_1^{3,k}}. \tag{215}$$

We repeat the above process for each expert k to give $\{\delta\mathbf{h}_1^{3,k}\}_{k=1}^m$.

$$\mathbf{h}_1^3 - \mathbf{h}_0^3 = \sum_{k=1}^{m}\left(\tilde{h}_{0,k}^2(\mathbf{h}_1^{3,k} - \mathbf{h}_0^{3,k}) + (\tilde{h}_{1,k}^2 - \tilde{h}_{0,k}^2)\mathbf{h}_1^{3,k}\right) \tag{216}$$

$$= \sum_{k=1}^{m}\left(\theta_{\delta h_1^{3,k}}\tilde{h}_{0,k}^2\delta\mathbf{h}_1^{3,k} + \theta_{\delta\tilde{h}_1^2}\delta\tilde{h}_{1,k}^2\mathbf{h}_1^{3,k}\right), \tag{217}$$

$$(\delta\mathbf{h}_1^3)_\alpha = \phi\left(\{(\delta\mathbf{h}_1^{3,k})_\alpha\}_{k=1}^m, \{(\mathbf{h}_1^{3,k})_\alpha\}_{k=1}^m; \theta_{\delta h_1^{3,k}/\delta h_1^3}, \theta_{\delta\tilde{h}_1^2/\delta h_1^3}, \{\tilde{h}_{0,k}^2\}_{k=1}^m, \{\delta\tilde{h}_{1,k}^2\}_{k=1}^m\right), \tag{Nonlin}$$

$$\phi(\ldots) := \sum_{k=1}^{m}\left(\theta_{\delta h_1^{3,k}/\delta h_1^3}\tilde{h}_{0,k}^2(\delta\mathbf{h}_1^{3,k})_\alpha + \theta_{\delta\tilde{h}_1^2/\delta h_1^3}\delta\tilde{h}_{1,k}^2(\mathbf{h}_1^{3,k})_\alpha\right), \tag{219}$$

$$\theta_{\delta h_1^3} := \max(\theta_{\delta h_1^{3,k}}, \theta_{\delta\tilde{h}_1^2}), \tag{220}$$

$$\theta_{\delta h_1^{3,k}/\delta h_1^3} := \theta_{\delta h_1^{3,k}}/\theta_{\delta h_1^3}, \tag{221}$$

$$\theta_{\delta\tilde{h}_1^2/\delta h_1^3} := \theta_{\delta\tilde{h}_1^2}/\theta_{\delta h_1^3}. \tag{222}$$

$$\mathbf{f}_1 - \mathbf{f}_0 = \mathbf{W}_0^4(\mathbf{h}_1^3 - \mathbf{h}_0^3) + (\mathbf{W}_1^4 - \mathbf{W}_0^4)\mathbf{h}_1^3 \tag{223}$$

$$= \theta'_{W_0^4}\theta_{\delta h_1^3}\frac{\overline{\mathbf{W}_0^4}\delta\mathbf{h}_1^3}{n} - \theta'_{\delta W_1^4}\eta\frac{1}{n}Q_1^4\left(n^{d_4}\boldsymbol{\chi}_0(\mathbf{h}_0^3)^T\right)\mathbf{h}_1^3, \tag{224}$$

$$\theta_{\delta f_1}(\delta\mathbf{f}_1)_\beta = \frac{1}{n}\sum_{\alpha=1}^{n}\phi\left(\overline{\mathbf{W}_0^4}_{\beta\alpha}, (\delta\mathbf{h}_1^3)_\alpha, (\mathbf{h}_0^3)_\alpha, (\mathbf{h}_1^3)_\alpha; \theta'_{W_0^4}, \theta_{\delta h_1^3}, \theta'_{\delta W_1^4}, n^{d_4}, \eta, (\boldsymbol{\chi}_0)_\beta\right), \tag{Moment}$$

$$\phi(\ldots) := \theta'_{W_0^4}\theta_{\delta h_1^3}(\overline{\mathbf{W}_0^4})_{\beta\alpha}(\delta\mathbf{h}_1^3)_\alpha - \theta'_{\delta W_1^4}\eta Q_1^4\left(n^{d_4}(\boldsymbol{\chi}_0)_\beta(\mathbf{h}_0^3)_\alpha\right)(\mathbf{h}_1^3)_\alpha, \tag{226}$$

$$\theta_{\delta f_1} := \max(\theta'_{W_0^4}\theta_{\delta h_1^4}, \theta'_{\delta W_1^4}). \tag{227}$$

***t*th Backward Pass:** Same as Appendix A.1 except for the gradient $\mathrm{d}\mathbf{h}_t^1$.

$$\mathrm{d}\mathbf{h}_t^1 = \sum_{k=1}^{m} (\overline{\mathbf{W}_t^2})_{:,k}^T \mathrm{d}h_{t,k}^2 + \sum_{k=1}^{m} (\mathbf{W}_0^{3,k})^T \mathrm{d}\mathbf{h}_t^{3,k} - \sum_{k=1}^{m} \sum_{s=0}^{t-1} \eta \frac{1}{n} Q_{s+1}^3 \left(n^{(d_3-1)} \mathbf{h}_s^1 (\boldsymbol{\chi}_s)^T (\mathrm{d}\mathbf{h}_s^{3,k})^T \right) \mathrm{d}\mathbf{h}_t^{3,k}, \quad \in \mathbb{R}^{n \times d_{\text{out}}}, \tag{228}$$

$$(\mathrm{d}\mathbf{h}_t^1)_{\alpha\beta} = \frac{1}{n} \sum_{\gamma=1}^{n} \phi \Bigg(\{(\overline{\mathbf{W}_t^2})_{\alpha k}^T, (\mathbf{V}_t^k)_{\alpha\beta}\}_{k=1}^m, \{(\mathbf{h}_s^1)_\alpha\}_{s=0}^{t-1}; \{(\mathrm{d}\mathbf{h}_s^{3,k})_{\gamma\beta}\}_{s=0}^{t-1}, (\mathrm{d}\mathbf{h}_t^{3,k})_{\gamma\beta}; n^{(d_3-1)}, \{(\mathrm{d}h_{t,k}^2)_\beta\}_{k=1}^m, \{(\boldsymbol{\chi}_s)_\beta\}_{s=0}^{t-1} \Bigg), \tag{229}$$

(OuterNo

$$(\mathbf{V}_t^k)_{:,\beta} := (\mathbf{W}_0^{3,k})^T (\mathrm{d}\mathbf{h}_t^{3,k})_{:,\beta}, \tag{230}$$

(MatMul)

$$\phi(\ldots) := \sum_{k=1}^{m} (\overline{\mathbf{W}_t^2})_{\alpha k}^T (\mathrm{d}h_{t,k}^2)_\beta + \sum_{k=1}^{m} (\mathbf{V}_t^k)_{\alpha\beta} - \sum_{k=1}^{m} \sum_{s=0}^{t-1} \eta \frac{1}{n} Q_{s+1}^3 \left(n^{(d_3-1)} (\mathbf{h}_s^1)_\alpha (\boldsymbol{\chi}_s)_\beta (\mathrm{d}\mathbf{h}_s^{3,k})_{\gamma\beta} \right) (\mathrm{d}\mathbf{h}_t^{3,k})_{\gamma\beta}, \tag{231}$$

***t*th Weight Updates:**

$$\mathbf{W}_t^4 - \mathbf{W}_{t-1}^4 = -n^{-c_4} \eta Q_t^4 \left(n^{d_4} \boldsymbol{\chi}_{t-1} (\mathbf{h}_{t-1}^3)^T \right) \tag{232}$$

$$\theta_{\delta W_t^4} := n^{-c_4}. \tag{233}$$

$$\mathbf{W}_t^{3,k} - \mathbf{W}_{t-1}^{3,k} = -n^{-c_3} \eta Q_t^3 \left(n^{d_3} \theta_{W_{t-1}^4} \mathrm{d}\mathbf{h}_{t-1}^{3,k} \boldsymbol{\chi}_{t-1} (\mathbf{h}_{t-1}^1)^T \right) \tag{234}$$

$$\theta_{\delta W_t^3} := n^{-c_3}. \tag{235}$$

$$\mathbf{W}_t^2 - \mathbf{W}_{t-1}^2 = -n^{-c_2} \eta Q_t^2 \left(n^{d_2} n \theta_{W_{t-1}^4} \mathrm{d}\mathbf{h}_{t-1}^2 \boldsymbol{\chi}_{t-1} (\mathbf{h}_{t-1}^1)^T \right) \tag{236}$$

$$\theta_{\delta W_t^2} := n^{-c_2}. \tag{237}$$

$$\mathbf{W}_t^1 - \mathbf{W}_{t-1}^1 = -n^{-c_1} \eta Q_t^1 \left(n^{d_1} n \theta_{W_{t-1}^4} \theta_{W_{t-1}^2} \mathrm{d}\mathbf{h}_{t-1}^1 \boldsymbol{\chi}_{t-1} (\boldsymbol{\xi})^T \right) \tag{238}$$

$$\theta_{\delta W_t^1} := n^{-c_1}. \tag{239}$$

$$\mathbf{b}_t^2 - \mathbf{b}_{t-1}^2 = -\eta \theta'_{W_{t-1}^4} \mathrm{d}\mathbf{h}_{t-1}^2 \boldsymbol{\chi}_{t-1}, \tag{240}$$

***t*th Feature Updates:**

$$\mathbf{h}_t^1 - \mathbf{h}_{t-1}^1 = (\mathbf{W}_t^1 - \mathbf{W}_{t-1}^1) \boldsymbol{\xi} \tag{241}$$

$$= -\theta_{\delta W_t^1} \eta Q_t^1 \left(n^{d_1} \theta'_{W_{t-1}^2} \theta_{W_{t-1}^4} \mathrm{d}\mathbf{h}_{t-1}^1 \boldsymbol{\chi}_{t-1} \boldsymbol{\xi}^T \right) \boldsymbol{\xi} \tag{242}$$

$$(\delta \mathbf{h}_t^1)_\alpha = \phi \left(\{(\mathrm{d}\mathbf{h}_{t-1}^1)_{\alpha\beta}\}_{\beta=1}^{d_{\text{out}}}; n^{d_1} \theta'_{W_{t-1}^2} \theta_{W_{t-1}^4}, -\eta, \{(\boldsymbol{\chi}_{t-1})_\beta\}_{\beta=1}^{d_{\text{out}}}, \boldsymbol{\xi} \right) \tag{243}$$

(Nonlin)

$$\phi(\ldots) := -\eta Q_t^1 \left(n^{d_1} \theta'_{W_{t-1}^2} \theta_{W_{t-1}^4} \sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathbf{h}_{t-1}^1)_{\alpha\beta} (\boldsymbol{\chi}_{t-1})_\beta \boldsymbol{\xi}^T \right) \boldsymbol{\xi}, \tag{244}$$

$$\theta_{\delta h_t^1} := \theta_{\delta W_t^1}, \tag{245}$$

$$\mathbf{h}_t^2 - \mathbf{h}_{t-1}^2 = \mathbf{W}_{t-1}^2(\mathbf{h}_t^1 - \mathbf{h}_{t-1}^1) + (\mathbf{W}_t^2 - \mathbf{W}_{t-1}^2)\mathbf{h}_t^1 + (\mathbf{b}_t^2 - \mathbf{b}_{t-1}^2) \tag{246}$$

$$= \theta'_{W_{t-1}^2}\theta_{\delta h_t^1}\frac{\overline{\mathbf{W}_{t-1}^2}\delta\mathbf{h}_t^1}{n} - \theta'_{\delta W_t^2}\eta\frac{1}{n}Q_t^2\left(n^{d_2}\theta'_{W_{t-1}^4}\mathrm{d}\mathbf{h}_{t-1}^2\boldsymbol{\chi}_{t-1}(\mathbf{h}_{t-1}^1)^T\right)\mathbf{h}_t^1 \tag{247}$$

$$- \eta\theta'_{W_{t-1}^4}\mathrm{d}\mathbf{h}_{t-1}^2\boldsymbol{\chi}_{t-1} \tag{248}$$

$$\theta_{h_t^2}(\delta\mathbf{h}_t^2)_k = \frac{1}{n}\sum_{\alpha=1}^{n}\phi\Bigg(\overline{\mathbf{W}_{t-1}^2}_{k\alpha}, (\delta\mathbf{h}_t^1)_\alpha, (\mathbf{h}_{t-1}^1)_\alpha, (\mathbf{h}_t^1)_\alpha; \tag{Moment}$$

$$\theta'_{W_{t-1}^2}, \theta_{\delta h_t^1}, \theta'_{\delta W_t^2}, n^{d_2}, \theta'_{W_{t-1}^4}, \eta, \{(\mathrm{d}\mathbf{h}_{t-1}^2)_{k\beta}, (\boldsymbol{\chi}_{t-1})_\beta\}_\beta^{d_{\text{out}}}\Bigg) \tag{249}$$

$$\phi(\ldots) := \theta'_{W_{t-1}^2}\theta_{\delta h_t^1}\overline{\mathbf{W}_{t-1}^2}_{k\alpha}(\delta\mathbf{h}_t^1)_\alpha \tag{250}$$

$$- \theta'_{\delta W_t^2}\eta Q_t^2\left(n^{d_2}\theta'_{W_{t-1}^4}\sum_{\beta=1}^{d_{\text{out}}}(\mathrm{d}\mathbf{h}_{t-1}^2)_{k\beta}(\boldsymbol{\chi}_{t-1})_\beta(\mathbf{h}_{t-1}^1)_\alpha\right)(\mathbf{h}_t^1)_\alpha \tag{251}$$

$$- \eta\theta'_{W_{t-1}^4}\mathrm{d}\mathbf{h}_{t-1}^2\boldsymbol{\chi}_{t-1} \tag{252}$$

$$\theta_{\delta h_t^2} := \max(\theta'_{W_{t-1}^2}\theta_{\delta h_t^1}, \theta'_{\delta W_t^2}, \theta'_{W_{t-1}^4}), \tag{253}$$

$$\tilde{\mathbf{h}}_t^2 - \tilde{\mathbf{h}}_{t-1}^2 = \mathrm{G}(\mathbf{h}_t^2) - \mathrm{G}(\mathbf{h}_{t-1}^2) \tag{254}$$

$$= \mathrm{G}\left(\mathbf{h}_{t-1}^2 + \theta_{h_t^2}\delta\mathbf{h}_t^2\right) - \mathrm{Softmax}(\mathbf{h}_{t-1}^2) \tag{255}$$

$$\theta_{\tilde{h}_t^2}(\delta\tilde{\mathbf{h}}_t^2)_k = \frac{1}{n}\sum_{\alpha=1}^{n}\phi_k\left(-; \{(\mathbf{h}_{t-1}^2)_k, (\delta\mathbf{h}_t^2)_k\}_{k=1}^m\right), \qquad \texttt{(Moment)}, \tag{256}$$

$$\theta_{\delta\tilde{h}_t^2} := \theta_{\delta h_t^2}. \tag{257}$$

$$\mathbf{h}_t^{3,k} - \mathbf{h}_{t-1}^{3,k} = \mathbf{W}_{t-1}^{3,k}(\mathbf{h}_t^1 - \mathbf{h}_{t-1}^1) + (\mathbf{W}_t^{3,k} - \mathbf{W}_{t-1}^{3,k})\mathbf{h}_t^1, \tag{258}$$

$$= \theta_{\delta h_t^1}\mathbf{W}_0^{3,k}\overline{\delta\mathbf{h}_t^1} - \sum_{s=0}^{t-2}\theta'_{\delta W_{s+1}^3}\theta_{\delta h_t^1}\frac{\eta}{n}Q_{s+1}^3\left(n^{d_3}\theta_{W_s^4}\mathrm{d}\mathbf{h}_s^{3,k}\boldsymbol{\chi}_s(\mathbf{h}_s^1)^T\right)\delta\mathbf{h}_t^1 \tag{259}$$

$$- \theta'_{\delta W_t^3}\frac{\eta}{n}Q_t^3\left(n^{d_3}\theta_{W_{t-1}^4}\mathrm{d}\mathbf{h}_{t-1}^{3,k}\boldsymbol{\chi}_{t-1}(\mathbf{h}_{t-1}^1)^T\right)(\mathbf{h}_t^1), \tag{260}$$

$$(\delta\mathbf{h}_t^{3,k})_\alpha = \frac{1}{n}\sum_{\gamma=1}^{n}\phi\Bigg(\Upsilon_\alpha, \{\{(\mathrm{d}\mathbf{h}_s^{3,k})_{\alpha l}\}_{l=1}^{d_{\text{out}}}\}_{s=0}^{t-2}, \{(\mathrm{d}\mathbf{h}_{t-1}^{3,k})_{\alpha l}\}_{l=1}^{d_{\text{out}}}; \tag{OuterNonlin}$$

$$\{(\mathbf{h}_s^1)_\gamma\}_{s=0}^{t-2}, (\delta\mathbf{h}_t^1)_\gamma, (\mathbf{h}_{t-1}^1)_\gamma, (\mathbf{h}_t^1)_\gamma;$$

$$\theta_{\delta h_t^1/\delta h_t^{3,k}}, \{\theta'_{\delta W_{s+1}^3\delta h_t^1/\delta h_t^{3,k}}\}_{s=0}^{t-2}, \theta'_{\delta W_t^3/\delta h_t^{3,k}}, n^{d_3}, \{\theta_{W_s^4}\}_{s=0}^{t-1}\Bigg) \tag{261}$$

$$\Upsilon := \mathbf{W}_0^{3,k}\delta\mathbf{h}_t^1, \tag{MatMul}$$

(262)

$$\theta_{\delta h_t^{3,k}} = \max\left(\theta_{\delta h_t^1}, \max_{0\le s\le t-2}\left(\theta'_{\delta W_{s+1}^3}\theta_{\delta h_t^1}\right), \theta'_{\delta W_t^3}\right), \tag{263}$$

$$\theta_{\delta h_t^1/\delta h_t^{3,k}} := \theta_{\delta h_t^1}/\theta_{\delta h_t^{3,k}}, \tag{264}$$

$$\theta'_{\delta W_{s+1}^3\delta h_t^1/\delta h_t^{3,k}} := \theta'_{\delta W_{s+1}^3}\theta_{\delta h_t^1}/\theta_{\delta h_t^{3,k}}, \tag{265}$$

$$\theta'_{\delta W_t^3/\delta h_t^{3,k}} := \theta'_{\delta W_t^3}/\theta_{\delta h_t^{3,k}}. \tag{266}$$

We repeat the above process for each expert k.

$$\mathbf{h}_t^3 - \mathbf{h}_{t-1}^3 = \sum_{k=1}^m \left(\tilde{h}_{t-1,k}^2 (\mathbf{h}_t^{3,k} - \mathbf{h}_{t-1}^{3,k}) + (\tilde{h}_{t,k}^2 - \tilde{h}_{t-1,k}^2) \mathbf{h}_t^{3,k} \right) \tag{267}$$

$$= \sum_{k=1}^m \left(\theta_{\delta h_t^{3,k}} \tilde{h}_{t-1,k}^2 \delta \mathbf{h}_t^{3,k} + \theta_{\delta \tilde{h}_t^2} \delta \tilde{h}_{t,k}^2 \mathbf{h}_t^{3,k} \right), \tag{268}$$

$$(\delta \mathbf{h}_t^3)_\alpha = \phi \left(\{(\delta \mathbf{h}_t^{3,k})_\alpha\}_{k=1}^m, \{(\mathbf{h}_t^{3,k})_\alpha\}_{k=1}^m; \theta_{\delta h_t^{3,k}/\delta h_t^3}, \theta_{\delta \tilde{h}_t^2/\delta h_t^3}, \{\tilde{h}_{t-1,k}^2\}_{k=1}^m, \{\delta \tilde{h}_{t,k}^2\}_{k=1}^m \right), \quad \texttt{(Nonlin)} \tag{269}$$

$$\phi(\ldots) := \sum_{k=1}^m \left(\theta_{\delta h_t^{3,k}} \tilde{h}_{t-1,k}^2 (\delta \mathbf{h}_t^{3,k})_\alpha + \theta_{\delta \tilde{h}_t^2} \delta \tilde{h}_{t,k}^2 (\mathbf{h}_t^{3,k})_\alpha \right), \tag{270}$$

$$\theta_{\delta h_t^3} := \max(\theta_{\delta h_t^{3,k}}, \theta_{\delta \tilde{h}_t^2}), \tag{271}$$

$$\theta_{\delta h_t^{3,k}/\delta h_t^3} := \theta_{\delta h_t^{3,k}} / \theta_{\delta h_t^3}, \tag{272}$$

$$\theta_{\delta \tilde{h}_t^2/\delta h_t^3} := \theta_{\delta \tilde{h}_t^2} / \theta_{\delta h_t^3}. \tag{273}$$

$$\mathbf{f}_t - \mathbf{f}_{t-1} = \mathbf{W}_{t-1}^4 (\mathbf{h}_t^3 - \mathbf{h}_{t-1}^3) + (\mathbf{W}_t^4 - \mathbf{W}_{t-1}^4) \mathbf{h}_t^3 \tag{274}$$

$$= \theta'_{W_{t-1}^4} \theta_{\delta h_t^3} \frac{\overline{\mathbf{W}_{t-1}^4} \delta \mathbf{h}_t^3}{n} - \theta'_{\delta W_t^4} \eta \frac{1}{n} Q_t^4 \left(n^{d_4} \boldsymbol{\chi}_{t-1} (\mathbf{h}_{t-1}^3)^T \right) \mathbf{h}_t^3, \tag{275}$$

$$\theta_{f_t} (\delta \mathbf{f}_t)_\beta = \frac{1}{n} \sum_{\alpha=1}^n \phi \left(\overline{\mathbf{W}_{t-1}^4}_{\beta\alpha}, (\delta \mathbf{h}_t^3)_\alpha, (\mathbf{h}_{t-1}^3)_\alpha, (\mathbf{h}_t^3)_\alpha; \theta'_{W_{t-1}^4}, \theta_{\delta h_t^3}, \theta'_{\delta W_t^4}, n^{d_4}, \eta, (\boldsymbol{\chi}_{t-1})_\beta \right), \quad \texttt{(Moment)} \tag{276}$$

$$\phi(\ldots) := \theta'_{W_{t-1}^4} \theta_{\delta h_t^3} (\overline{\mathbf{W}_{t-1}^4})_{\beta\alpha} (\delta \mathbf{h}_t^3)_\alpha - \theta'_{\delta W_t^4} \eta Q_t^4 \left(n^{d_4} (\boldsymbol{\chi}_{t-1})_\beta (\mathbf{h}_{t-1}^3)_\alpha \right) (\mathbf{h}_t^3)_\alpha, \tag{277}$$

$$\theta_{\delta f_t} := \max(\theta'_{W_{t-1}^4} \theta_{\delta h_t^4}, \theta'_{\delta W_t^4}). \tag{278}$$

A.4 Infinite Width Limit for Soft MoE with Adam

See Appendix A.2 for an explanation of the limit notation.

Adam requires defining the limit of an extra operation, `OuterNonlin` [42]:
Z OuterNonlin: Given $r \geq 0$ and function $\psi : \mathbb{R}^{(r+1)+\ell} \to \mathbb{R}$ such that $y_\alpha = \frac{1}{n^r} \sum_{\beta_1,\ldots,\beta_r=1}^n \psi(x_\alpha^0; x_{\beta_1}^1; \ldots; x_{\beta_r}^r; \theta_1, \ldots, \theta_l)$, then

$$Z^y := \mathbb{E}_{Z_1^{x^1}, \ldots Z_1^{x^r}} \left[\psi \left(Z^{x^0}; Z_1^{x^1}; \ldots; Z_1^{x^r}; \mathring{\theta}_1, \ldots, \mathring{\theta}_l \right) \right], \tag{279}$$

where $(Z_1^{x^1}, \ldots Z_1^{x^r}) \overset{d}{=} (Z^{x^1}, \ldots Z^{x^r})$ is an iid copy of $(Z^{x^1}, \ldots Z^{x^r})$ and is independent of Z^{x^0}. For Adam, the copied variables are jointly sampled over all time indices used. An expectation involving Q_t^l includes its copied gradient history through time $t-1$ and any copied variables multiplying its output, even when only the latest copied variables are displayed in the subscript.

Assumption A.6 (Pseudo Lipschitz Q, Assumption 2.3.2 [42])**.** Assume that Q_t^ℓ, ϵ are pseudo-Lipschitz for all ℓ, t.

Assumption A.7 (Faithfulness at initialisation)**.** If we want faithfulness in the first weight updates, it already restricts the values of d to $d_1 = 1, d_2 = 0, d_3 = 1, d_4 = 0$.

First Forward Pass:

$$Z^{h_0^1} = \sum_{j=1}^{d_{\text{in}}} \xi_j Z^{W_{0,j}^1}, \tag{280}$$

$$\mathring{\mathbf{h}}_0^2 = \mathbf{b}_0^2, \tag{281}$$

$$\mathring{\tilde{\mathbf{h}}}_0^2 = G(\mathbf{b}_0^2) \tag{282}$$

$$Z^{h_0^{3,k}} = Z^{W_0^{3,k} h_0^1} \text{ for every expert } k, \tag{283}$$

$$Z^{h_0^3} = \sum_{k=1}^{m} \mathring{\tilde{h}}_{0,k}^2 Z^{h_0^{3,k}} \tag{284}$$

$$\mathring{\mathbf{f}}_0 = \mathbf{0}. \tag{285}$$

Here, $Z^{W_0^1}$ is standard Gaussian by construction. $\mathring{\mathbf{h}}_0^2 = \mathbf{0}$ as $\mathring{h}_0^2 = \mathbb{E}[Z^{W_0^2}]\mathbb{E}[Z^{h_0^1}] = 0$. $\mathring{\tilde{\mathbf{h}}}_0^2$ then follows by the definition of $\mathrm{Softmax}(\mathbf{0})$. $Z^{W_0^{3,k} h_0^1} = \hat{Z}^{W_0^{3,k} h_0^1}$ from the Z MatMul rule. Finally $\mathring{\mathbf{f}}_0 = \mathbf{0}$ as $\mathring{f}_0 = \mathbb{E}[Z^{W_0^4}]\mathbb{E}[Z^{h_0^3}] = 0$.

First Backward Pass:

$$Z^{\mathrm{d}h_0^3} = Z^{\overline{W_0^4}}, \tag{286}$$

$$\mathring{\mathrm{d}\tilde{h}}_{0,k}^2 = 0 \text{ for every expert } k, \tag{287}$$

$$\mathring{\mathrm{d}h}^2{}_{0,k} = 0 \text{ for every expert } k, \tag{288}$$

$$Z^{\mathrm{d}h_0^{3,k}} = \mathring{\tilde{\mathbf{h}}}_{0,k}^2 Z^{\mathrm{d}h_0^3} \text{ for every expert } k, \tag{289}$$

$$Z^{\mathrm{d}h_0^1} = \sum_{k=1}^{m} Z^{(W_0^{3,k})^T \mathrm{d}h_0^{3,k}} \tag{290}$$

$$\tag{291}$$

$Z^{\overline{W_0^4}}$ is a standard Gaussian by construction. $\mathring{\mathrm{d}\tilde{h}}_{0,k}^2 = \mathbb{E}[Z^{h_0^{3,k}}]\mathbb{E}[\overline{W_0^4}] = 0$. From this, $\mathring{\mathrm{d}h}^2{}_{0,k} = 0$ follows. $Z^{(W_0^{3,k})^T \mathrm{d}h_0^3} = \hat{Z}^{(W_0^{3,k})^T \mathrm{d}h_0^3}$ from the Z MatMul rule.

First Feature Update:

$$Z^{\delta h_1^1} = -\eta Q_1^1 \left(\sum_{\beta=1}^{d_{\text{out}}} Z^{(\mathrm{d}h_0^1)_\beta} (\mathring{\boldsymbol{\chi}}_0)_\beta \boldsymbol{\xi}^T \right) \boldsymbol{\xi}, \tag{292}$$

$$\delta \mathring{h}_{1,k}^2 = \mathring{\theta}_{\delta h_1^1} \mathbb{E}[Z^{(\overline{W_0^2})_{k,:}} Z^{\delta h_1^1}] - \mathring{\theta}'_{\delta W_1^2} \eta \mathbb{E}\left[Q_1^2 \left(\sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathring{h}_{0,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_0)_\beta Z^{h_0^1} \right) Z^{h_1^1} \right], \tag{293}$$

$$\delta \mathring{\tilde{\mathbf{h}}}_1^2 = \begin{cases} \mathrm{G}(\mathring{\mathbf{h}}_1^2) - \mathrm{G}(\mathring{\mathbf{h}}_0^2) \text{ if } \mathring{\theta}_{\delta h_1^2} = 1 \\ \mathrm{G}'(\mathring{\mathbf{h}}_0^2)(\delta \mathring{\mathbf{h}}_1^2) \text{ if } \mathring{\theta}_{\delta h_1^2} = 0 \end{cases}, \tag{294}$$

$$\begin{aligned} Z^{\delta h_1^{3,k}} &= \mathring{\theta}_{\delta h_1^1 / \delta h_1^{3,k}} Z^{W_0^{3,k} \delta h_1^1} \\ &\quad - \mathring{\theta}'_{\delta W_1^3 / \delta h_1^{3,k}} \eta \mathbb{E}_{\left(Z^{h_0^1}, Z_1^{h_1^1}\right)} \left[Q_1^3 \left(\sum_{\beta=1}^{d_{\text{out}}} Z^{(\mathrm{d}h^{3,k})_\beta} (\mathring{\boldsymbol{\chi}}_0)_\beta Z_1^{h_0^1} \right) Z_1^{h_1^1} \right], \text{ for every expert } k, \end{aligned} \tag{295}$$

$$Z^{\delta h_1^3} = \sum_{k=1}^{m} \mathring{\theta}_{\delta h_1^{3,k} / \delta h_1^3} \mathring{\tilde{h}}_{0,k}^2 Z^{\delta h_1^{3,k}} + \mathring{\theta}_{\delta \tilde{h}_1^2} \mathring{\theta}_{h_1^{3,k} / \delta h^3} \delta \mathring{\tilde{h}}_{1,k}^2 Z^{h_1^{3,k}}, \tag{296}$$

$$\delta \mathring{f}_1 = \mathring{\theta}_{\delta h_1^3} \mathbb{E}[Z^{\overline{W_0^4}} Z^{\delta h_1^3}] - \mathring{\theta}'_{\delta W_1^4} \eta \mathbb{E}\left[Q_1^4 \left(\mathring{\chi}_0 Z^{h_0^3} \right) Z^{h_1^3} \right], \tag{297}$$

where $Z^{W_0^{3,k}\delta h_1^1}$ is given by the Z MatMul rule,

$$Z^{W_0^{3,k}\delta h_1^1} = \hat{Z}^{W_0^{3,k}\delta h_1^1} + \dot{Z}^{W_0^{3,k}\delta h_1^1} \tag{298}$$

$$= \hat{Z}^{W_0^{3,k}\delta h_1^1} + Z^{\mathrm{d}h_0^{3,k}} \mathbb{E}\left[\frac{\partial Z^{\delta h_1^1}}{\partial \hat{Z}^{(W_0^{3,k})^T \mathrm{d}h_0^{3,k}}}\right]. \tag{299}$$

Furthermore, the random variables $(Z_1^{h_0^1}, Z_1^{h_1^1}) \overset{d}{=} (Z^{h_0^1}, Z^{h_1^1})$ is an iid copy and is independent from $Z^{\mathrm{d}h^{3,k}}$.

We write the limit of each vector in the program as

$$Z^{h_t} = Z^{h_0} + \mathring{\theta}_{\delta h_1} Z^{\delta h_1} + \ldots + \mathring{\theta}_{\delta h_t} Z^{\delta h_t}. \tag{300}$$

Weight updates: The distribution of certain weight updates are required in the backward pass and thus the forward passes.

$$Z^{\overline{\delta W_t^4}} = -\eta Q_t^4 \left(\mathring{\chi}_{t-1} Z^{h_{t-1}^3}\right), \tag{301}$$

$$Z^{(\overline{\delta W_t^2})_{k,:}} = -\eta Q_t^2 \left(\sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathring{h}_{t-1,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_{t-1})_\beta Z^{h_{t-1}^1}\right). \tag{302}$$

tth Backward Pass:

$$Z^{\mathrm{d}h_t^3} = Z^{(\overline{W_t^4})^T}, \tag{303}$$

$$\mathrm{d}\mathring{\tilde{h}}_{t,k}^2 = \mathbb{E}[Z^{h_t^{3,k}} Z^{\mathrm{d}h_t^3}], \tag{304}$$

$$\mathrm{d}\mathring{h}_{t,k}^2 = \sum_{j=1}^m J(\mathring{h}_t^2)_{kj} \mathrm{d}\mathring{\tilde{h}}_{t,j}^2, \tag{305}$$

$$Z^{\mathrm{d}h_t^{3,k}} = \mathring{\tilde{h}}_{t,k}^2 Z^{\mathrm{d}h_t^3}, \tag{306}$$

$$Z^{\mathrm{d}h_t^1} = \sum_{k=1}^m \left(Z^{(\overline{W_t^2})_{k,:}} \mathrm{d}\mathring{h}_{t,k}^2 + Z^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} - \eta \sum_{s=0}^{t-1} \mathbb{E}_{\left(Z_1^{\mathrm{d}h_s^{3,k}}, Z_1^{\mathrm{d}h_t^{3,k}}\right)} \left[Q_{s+1}^3 \left(Z^{h_s^1} \mathring{\chi}_s Z_1^{\mathrm{d}h_s^{3,k}}\right) Z_1^{\mathrm{d}h_t^{3,k}}\right]\right), \tag{307}$$

where $Z^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}}$ is given by the Z MatMul rule,

$$Z^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} = \hat{Z}^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} + \dot{Z}^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} \tag{308}$$

$$= \hat{Z}^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} + \sum_{v \in \mathcal{V}: W_0^{3,k} v \in \mathcal{V}} Z^v \mathbb{E}\left[\frac{\partial Z^{\mathrm{d}h_t^{3,k}}}{\partial \hat{Z}^{W_0^{3,k} v}}\right], \tag{309}$$

where $\mathcal{V} := \{v \text{ for some vector in the program } v\}$ is the set of all vectors in the program. Further the random variables $(Z_1^{\mathrm{d}h_s^{3,k}}, Z_1^{\mathrm{d}h_t^{3,k}}) \overset{d}{=} (Z^{\mathrm{d}h_s^{3,k}}, Z^{\mathrm{d}h_t^{3,k}})$ is an iid copy and is independent of $Z^{h_s^1}$.

tth Forward Pass:

$$Z^{\delta h_t^1} = -\eta Q_t^1 \left(\sum_{\beta=1}^{d_{\text{out}}} Z^{(\mathrm{d}h_{t-1}^1)_\beta} (\mathring{\boldsymbol{\chi}}_{t-1})_\beta \boldsymbol{\xi}^T \right) \boldsymbol{\xi}, \tag{310}$$

$$\delta \mathring{h}_{t,k}^2 = \mathring{\theta}'_{W_{t-1}^2} \mathring{\theta}_{\delta h_t^1} \mathbb{E}[Z^{(\overline{W_{t-1}^2})_{k,:}} Z^{\delta h_t^1}] - \mathring{\theta}'_{\delta W_t^2} \eta \mathbb{E}\left[Q_t^2 \left(\sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathring{h}_{t-1,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_{t-1})_\beta Z^{h_{t-1}^1} \right) Z^{h_t^1} \right], \tag{311}$$

$$\delta \mathring{\tilde{\mathbf{h}}}_t^2 = \begin{cases} \mathrm{G}(\mathring{\mathbf{h}}_t^2) - \mathrm{G}(\mathring{\mathbf{h}}_{t-1}^2) \text{ if } \mathring{\theta}_{\delta h_t^2} = 1 \\ \mathrm{G}'(\mathring{\mathbf{h}}_{t-1}^2)(\delta \mathring{\mathbf{h}}_t^2) \text{ if } \mathring{\theta}_{\delta h_t^2} = 0 \end{cases}, \tag{312}$$

$$Z^{\delta h_t^{3,k}} = \mathring{\theta}_{\delta h_t^1 / \delta h_t^{3,k}} Z^{W_0^{3,k} \delta h_t^1} \tag{313}$$

$$- \sum_{s=0}^{t-2} \mathring{\theta}'_{\delta W_{s+1}^{3,k} \delta h_t^1 / \delta h_t^{3,k}} \eta \mathbb{E}_{\left(Z_1^{h_s^1}, Z_1^{\delta h_t^1}\right)} \left[Q_{s+1}^3 \left(\sum_{\beta}^{d_{\text{out}}} Z^{(\mathrm{d}h_s^{3,k})_\beta} (\mathring{\boldsymbol{\chi}}_s)_\beta Z_1^{h_s^1} \right) Z_1^{\delta h_t^1} \right] \tag{314}$$

$$- \mathring{\theta}'_{\delta W_t^{3,k} / \delta h_t^{3,k}} \eta \mathbb{E}_{\left(Z_1^{h_{t-1}^1}, Z_1^{h_t^1}\right)} \left[Q_t^3 \left(\sum_{\beta}^{d_{\text{out}}} Z^{(\mathrm{d}h_{t-1}^{3,k})_\beta} (\mathring{\boldsymbol{\chi}}_{t-1})_\beta Z_1^{h_{t-1}^1} \right) Z_1^{h_t^1} \right], \tag{315}$$

$$Z^{\delta h_t^3} = \sum_{k=1}^{m} \mathring{\theta}_{\delta h_t^{3,k} / \delta h_t^3} \mathring{\tilde{h}}_{k,t-1}^2 Z^{\delta h_t^{3,k}} + \mathring{\theta}_{\delta \tilde{h}_t^2 / \delta h_t^3} \delta \mathring{\tilde{h}}_{t,k}^2 Z^{h_t^{3,k}}, \tag{316}$$

$$\delta \mathring{f}_t = \mathring{\theta}'_{W_{t-1}^4} \mathring{\theta}_{\delta h_t^3} \mathbb{E}[Z^{\overline{W_{t-1}^4}} Z^{\delta h_t^3}] - \mathring{\theta}'_{\delta W_t^4} \eta \mathbb{E}\left[Q_t^4 \left(\mathring{\chi}_{t-1} Z^{h_{t-1}^3} \right) Z^{h_t^3} \right], \tag{317}$$

where

$$Z^{W_0^{3,k} \delta h_t^1} = \hat{Z}^{W_0^{3,k} \delta h_t^1} + \dot{Z}^{W_0^{3,k} \delta h_t^1} \tag{318}$$

$$= \hat{Z}^{W_0^{3,k} \delta h_t^1} + \sum_{s=0}^{t-1} Z^{\mathrm{d}h_s^{3,k}} \mathbb{E}\left[\frac{\partial Z^{\delta h_t^1}}{\partial \hat{Z}^{(W_0^{3,k})^T \mathrm{d}h_s^{3,k}}} \right] \tag{319}$$

B Derivation of Soft MoE Results

We derive the results from Section 4 here. The results rely on the Tensor Program derivation in Appendix A.

B.1 SP: Router Collapse

In this section, we derive the results for the soft MoE network with SP parametrisation. We use the Tensor Program derivations in Appendix A.1 and Appendix A.3 to do this.

Recall that the SP parametrisation refers to initialisation parameters $b_1 = 0$ and $b_\ell = 0.5$ for all layers $\ell \neq 1$. Furthermore, the learning rate and epsilon scaling parameters are the same for each layer $c = c_\ell$ and $d = d_\ell$ for all layers ℓ. The initial values of b_ℓ set the scalars for the network in Definition 2.1

$$\theta_{W_0^1} = 1, \tag{320}$$

$$\theta_{W_0^2} = n^{-0.5}, \tag{321}$$

$$\theta_{W_0^{3,k}} = 1, \tag{322}$$

$$\theta_{W_0^4} = n^{-0.5}. \tag{323}$$

$$\tag{324}$$

We write the Tensor Program for the first forward pass with SP as follows (the rest of the tensor program is similar to Appendix A.1). We write thie without the bias as it does not affect the result.

First Forward Pass:

$$\mathbf{h}_0^1 = \overline{\mathbf{W}_0^1}\boldsymbol{\xi}, \qquad (\mathbf{h}_0^1)_\alpha = \phi\left((\overline{\mathbf{W}_0^1})_{\alpha 1}, \ldots, (\overline{\mathbf{W}_0^1})_{\alpha d_{\text{in}}}; \xi_1, \ldots, \xi_{d_{\text{in}}}\right) = \sum_{j=1}^{d_{\text{in}}} (\overline{\mathbf{W}_0^1})_{\alpha j}\xi_j, \qquad \texttt{(Nonlin)}$$

$$\mathbf{h}_0^2 = \sqrt{n}\overline{\mathbf{W}_0^2}\mathbf{h}_0^1, \qquad (\mathbf{h}_0^2)_k = \frac{1}{n}\sum_{\alpha=1}^{n} \phi\left((\overline{\mathbf{W}_0^2})_{k\alpha}, (\mathbf{h}_0^1)_\alpha; \sqrt{n}\right) = \frac{1}{n}\sqrt{n}\sum_{\alpha=1}^{n} (\overline{\mathbf{W}_0^2})_{k\alpha}(\mathbf{h}_0^1)_\alpha, \qquad \texttt{(Moment)}$$

$$\tilde{\mathbf{h}}_0^2 = \mathrm{G}(\mathbf{h}_0^2), \qquad (\texttt{Moment, See E})$$

$$\hat{\mathbf{h}}_0^{3,k} = \overline{\mathbf{W}^{3,k}}\mathbf{h}_0^1, \qquad \texttt{(MatMul)}$$

$$\mathbf{h}_0^{3,k} = \hat{\mathbf{h}}_0^{3,k}, \qquad (\mathbf{h}_0^{3,k})_\alpha = \phi\left((\hat{\mathbf{h}}_0^{3,k})_\alpha; -\right) = (\hat{\mathbf{h}}_0^3)_\alpha \qquad \texttt{(Nonlin)}$$

$$\mathbf{h}_0^3 = \sum_{k=1}^{m} \mathbf{h}_0^{3,k}\tilde{h}_{0,k}^2, \qquad (\mathbf{h}_0^3)_\alpha = \phi\left((\mathbf{h}_0^{3,1})_\alpha, \ldots, (\mathbf{h}_0^{3,m})_\alpha; \tilde{h}_{0,1}^2, \ldots, \tilde{h}_{0,m}^2\right) = \sum_{k=1}^{m} (\mathbf{h}_0^{3,k})_\alpha \tilde{h}_{0,k}^2 \qquad \texttt{(Nonlin)}$$

$$\mathbf{f}_0 = \sqrt{n}\frac{1}{n}\overline{\mathbf{W}_0^4}\mathbf{h}_0^3, \qquad (\mathbf{f}_0)_\beta = \frac{1}{n}\sum_{\alpha=1}^{n} \phi\left((\overline{\mathbf{W}_0^4})_{\beta\alpha}, (\mathbf{h}_0^3)_\alpha; \sqrt{n}\right) = \frac{1}{n}\sqrt{n}\sum_{\alpha=1}^{n} (\overline{\mathbf{W}_0^4})_{\beta\alpha}(\mathbf{h}_0^3)_\alpha., \qquad \texttt{(Moment)}$$

where $G(\cdot)$ is the gating function, either Softmax or Sigmoid..

To find the scale of the output-like layers $\mathbf{h}^2$ and $\mathbf{f}$ we use Master's theorem on the variance of the coordinates

$$\mathbb{E}\left[(\mathbf{h}_0^2)_k^2\right] = \sum_{\alpha=1}^{n} \frac{1}{n}\mathbb{E}[(\overline{\mathbf{W}_0^2})_{k\alpha}^2]\mathbb{E}[(\mathbf{h}_0^1)_\alpha^2] = \mathbb{E}\left[\frac{1}{n}\sum_{\alpha=1}^{n}(\mathbf{h}_0^1)_\alpha^2\right], \qquad \texttt{(Moment)} \tag{325}$$

$$\mathbb{E}\left[(\mathbf{f}_0)_\beta^2\right] = \sum_{\alpha=1}^{n} \frac{1}{n}\mathbb{E}[(\overline{\mathbf{W}_0^4})_{\beta\alpha}^2]\mathbb{E}[(\mathbf{h}_0^3)_\alpha^2] = \mathbb{E}\left[\frac{1}{n}\sum_{\alpha=1}^{n}(\mathbf{h}_0^3)_\alpha^2\right]. \qquad \texttt{(Moment)} \tag{326}$$

We consider SGD updates here for ease of exposition. The result trivially holds for Adam updates as well (see Remark B.3).

Using the Master's theorem [41], the coordinates of each layer will have scale $\Theta(1)$,

$$\theta_{h_0^1} = 1, \tag{327}$$

$$\theta_{h_0^2} = 1, \tag{328}$$

$$\theta_{\tilde{h}_0^2} = 1, \tag{329}$$

$$\theta_{h_0^{3,k}} = 1, \tag{330}$$

$$\theta_{h_0^3} = 1, \tag{331}$$

$$\theta_{f_0} = 1. \tag{332}$$

Without loss of generality, we assume that values of c is chosen such that $\delta\mathbf{h}_1^1$ and $\mathbf{h}_1^1 = \mathbf{h}_0^1 + \delta\mathbf{h}_1^1$ are $\Theta(1)$. That is $\theta_{\delta h_1^1} = \theta_{h_1^1} = 1$. It is safe to make this assumption as $\delta\mathbf{h}_1^1$ exploding or vanishing with width are both undesirable. The update to $\mathbf{h}^2$ then has the form (Appendix A.1)

$$\delta\mathbf{h}_1^2 = \theta'_{W_0^2}\frac{\overline{\mathbf{W}_0^2}\delta\mathbf{h}_1^1}{n} - \theta'_{\delta W_1^2}\eta \mathrm{d}\mathbf{h}_0^2\boldsymbol{\chi}_0\frac{(\mathbf{h}_0^1)^T\mathbf{h}_1^1}{n}, \tag{333}$$

where $\overline{\mathbf{W}_0^2}, \delta\mathbf{h}_1^1$ are the unscaled ($\Theta(1)$) terms. The above operation is written as a Moment in the Tensor Program. By the TP Master Theorem, $\frac{\overline{\mathbf{W}_0^2}\delta\mathbf{h}_1^1}{n} \to \mathbb{E}[Z^{\overline{W_0^2}}Z^{\delta h_1^1}]$ (Appendix A.1). As the gradients required to calculate δh_1^1 depend on $\overline{W_0^2}$, the limiting random variables $Z^{\overline{W_0^2}}, Z^{\delta h_1^1}$ are correlated, thus the expectation is non-zero and the term scales $\Theta(1)$. Thus, the first term has the scale $\theta'_{W_0^2} = n^{0.5}$, and the second scales $\theta'_{\delta W_0^2} = n^{1-c} = n^{1-c}$. Hence, the first term scales $\Theta(n^{0.5})$ irrespective of the scaling of the second term, which means that the entire term scales $\Omega(n^{0.5})$. As $\mathbf{h}_1^2 = \delta\mathbf{h}_1^2 + \mathbf{h}_0^2$, where the first term scales $\Omega(n^{0.5})$, and the second term scales $\Theta(1)$, hence $\mathbf{h}_1^2$ scales $\Omega(n^{0.5})$.

Saturation of Softmax $\tilde{\mathbf{h}}^2$: In the softmax network, $\tilde{\mathbf{h}}_1^2 = \mathrm{Softmax}(\mathbf{h}_1^2)$ where $\mathbf{h}_1^2$ scales $\Omega(n^{0.5})$. Using the well known property that softmax of diverging inputs results in a one-hot distribution, we get,

$$\lim_{n\to\infty} \frac{e^{(h_t^2)_k}}{\sum_{j=1}^m e^{(h_t^2)_j}} \to 1,$$

for some expert index $k \in [m]$. The gradient of the softmax layer is

$$\frac{\partial(\tilde{\mathbf{h}}_1^2)_i}{\partial(\mathbf{h}_1^2)_j} = (\tilde{h}_1^2)_i(\delta_{ij} - (\tilde{h}_1^2)_j),$$

where $(\tilde{\mathbf{h}}_1^2)_i$ is the ith component of the vector $\tilde{\mathbf{h}}_1^2$, and δ_{ij} is the Kronecker delta function. If $\tilde{\mathbf{h}}_1^2$ is one-hot, then the gradient $\frac{\partial\tilde{\mathbf{h}}_1^2}{\partial\mathbf{h}_1^2}$ is a zero matrix of size $m \times m$. As $\frac{\partial\tilde{\mathbf{h}}_1^2}{\partial\mathbf{h}_1^2}$ is required to calculate the gradient of the router $\frac{\partial\mathcal{L}}{\partial\mathbf{W}^2}$,

$$\frac{\partial\mathcal{L}}{\partial\mathbf{W}_1^2} = \frac{\partial\mathcal{L}}{\partial\tilde{\mathbf{h}}_1^2}\frac{\partial\tilde{\mathbf{h}}_1^2}{\partial\mathbf{h}_1^2}\frac{\partial\mathbf{h}_1^2}{\partial\mathbf{W}_1^2}$$

this implies that the router gradient will approach a matrix of zeroes as the width is increased. Hence $\mathbf{W}_t^2 = \mathbf{W}_1^2$ for all subsequent steps of training.

Saturation of Sigmoid $\tilde{\mathbf{h}}^2$: In the sigmoid network, $\tilde{\mathbf{h}}_1^2 = \sigma(\mathbf{h}_1^2)$ where $\mathbf{h}_1^2$ scales $\Omega(n^{0.5})$. With n increasing we get

$$\lim_{n\to\infty} \sigma((h_t^2)_k) \to \begin{cases} 1 & \text{if } (h_t^2)_k > 0 \\ 0 & \text{if } (h_t^2)_k < 0 \end{cases},$$

for every expert $k \in [m]$. The gradient of the sigmoid layer is

$$\frac{\partial(\tilde{\mathbf{h}}_1^2)_i}{\partial(\mathbf{h}_1^2)_i} = \sigma\left((\mathbf{h}_1^2)_i\right)\left(1 - \sigma((\mathbf{h}_1^2)_i)\right),$$

which approaches a zero matrix as n increases. Similarly to the softmax case, this results in a vanishing gradient for the router.

Remark B.1 (Zero initialisation of router with SGD). Zero intitialisation of the router with SGD ensures that the interaction between $\mathbf{W}_0^2$ and $\delta\mathbf{h}_1^1$ does not occur. However, here the results from Yang and Hu [41] hold, primarily that it is not possible for the "output-like" layers (router and function) to feature learn. This can easily be seen by the fact that $\theta_{\delta h_1^1} = n^{-(c+1)}$, which means that $c = -1$ is required for $\mathbf{h}_1^1$ to feature learn. This however causes the second term (effective updates) in $\delta\mathbf{h}_1^2$ to diverge as $\theta'_{\delta W_1^2} = n \cdot n^{-c} \cdot n \cdot \theta_{W_0^4} = n^3$.

Remark B.2 (Results hold for Top-K). The router logits diverge due to the interaction between $\mathbf{W}_0^2$ and $\delta\mathbf{h}_1^1$, and thus holds irrespective of the gating function, or sparse execution. If Softmax or Sigmoid activations are used with Top-K, the router probabilities will still saturate with SP.

Remark B.3. (Router diverges with Adam) The analysis above relies on the propagating updates $\mathbf{W}_0^2\delta\mathbf{h}_1^1$ of the router blowing up. Adam updates only affect the effective updates of the router $\delta\mathbf{W}_1^2\mathbf{h}_1^2$. Hence, the same analysis holds for Adam.

B.2 μP-MoE: Stability but with Router Stagnation without Router Bias

Table 1: *bcd* parametrisation vlaue for μP-MoE SGD

Layer (ℓ)	$\mathbf{b}_\ell$	$\mathbf{c}_\ell$
1	0	-1
2	1	1
3	0.5	0
4	1	1

Stability and Feature Learning for SGD: We show that the values in Table 1 for the network in Definition 2.1 satisfy our desired properties of stability and feature learning for SGD. The values in Setting A.3 already satisfy stability. To show feature learning, we show that the limiting values of the scalars $\mathring{\theta}_\bullet$ in the feature updates converge to one. Stability ($\mathcal{O}(1)$ features) combined with $\Theta(1)$ feature updates then imply that all the features scale $\Theta(1)$.

$$\mathring{\theta}_{\delta h_1^1} = 1, \qquad \mathring{\theta}'_{\delta W_1^2} = 1, \tag{334}$$

$$\mathring{\theta}_{\delta h_1^2} = 1, \tag{335}$$

$$\mathring{\theta}_{\delta h_1^1/\delta h_1^{3,k}} = 1, \qquad \mathring{\theta}'_{\delta W_1^3/\delta h_1^{3,k}} = 1, \tag{336}$$

$$\mathring{\theta}_{\delta h_1^{3,k}/\delta h_1^3} = 1, \qquad \mathring{\theta}_{\delta \tilde{h}_1^2/\delta h_1^3} = 1, \tag{337}$$

$$\mathring{\theta}_{\delta h_1^3} = 1, \qquad \mathring{\theta}'_{\delta W_1^4} = 1. \tag{338}$$

For the next forward passes we have $\theta_{W_t^2} = \theta_{W_0^2}$ and $\theta_{W_t^4} = \theta_{W_0^4}$, thus we have

$$\mathring{\theta}'_{W_{t-1}^2}\mathring{\theta}_{\delta h_t^1} = 1, \qquad \mathring{\theta}'_{\delta W_t^2} = 1, \tag{339}$$

$$\mathring{\theta}_{\delta h_t^2} = 1, \tag{340}$$

$$\mathring{\theta}_{\delta h_t^1/\delta h_t^{3,k}} = 1, \qquad \mathring{\theta}'_{\delta W_s^3 \delta h_t^1/\delta h_t^{3,k}} = 1 \qquad \mathring{\theta}'_{W_t^3/\delta h_t^{3,k}} = 1, \tag{341}$$

$$\mathring{\theta}_{\delta h_t^{3,k}/\delta h_t^3} = 1, \qquad \mathring{\theta}_{\delta \tilde{h}_t^2/\delta h_t^3} = 1, \tag{342}$$

$$\mathring{\theta}'_{W_{t-1}^4}\mathring{\theta}_{\delta h_t^3} = 1, \qquad \mathring{\theta}'_{\delta W_t^4} = 1. \tag{343}$$

For a formal arguement to show feature learning (such that the limits are not degenerate), we refer to [41, H.7].

Table 2: *bcd* parametrisation vlaue for μP-MoE Adam

Layer (ℓ)	$\mathbf{b}_\ell$	$\mathbf{c}_\ell$	$\mathbf{d}_\ell$
1	0	0	1
2	1	1	0
3	0.5	1	1
4	1	1	0

Stability, Feature Learning, and Faithfulness for Adam: The faithfulness follows from the values at the first weight updates Assumption A.7. Similar to above, it is easy to see that all $\mathring{\theta}_\bullet$ converge to one.

Lack of specialisation: We now show that for μP-MoE without router bias, specialisation of the experts fails. We first show that, in the infinite-width limit, the gradient received by each router logit converges to the same deterministic value. Next, we show that this implies that the router logits, and hence the probabilities evolve synchronously, keeping the same value for each expert. In Softmax-Soft MoEs, due to the form of the softmax gradient, the above results imply that the gradient received by the router is zero. We finish with remarks that these theorems hold for Adam, and we find the finite-width scaling of the router gradient in Softmax-Soft MoEs.

Proposition B.4. *Consider a soft MoE network without a router bias with m experts with either Softmax or Sigmoid gating parametrised such that stability holds throughout training (Setting A.1), and such that the router logits converge to zero in the infinite width limit, $\mathring{\mathbf{h}}_0^2 = 0$ (Setting A.3, Table 1). For any time step $t \geq 0$, each router activation $\tilde{\mathbf{h}}^2$ component receives the same deterministic unscaled ($\Theta(1)$) gradient. That is,*

$$\mathrm{d}\mathring{\tilde{h}}_{t,k}^2 := \mathbb{E}\left[Z^{h_t^{3,k}} Z^{\mathrm{d}h_t^3}\right], \tag{344}$$

is identical for all experts $k \in \{1, \ldots, m\}$.

Proof. First, observe that $Z^{\mathrm{d}h_t^3} = Z^{\overline{W_t^4}}$ is a global variable shared by the entire network; it depends only on the output layer weights, making it independent of the specific expert index k. Therefore, it suffices to prove that the marginal distribution of the expert features $Z^{h_t^{3,k}}$ is identical for all k. We proceed by induction.

Base Case ($t = 0$): At initialization, the expert features are defined by the random variable:

$$Z^{h_0^{3,k}} = Z^{W_0^{3,k} h_0^1}. \tag{345}$$

The input features $\mathbf{h}_0^1$ are shared across all experts. The weights $\mathbf{W}_0^{3,k}$ are initialized i.i.d. such that $(\hat{W}_k^3)_{ij} \sim \mathcal{N}(0, n^{-1})$. Thus $Z^{W_0^{3,k} h_0^1}$ (derived by ZMatMul rule) is the same for every expert. Consequently, the random variables $Z^{h_0^{3,k}}$ are identically distributed for all k:

$$Z^{h_0^{3,k}} \overset{d}{=} Z^{h_0^{3,j}} \quad \forall k, j. \tag{346}$$

Thus, the expectation $\mathbb{E}[Z^{h_0^{3,k}} Z^{\mathrm{d}h_0^3}]$ is constant with respect to k , and thus $\mathrm{d}\mathring{\tilde{h}}_{0,k}^2$ is the same for each expert.

Note that for softmax or sigmoid gating, the router probabilities are the same for each expert at initialisation due to $\mathring{h}_{0,k}^2 = 0$ and without a router bias,

$$\tilde{h}_{0,k}^2 = \begin{cases} \frac{1}{m}\mathbf{1} & \text{if G} = \text{Softmax} \\ \frac{1}{2}\mathbf{1} & \text{if G} = \text{Sigmoid} \end{cases}.$$

Hence $\tilde{h}_{0,i}^2 = \tilde{h}_{0,j}^2$ for every $i, j \in [m]$.

Inductive Step: Assume that for all time steps $s < t$, the expert features are identically distributed across experts:

$$Z^{h_s^{3,k}} \overset{d}{=} Z^{h_s^{3,j}} \quad \Longrightarrow \quad \mathrm{d}\mathring{\tilde{h}}_{s,k}^2 = \mathring{\tilde{h}}_{s,j}^2 \quad \forall k, j. \tag{347}$$

The router logits at a given time step $s < t$ are

$$\mathring{h}_{s,k}^2 = \mathring{h}_{0,k}^2 + \sum_{p=1}^{s} \delta\mathring{h}_{p,k}^2,$$

where $\mathring{h}_{0,k}^2 = \mathring{h}_{0,j}^2$ for any experts k, j due to the initialisation assumption. We now show that the updates $\delta\mathring{h}_{p,k}^2$ are the same for each expert index k. The infinite width limit router weight updates and corresponding logit updates for each expert is

$$Z_{k,:}^{(\delta W_p^2)} = -\eta \sum_{\beta=1}^{d_{\text{out}}} (\mathring{\tilde{h}}_{p,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_{p-1})_\beta Z^{h_{p-1}^1}, \tag{348}$$

$$\delta\mathring{h}_{p,k}^2 = \mathring{\theta}_{W_{p-1}^2} \mathring{\theta}_{\delta h_p^1} \mathbb{E}[Z^{(\overline{W_{p-1}^2})_{k,:}} Z^{\delta h_p^1}] - \mathring{\theta}'_{\delta W_p^2} \eta \sum_{\beta=1}^{d_{\text{out}}} (\mathring{\tilde{h}}_{p-1,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_{p-1})_\beta \mathbb{E}[Z^{h_{p-1}^1} Z^{h_p^1}]. \tag{349}$$

As the inductive hypothesis implies $\mathring{\tilde{h}}_{p,k}^2 = \mathring{\tilde{h}}_{p,j}^2 \forall k, j$, we see that updates are the same for each expert a every time step $p < s < t$ and hence that

$$\mathring{h}_{s,k}^2 = \mathring{h}_{s,j}^2 \quad \forall k, j.$$

As the gradient for each expert k at time step s is

$$Z^{\mathrm{d}h_s^{3,k}} = \mathring{\tilde{h}}_{s,k}^2 Z^{\mathrm{d}h_s^3},$$

where $Z^{\mathrm{d}h_s^3}$ is shared for all the experts. Using the fact that $\mathring{\tilde{h}}_{s,k}^2 = \mathring{\tilde{h}}_{s,j}^2$ for all experts k, j, we get

$$Z^{\mathrm{d}h_s^{3,k}} \overset{d}{=} Z^{\mathrm{d}h_s^{3,j}} \quad \forall k, j.$$

Consider the update at step t, defined in the infinite width limit as:

$$\mathbf{h}_t^{3,k} = \mathbf{h}_{t-1}^{3,k} + \delta\mathbf{h}_t^{3,k}, \tag{350}$$

$$\begin{aligned} Z^{\delta h_t^{3,k}} &= \mathring{\theta}_{\delta h_t^1/\delta h_t^{3,k}} Z^{W_0^{3,k}\delta h_t^1} - \sum_{s=0}^{t-2} \mathring{\theta}'_{\delta W_{s+1}^{3,k}\delta h_t^1/\delta h_t^{3,k}\eta} \sum_{\beta=1}^{d_{\text{out}}} Z_\beta^{(\mathrm{d}h_s^{3,k})}(\mathring{\boldsymbol{\chi}}_s)_\beta \mathbb{E}[Z^{h_s^1} Z^{\delta h_t^1}] \\ &\quad - \mathring{\theta}'_{\delta W_t^{3,k}/\delta h_t^{3,k}\eta} \sum_{\beta=1}^{d_{\text{out}}} Z^{(\mathrm{d}h_{t-1}^{3,k})_\beta}(\mathring{\boldsymbol{\chi}}_{t-1})_\beta \mathbb{E}[Z^{h_{t-1}^1} Z^{h_t^1}], \end{aligned} \tag{351}$$

Note that $\mathring{\theta}'_{\delta W_t^{3,k}}$ is the same for each expert. As we have shown, inductive hypothesis implies that $Z^{\mathrm{d}h_s^{3,k}} \overset{d}{=} Z^{\mathrm{d}h_s^{3,j}}$, for all experts k, j at every time step $s < t$. Furthermore, $Z^{W_0^{3,k}\delta h_t^1}$ (from the ZMatMul rule) is the same for each expert as $Z^{W_0^{3,k}}$ is the same for each expert and $Z^{\delta h_t^1}$ is shared by all the experts. Thus, using the inductive hypothesis that $Z^{h_{t-1}^{3,k}} \overset{d}{=} Z^{h_{t-1}^{3,j}}$ we have,

$$Z^{\delta h_t^{3,k}} \overset{d}{=} Z^{\delta h_t^{3,j}} \quad \implies Z^{h_t^{3,k}} \overset{d}{=} Z^{h_t^{3,j}} \quad \forall k, j. \tag{352}$$

This is turn implies that

$$\mathrm{d}\mathring{\tilde{h}}_{t,k}^2 = \mathrm{d}\mathring{\tilde{h}}_{t,j}^2 \quad \forall k, j. \tag{353}$$

□

Corollary B.5. *Consider a soft MoE network without a router bias with m experts with either softmax or sigmoid gating parametrised such that stability holds throughout training (Setting A.1), and such that the router logits converge to zero in the infinite width limit, $\mathring{\mathbf{h}}_0^2 = 0$ (Setting A.3, Table 1). For any time step $t \geq 0$, the router probability is the same for every expert in the infinite width limit. That is,*

$$\mathring{\tilde{h}}_{t,i}^2 = \mathring{\tilde{h}}_{t,j}^2 \tag{354}$$

for every expert $i, j \in [m]$.

Proof. Result follows from the inductive proof in Proposition B.4. For the last step of the induction, the result of Proposition B.4, which implies that the gradient of the router logits are the same for each expert index $\mathrm{d}\mathring{h}_{t,k}^2 = \mathrm{d}\mathring{h}_{t,j}^2$. This, combined with the initialisation $\mathring{\mathbf{h}}_0^2 = 0$ implies that $\mathring{h}_{t,i}^2 = \mathring{h}_{t,j}^2$ for every expert $i, j \in [m]$ and time step t. The result for $\mathring{\tilde{h}}_{t,i}^2 = \mathring{\tilde{h}}_{t,j}^2$ for every expert $i, j \in [m]$ follows. □

Corollary B.6. *Consider a Softmax-Soft MoE netwok without a router bias. For any time step $t \geq 0$, in the infinite width limit, the gradient with respect to the router vanishes,*

$$\mathrm{d}\mathring{h}_{t,k}^2 = 0, \quad \textit{for every expert } k. \tag{355}$$

Consequently, the router weights do not evolve from initialisation,

$$Z^{(\delta W_t^2)_{k,:}} = 0, \tag{356}$$

$$\implies \mathring{\tilde{h}}_{t,k}^2 = \mathbb{E}[Z^{(\overline{W_0^2})_{k,:}} Z^{\delta h_t^1}], \tag{357}$$

for every k. That is, in the limit, the router is stuck at initialisation.

Proof. The limiting value of the gradient with respect to the router is

$$\mathrm{d}\mathring{h}_{t,k}^2 = \sum_{j=1}^{m} J(\mathring{h}_t^2)_{kj} \mathrm{d}\mathring{\tilde{h}}_{t,j}^2, \tag{358}$$

where $J(h^2)_{kj} = \tilde{h}^2_k(\delta_{kj} - \tilde{h}^2_j)$ is the Jacobian of the Softmax and $\tilde{h}^2 = \mathrm{Softmax}(h^2)$. From Proposition B.4, $\mathrm{d}\mathring{\tilde{h}}^2_{t,j} = \mathrm{d}\mathring{\tilde{h}}^2_{t,k}$ for all experts k, j. Let $C := \mathrm{d}\mathring{\tilde{h}}^2_{t,j}$. For every k, we have

$$\mathrm{d}\mathring{h}^2_{t,k} = \sum_{j=1}^{m} J(\mathring{h}^2_t)_{kj} C \tag{359}$$

$$= C\mathring{\tilde{h}}^2_{t,k} \sum_{j=1}^{m} \left(\delta_{kj} - \mathring{\tilde{h}}^2_{t,j}\right) \tag{360}$$

$$= 0, \tag{361}$$

where we use the fact that $\sum_{j=1}^{m} \mathring{\tilde{h}}^2_{t,j} = 1$. The rest follows from looking at the infinite width limits of $\delta \mathbf{W}^2_t$ and $\delta \mathbf{h}^2_{t,k}$. □

Remark B.7 (Router bias prevents the stagnation in Corollary B.5). The reason for the router stagnation is the symmetric initialisation due to μP-MoE and the resulting identical deterministic gradients for each expert. A router bias (or noise) ensures that at initialisation the router is not symmetric, hence leading to varied deterministic gradient for each expert. The router thus does not stagnate.

Remark B.8 (Results hold for any stable parametrisation). For the above results, we only assume stability and the results are shown using unscaled quantities (quantities that scale $\Theta(1)$). As such, any values of *bcd* do not change our results. We also assume $\mathring{\tilde{\mathbf{h}}}^2_0 = 0$ — the only stable parametrisation where this is not true is with $b_2 = 0.5$. We refer to [41, H.4.1] for discussion that. Furthermore, $b_2 = 1$ is required for feature learning considerations (Setting A.3).

Remark B.9. (Extension to Adam) The above theorems hold trivially for Adam. The only change required is in the proof for Proposition B.4. Here, the infinite width limit update to $\mathbf{h}^{3,k}_t$ will be (Appendix A.4)

$$Z^{\delta h^{3,k}_t} = \mathring{\theta}_{\delta h^1_t / \delta h^{3,k}_t} Z^{W^{3,k}_0 \delta h^1_t} \tag{362}$$

$$- \sum_{s=0}^{t-2} \mathring{\theta}'_{\delta W^{3,k}_{s+1} \delta h^1_t / \delta h^{3,k}_t} \eta \mathbb{E}_{\left(Z^{h^1_s}_1, Z^{\delta h^1_t}_1\right)} \left[Q^3_{s+1} \left(\sum_{\beta}^{d_{\text{out}}} Z^{(\mathrm{d}h^{3,k}_s)_\beta} (\mathring{\boldsymbol{\chi}}_s)_\beta Z^{h^1_s}_1 \right) Z^{\delta h^1_t}_1 \right] \tag{363}$$

$$- \mathring{\theta}'_{\delta W^{3,k}_t / \delta h^{3,k}_t} \eta \mathbb{E}_{\left(Z^{h^1_{t-1}}_1, Z^{h^1_t}_1\right)} \left[Q^3_t \left(\sum_{\beta}^{d_{\text{out}}} Z^{(\mathrm{d}h^{3,k}_{t-1})_\beta} (\mathring{\boldsymbol{\chi}}_{t-1})_\beta Z^{h^1_{t-1}}_1 \right) Z^{h^1_t}_1 \right], \tag{364}$$

where Q is the Adam update function (Definition A.5). Here it is easy to see that $Z^{\mathrm{d}h^{3,k}_s} \overset{d}{=} Z^{\mathrm{d}h^{3,j}_s}$, for all experts k, j at every time step $s < t$ implies the same result as Proposition B.4. The result from Corollary B.6 follows.

Remark B.10 (Scale of $\mathrm{d}h^2_{t,k}$ for finite n). In the tensor program, for each output coordinate β we have

$$(\mathrm{d}\tilde{h}^2_{t,k})_\beta = \frac{1}{n} \sum_{\alpha=1}^{n} \theta'_{W^4_t} \theta_{h^{3,k}_t} (h^{3,k}_t)_\alpha (\mathrm{d}h^3_t)_{\alpha\beta} = \theta'_{W^4_t} \theta_{h^{3,k}_t} \frac{1}{n} \sum_{\alpha=1}^{n} (Y_{t,k})_{\alpha\beta}, \quad \texttt{(Moment)} \tag{365}$$

where we defined

$$(Y_{t,k})_{\alpha\beta} := (h^{3,k}_t)_\alpha (\mathrm{d}h^3_t)_{\alpha\beta}. \tag{366}$$

The gradient with respect to the router logits is

$$(\mathrm{d}h^2_{t,k})_\beta = \sum_{j=1}^{m} J(\tilde{h}^2_t)_{kj} (\mathrm{d}\tilde{h}^2_{t,j})_\beta = \theta'_{W^4_t} \theta_{h^{3,k}_t} \frac{1}{n} \sum_{\alpha=1}^{n} X_{t,k,\alpha\beta}, \quad \texttt{(Moment)} \tag{367}$$

where

$$X_{t,k,\alpha\beta} := \sum_{j=1}^{m} J(\tilde{h}^2_t)_{kj} (Y_{t,j})_{\alpha\beta}. \tag{368}$$

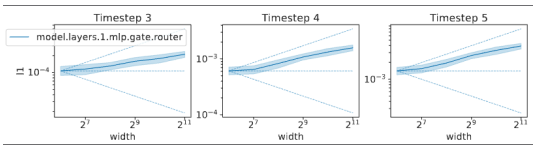


Figure 4: Average norm of router features as width is increased with router $d = 0.5$. Shows that for high ϵ, this causes the router logits to diverge.

In the infinite-width limit, we showed above that $\mathrm{d}\mathring{h}^2_{t,k} = 0$, so in particular $\mathbb{E}[X_{t,k,\alpha\beta}] = 0$. To capture the scale, consider the second-moment "Moment" variable

$$\theta'_{W_t^4}\theta_{h_t^{3,k}}\frac{1}{n}\sum_{\alpha=1}^{n}(X_{t,k,\alpha\beta})^2 = \theta'_{W_t^4}\theta_{h_t^{3,k}}\sum_{j=1}^{m}\sum_{i=1}^{m}J(\tilde{h}^2_t)_{kj}J(\tilde{h}^2_t)_{ki}\frac{1}{n}\sum_{\alpha=1}^{n}(Y_{t,j})_{\alpha\beta}(Y_{t,i})_{\alpha\beta}, \tag{369}$$

whose infinite-width limit, by the Master theorem, is

$$\mathring{\theta}'_{W_t^4}\mathring{\theta}_{h_t^{3,k}}\sum_{j=1}^{m}\sum_{i=1}^{m}J(\mathring{\tilde{h}}^2_t)_{kj}J(\mathring{\tilde{h}}^2_t)_{ki}\mathbb{E}\big[Z^{Y_{t,j}}Z^{Y_{t,i}}\big]. \tag{370}$$

Thus the variance of $(\mathrm{d}h^2_{t,k})_\beta = n^{-1}\sum_{\alpha=1}^{n}X_{t,k,\alpha\beta}$ scales as

$$\mathrm{Var}\big[(\mathrm{d}h^2_{t,k})_\beta\big] = \frac{1}{n}(\theta'_{W_t^4}\theta_{h_t^{3,k}})^2\mathrm{Var}(X_{t,k,\alpha\beta}), \tag{371}$$

where $\mathrm{Var}(X_{t,k,\alpha\beta})$ is some non-zero scalar. Consequently, we have that $\mathrm{d}h^2_{t,k}$ scales $\Theta(n^{-\frac{1}{2}})$.

Remark B.11 (Scale of $\mathbf{h}^2_t - \mathbf{h}^2_{t-1}$ for Softmax-Soft MoE networks under μP-Moe parametrisation). Here, we find the scale of $\mathbf{h}^2_t - \mathbf{h}^2_{t-1}$ for Softmax-Soft MoE networks under the μP-MoE parametrisation. We find the scale under zero-initisalisation of $\mathbf{W}^2_0$, so that the initialisation of the network matches the infinite-width limit, as recommended in Yang et al. [43]. Here, only the effective updates decide the scale of the feature update. The scale of the router weight update is $\theta_{\delta W^2_t} = n^{-1}\cdot n^{-0.5} = n^{-1.5}$, where the extra $n^{-0.5}$ comes from the finite width scale of the router gradient, due to the softmax Jacobian (Remark B.10). Thus, the scale of $\mathbf{h}^2_t - \mathbf{h}^2_{t-1}$ is $\theta'_{\delta W^2_t} = n\cdot n^{-1.5} = n^{-0.5}$.

Remark B.12 (Benign Scaling for Adam). In Softmax-Soft MoEs, the scale of the gradient has an extra $n^{-0.5}$ factor. This implies setting $d = 0.5$ for the router can be used to rescale the variance such that it is $\Theta(1)$. However, as this results in a random scalar in the program, this limit is not expressible in Tensor Programs [41, Theorem 7.4]. In practice we find this can lead to diverging logits Figure 4.

C Proof of Proposition 5.1

From Appendix A.2, we see that at initialisation the router logits converge deterministically to zero under μP-MoE scaling without a bias. A simple CLT argument, (formalised in Yang [39]) shows that $\sqrt{n}(\mathbf{h}^2(n) - C\mathbf{1})$ converges in distribution to a Gaussian with variance proportional to $\mathbb{E}[(Z^{h^1})^2]$. Thus, we can write

$$\mathbf{h}^2(n) = C\mathbf{1} + \sigma_n\epsilon_n,$$

with $\epsilon_n = \sqrt{n}(\mathbf{h}^2(n) - C\mathbf{1})$ and $\sigma_n = \frac{1}{\sqrt{n}}$.

Theorem C.1 (Limit of the Top-K masked vector)**.** *Let*

$$S_n := \mathrm{TopK}(\varepsilon_n) = \mathrm{TopK}\big(\{h^2_i(n)\}_{i=1}^m\big),$$

and define

$$\tilde{h}^2_i(n) := h^2_i(n)\,\mathbf{1}\{i \in S_n\}, \qquad \tilde{\mathbf{h}}^2(n) := (\tilde{\mathbf{h}}^2_1(n), \ldots, \tilde{\mathbf{h}}^2_M(n)).$$

Then,

$$\tilde{\mathbf{h}}^2(n) \overset{d}{\Rightarrow} C\,\mathbf{1}_S \quad \textit{in } \mathbb{R}^M, \textit{ where } S := \mathrm{TopK}(\varepsilon)$$

where 1_S is the indicator function indicating whether index $j \in S$.

Proof. Define the Top-K indicator map

$$T : \mathbb{R}^M \to \{0,1\}^M, \qquad T(z) := \mathbf{1}_{\mathrm{TopK}(z)}.$$

This map is discontinuous only at points where ties occur among the order statistics. As the probability of ties when sampling from ϵ is zero (as Gaussian as a continuous joint density),

$$\mathbb{P}(\varepsilon \in D_T) = 0,$$

where D_T denotes the discontinuity set of T. Hence, by the extended continuous mapping theorem [1],

$$T(\varepsilon_n) \overset{d}{\Rightarrow} T(\varepsilon), \qquad \text{i.e. } \mathbf{1}_{S_n} \overset{d}{\Rightarrow} \mathbf{1}_S.$$

Now write

$$\tilde{\mathbf{h}}^2(n) = \big(C\mathbf{1} + \sigma_n \varepsilon_n\big) \odot T(\varepsilon_n) = C\,T(\varepsilon_n) + \sigma_n\big(\varepsilon_n \odot T(\varepsilon_n)\big).$$

Since $\varepsilon_n \overset{d}{\Rightarrow} \varepsilon$, the sequence $\{\varepsilon_n\}$ is tight, and therefore

$$\|\varepsilon_n\|_\infty = O_p(1).$$

As $\sigma_n \to 0$, it follows that

$$\Big\|\sigma_n\big(\varepsilon_n \odot T(\varepsilon_n)\big)\Big\|_\infty \le \sigma_n \|\varepsilon_n\|_\infty \xrightarrow{p} 0.$$

Combining this with $\mathbf{1}_{S_n} \overset{d}{\Rightarrow} \mathbf{1}_S$ and applying Slutsky's theorem yields

$$\tilde{\mathbf{h}}^2(n) \overset{d}{\Rightarrow} C\,\mathbf{1}_S.$$

□

Thus, in the first forward pass (at initialisation), where the router is symmetric, the Top-K index set is not deterministic as $S := \mathrm{TopK}(\epsilon)$.

D Tensor Program for Top-K MoE

In this section we derive the Tensor program and the corresponding infinite width limit for Top-K MoEs (Definition 2.2). Note we only derive this for SGD, corresponding limit to Adam is analogous to Appendix A.3.

D.1 Tensor Program for Top-K MoE with SGD

We use the same notation as in Appendix A.1. Throughout we make the assumption that the parametrisation is stable (Setting A.1).

Network:

$$\mathbf{h}^1 = \mathbf{W}^1 \boldsymbol{\xi}, \qquad \in \mathbb{R}^{n\times 1} \tag{372}$$

$$\mathbf{h}^2 = \mathbf{W}^2 \mathbf{h}^1 + \mathbf{b}^2, \qquad \in \mathbb{R}^{m\times 1} \tag{373}$$

$$\mathbf{s} = \mathrm{T}(\mathbf{h}^2) \qquad \text{where } \mathrm{T}_k(\mathbf{h}^2) := \mathbf{1}\{k \in \mathrm{TopK}(\mathbf{h}^2)\} \tag{374}$$

$$\tilde{\mathbf{h}}^2 = \mathrm{G}(\mathbf{h}^2, \mathbf{s}), \qquad \in [0,1]^{m\times 1} \tag{375}$$

$$\mathbf{h}^{3,k} = \mathbf{W}^{3,k} \mathbf{h}^1, \qquad \in \mathbb{R}^{n\times 1} \tag{376}$$

$$\mathbf{h}^3 = \sum_{k=1}^{m} \mathbf{h}^{3,k} \tilde{h}^2_k, \qquad \in \mathbb{R}^{n\times 1} \tag{377}$$

$$\mathbf{f} = \mathbf{W}^4 \mathbf{h}^3, \qquad \in \mathbb{R}^{d_{\text{out}}\times 1} \tag{378}$$

where $\mathbf{b}^2 \in \mathbb{R}^m$ is a bias parameter that is represented as an initial scalar in the program, $\mathbf{s}$ is the Top-K mask, and $\mathrm{G}(\cdot, \mathbf{s})$ is the activations applied to the index set $\mathbf{s}$. For Softmax

$$\mathrm{G}_k(u, s) := \frac{s_k \, \exp(u_k)}{\sum_{j=1}^m s_j \, \exp(u_j)}, \tag{379}$$

and for Sigmoid

$$\mathrm{G}_k(u,s) := s_k\,\sigma(u_k).$$

Conditioned on noise/bias: We derive the Tensor program conditioned on the bias term noises $\{\mathbf{b}_t^2\}_{t=0}^T$ or the learnable bias $\mathbf{b}^2$. We assume that they have been sampled from a distribution such that $\mathbb{P}(b_i^2 = b_j^2) = 0$. Practically, we sample $b_i \sim \mathcal{N}(0,1)$.

Remark D.1. Top-K MoEs are not admissible in the Tensor program framework without a router bias. This is because the router history is not deterministic, whereas the Tensor program Master Theorem requires scalars (such as router mask) to converge to deterministic constants.

First Forward Pass:

$$\mathbf{h}_0^1 = \theta_{W_0^1}\overline{\mathbf{W}_0^1}\boldsymbol{\xi}, \qquad (\mathbf{h}_0^1)_\alpha = \phi\left(\{(\overline{\mathbf{W}_0^1})_{\alpha\beta}\}_{\beta=1}^{d_{\text{in}}}; \xi_1,\dots,\xi_{d_{\text{in}}}, \theta_{W_0^1}\right) = \sum_{j=1}^{d_{\text{in}}} \theta_{W_0^1}(\overline{\mathbf{W}_0^1})_{\alpha j}\xi_j,$$

$$\mathbf{h}_0^2 = \theta'_{W_0^2}\frac{1}{n}\overline{\mathbf{W}_0^2}\mathbf{h}_0^1 + \mathbf{b}_0^2, \qquad (\mathbf{h}_0^2)_k = \frac{1}{n}\sum_{\alpha=1}^n \phi\left((\overline{\mathbf{W}_0^2})_{k\alpha}, (\mathbf{h}_0^1)_\alpha; \theta'_{W_0^2}, \{(b_0^2)_k\}_{k=1}^m\right) = \frac{1}{n}\theta'_{W_0^2}\sum_{\alpha=1}^n (\overline{\mathbf{W}_0^2})_{k\alpha}(\mathbf{h}_0^1)_\alpha + (\mathbf{b}_0^2)_k,$$

$$\mathbf{s} = \mathrm{T}(\mathbf{h}^2), \qquad (\texttt{Moment})$$

$$\tilde{\mathbf{h}}_0^2 = \mathrm{G}(\hat{\mathbf{h}}_0^2, \mathbf{s}), \qquad (\texttt{Moment},\ \text{See E})$$

$$\hat{\mathbf{h}}_0^{3,k} = \overline{\mathbf{W}^{3,k}}\mathbf{h}_0^1, \qquad \texttt{(MatMul)}$$

$$\mathbf{h}_0^{3,k} = \theta_{W_0^3}\hat{\mathbf{h}}_0^{3,k}, \qquad (\mathbf{h}_0^{3,k})_\alpha = \phi\left((\hat{\mathbf{h}}_0^{3,k})_\alpha; \theta_{W_0^3}\right) = \theta_{W_0^3}(\hat{\mathbf{h}}_0^3)_\alpha$$

$$\mathbf{h}_0^3 = \sum_{k=1}^m \mathbf{h}_0^{3,k}\tilde{h}_{0,k}^2, \qquad (\mathbf{h}_0^3)_\alpha = \phi\left((\mathbf{h}_0^{3,1})_\alpha,\dots,(\mathbf{h}_0^{3,m})_\alpha; \tilde{h}_{0,1}^2,\dots,\tilde{h}_{0,m}^2\right) = \sum_{k=1}^m (\mathbf{h}_0^{3,k})_\alpha\tilde{h}_{0,k}^2$$

$$\mathbf{f}_0 = \theta'_{W_0^4}\frac{1}{n}\overline{\mathbf{W}_0^4}\mathbf{h}_0^3, \qquad (\mathbf{f}_0)_\beta = \frac{1}{n}\sum_{\alpha=1}^n \phi\left((\overline{\mathbf{W}_0^4})_{\beta\alpha}, (\mathbf{h}_0^3)_\alpha; \theta'_{W_0^4}\right) = \frac{1}{n}\theta'_{W_0^4}\sum_{\alpha=1}^n (\overline{\mathbf{W}_0^4})_{\beta\alpha}(\mathbf{h}_0^3)_\alpha.$$

We assume stability at initialisation and thus the initialisation values in Setting A.3. For these values, the scale of $\mathbf{W}_0^2\mathbf{h}_0^1$ is $n^{-0.5}$.

Remark D.2 (Initial mask is deterministic conditioned on the bias). With the initialisation values in Setting A.3, as the width increases $\mathbf{W}_0^2\mathbf{h}_0^1 \to \mathbf{0}$, with vanishing finite-width fluctuation [39]. Consequently, if $\mathring{\mathbf{h}}_0^2$ denotes the limit of $\mathbf{h}_0^2$

$$\mathrm{T}(\mathring{\mathbf{h}}_0^2) = \mathrm{T}(\mathbf{b}_0^2).$$

Hence the Top-K operator is determined by the initial bias and is deterministic in the limit. This can be shown with the result in Appendix C.

Noise and Learnable Bias: We consider two cases here:

1. Case 1: A different noise is added at each time step $\{\mathbf{b}_t^2\}_{t=1}^T$.
2. Case 2: The term $\mathbf{b}_0^2$ is learned. Note that the updates below are written only for learnable biases.

First Backward Pass: Same as Appendix A.1, except the gradient $\mathrm{d}h_{0,k}^2$ requires gradient of the Top-K mask.

$$\mathrm{d}h_{0,k}^2 = \mathbf{s}_0\sum_{j\in\mathcal{S}} J(\tilde{\mathbf{h}}_0^2)_{jk}\mathrm{d}\tilde{h}_{0,j}^2, \qquad \in \mathbb{R}^{1\times d_{\text{out}}} \quad \texttt{(Moment)} \tag{380}$$

where $J(\tilde{\mathbf{h}}_0^2)_{jk} = \frac{\partial(\tilde{\mathbf{h}}_0^2)_j}{\partial(\hat{\mathbf{h}}_0^2)_k}$ is the Jacobian of the activation function on the sparse logits.

First Weight Updates: We need to consider the update to the learnable $\mathbf{b}^2$

$$\mathbf{b}_1^2 - \mathbf{b}_0^2 = -\eta\theta'_{W_0^4}\mathrm{d}\mathbf{h}_0^2\boldsymbol{\chi}_0. \tag{381}$$

The rest is the same as Appendix A.1.

First Feature Updates: The only feature update that will change from Appendix A.1 is $\mathbf{h}^2$ and $\tilde{\mathbf{h}}^2$:

$$\mathbf{h}_1^2 - \mathbf{h}_0^2 = \mathbf{W}_0^2(\mathbf{h}_1^1 - \mathbf{h}_0^1) + (\mathbf{W}_1^2 - \mathbf{W}_0^2)\mathbf{h}_1^1 + \mathbf{b}_1^2 - \mathbf{b}_0^2 \tag{382}$$

$$= \theta_{\delta h_1^1}\theta'_{W_0^2}\frac{\overline{\mathbf{W}_0^2}\delta\mathbf{h}_1^1}{n} - \theta'_{\delta W_1^2}\eta \mathrm{d}\mathbf{h}_0^2\boldsymbol{\chi}_0\frac{(\mathbf{h}_0^1)^T\mathbf{h}_1^1}{n} - \eta\theta'_{W_0^4}\mathrm{d}\mathbf{h}_0^2\boldsymbol{\chi}_0 \tag{383}$$

$$\theta_{h_1^2}(\delta\mathbf{h}_1^2)_k = \frac{1}{n}\sum_{\alpha=1}^{n}\phi\left(\overline{\mathbf{W}_0^2}_{k\alpha}, (\delta\mathbf{h}_1^1)_\alpha, (\mathbf{h}_0^1)_\alpha, (\mathbf{h}_1^1)_\alpha; \theta_{\delta h_1^1}, \theta'_{W_0^2}, \theta'_{\delta W_1^2}, \eta, \{(\mathrm{d}\mathbf{h}_0^2)_{k\beta}, (\boldsymbol{\chi}_0)_\beta\}_\beta^{d_{\text{out}}}\right) \qquad \texttt{(Moment)} \tag{384}$$

$$\phi(\ldots) := \theta_{\delta h_1^1}\theta'_{W_0^2}\overline{\mathbf{W}_0^2}_{k\alpha}(\delta\mathbf{h}_1^1)_\alpha - \theta'_{\delta W_1^2}\eta\sum_{\beta=1}^{d_{\text{out}}}(\mathrm{d}\mathbf{h}_0^2)_{k\beta}(\boldsymbol{\chi}_0)_\beta(\mathbf{h}_0^1)_\alpha(\mathbf{h}_1^1)_\alpha - \eta\theta'_{W_0^4}\sum_{\beta=1}^{d_{\text{out}}}(\mathrm{d}\mathbf{h}_0^2)_{k\beta}(\boldsymbol{\chi}_0)_\beta, \tag{385}$$

$$\theta_{\delta h_1^2} := \max(\theta_{\delta h_1^1}\theta'_{W_0^2}, \theta'_{\delta W_1^2}, \theta'_{W_0^4}). \tag{386}$$

$$\tilde{\mathbf{h}}_1^2 - \tilde{\mathbf{h}}_0^2 = \mathrm{G}(\mathbf{h}_1^2, \mathbf{s}_1) - \mathrm{G}(\mathbf{h}_0^2, \mathbf{s}_0), \qquad \texttt{(Moment)} \tag{387}$$

The rest of the feature updates are the same as Appendix A.1.

***t*th Backward Pass:** Same as Appendix A.1, except for the mask

$$\mathrm{d}h_{t,k}^2 = \mathbf{s}_t\sum_{j\in\mathcal{S}}J(\tilde{\mathbf{h}}_t^2)_{jk}\mathrm{d}\tilde{h}_{t,j}^2, \qquad \in \mathbb{R}^{1\times d_{\text{out}}} \quad \texttt{(Moment)} \tag{388}$$

where $J(\tilde{\mathbf{h}}_t^2)_{jk} = \frac{\partial(\tilde{\mathbf{h}}_t^2)_j}{\partial(\tilde{\mathbf{h}}_t^2)_k}$ is the Jacobian of the activation function on the sparse logits.

***t*th Weight Updates:** We need to consider the update to $\mathbf{b}^2$

$$\mathbf{b}_t^2 - \mathbf{b}_{t-1}^2 = -\eta\theta'_{W_{t-1}^4}\mathrm{d}\mathbf{h}_{t-1}^2\boldsymbol{\chi}_{t-1}. \tag{389}$$

The rest is the same as Appendix A.1.

***t*th Feature Updates:** The only feature update that will change from Appendix A.1 is $\mathbf{h}^2$ and $\hat{\mathbf{h}}^2$:

$$\mathbf{h}_t^2 - \mathbf{h}_{t-1}^2 = \mathbf{W}_{t-1}^2(\mathbf{h}_t^1 - \mathbf{h}_{t-1}^1) + (\mathbf{W}_t^2 - \mathbf{W}_{t-1}^2)\mathbf{h}_t^1 + \mathbf{b}_t^2 - \mathbf{b}_{t-1}^2 \tag{390}$$

$$= \theta_{\delta h_t^1}\theta'_{W_{t-1}^2}\frac{\overline{\mathbf{W}_{t-1}^2}\delta\mathbf{h}_t^1}{n} - \theta'_{\delta W_t^2}\eta\mathrm{d}\mathbf{h}_{t-1}^2\boldsymbol{\chi}_{t-1}\frac{(\mathbf{h}_{t-1}^1)^T\mathbf{h}_t^1}{n} - \eta\theta'_{W_{t-1}^4}\mathrm{d}\mathbf{h}_{t-1}^2\boldsymbol{\chi}_{t-1} \tag{391}$$

$$\theta_{h_t^2}(\delta\mathbf{h}_t^2)_k = \frac{1}{n}\sum_{\alpha=1}^{n}\phi\left(\overline{\mathbf{W}_{t-1}^2}_{k\alpha}, (\delta\mathbf{h}_t^1)_\alpha, (\mathbf{h}_{t-1}^1)_\alpha, (\mathbf{h}_t^1)_\alpha; \theta_{\delta h_t^1}, \theta'_{W_{t-1}^2}, \theta'_{\delta W_t^2}, \eta, \{(\mathrm{d}\mathbf{h}_{t-1}^2)_{k\beta}, (\boldsymbol{\chi}_{t-1})_\beta\}_\beta^{d_{\text{out}}}\right) \tag{392}$$

$$\phi(\ldots) := \theta_{\delta h_t^1}\theta'_{W_{t-1}^2}\overline{\mathbf{W}_{t-1}^2}_{k\alpha}(\delta\mathbf{h}_t^1)_\alpha - \theta'_{\delta W_t^2}\eta\sum_{\beta=1}^{d_{\text{out}}}(\mathrm{d}\mathbf{h}_{t-1}^2)_{k\beta}(\boldsymbol{\chi}_{t-1})_\beta(\mathbf{h}_{t-1}^1)_\alpha(\mathbf{h}_t^1)_\alpha - \eta\theta'_{W_{t-1}^4}\sum_{\beta=1}^{d_{\text{out}}}(\mathrm{d}\mathbf{h}_{t-1}^2)_{k\beta}(\boldsymbol{\chi}_{t-1})_\beta, \tag{393}$$

$$\theta_{\delta h_t^2} := \max(\theta_{\delta h_t^1}\theta'_{W_{t-1}^2}, \theta'_{\delta W_t^2}, \theta'_{W_{t-1}^4}). \tag{394}$$

$$\tilde{\mathbf{h}}_t^2 - \tilde{\mathbf{h}}_{t-1}^2 = \mathrm{G}(\mathbf{h}_t^2) - \mathrm{G}(\mathbf{h}_{t-1}^2), \qquad \texttt{(Moment)} \tag{395}$$

The rest will be the same as Appendix A.1.

D.2 Infinite Width Limit for TopK MoE with SGD

We use the same notation as in Appendix A.2.

Remark D.3 (Deterministic Limiting Mask Conditioned on Noise/Bias). To find the limit, we need to show that the Top-K mask is deterministic in both cases: 1) where noise is added at each step to the router logits, 2) where a learnable bias is added to the router logits.

1. Note that the noise $\mathbf{b}_t^2$ is drawn independently of $\mathbf{W}_t\mathbf{h}_t^1$ at each step. This implies that the probability of ties $(\mathbf{h}^2)_i = (\mathbf{h}^2)_j$ is 0. $\mathbf{W}_t\mathbf{h}_t^1$ converges deterministically according to the Master Theorem: If $\mathbf{W}_t\mathbf{h}_t^1$ converges to the same value for each dimension such that the setting of Appendix C holds for $\mathbf{W}_t\mathbf{h}_t^1$, then the mask is $\mathrm{T}(\mathbf{b}_t^2)$ in the limit and thus deterministic.

2. With a learnable bias, the mask is deterministic at initialisation (Remark D.2). However, for subsequent steps, we must make the additional assumption that for widths $n > N$ for some $N > 0$, in the resulting logits $h^2_{t,(K)} - h^2_{t,(K+1)} > \epsilon$ where ϵ is width independent, for all t. As TP only allows for pseudo-Lipschitz non-linearities, scalars in the program converge to a deterministic value, and the logits are bounded under μP-MoE, this can be proven using induction. However, we leave this for future work.

First Forward Pass:

$$Z^{h_0^1} = \boldsymbol{\xi} Z^{W_0^1}, \tag{396}$$

$$\mathring{\mathbf{h}}_0^2 = \mathbf{b}_0^2, \tag{397}$$

$$\mathring{\mathbf{s}}_0 = \mathrm{T}(\mathring{\mathbf{h}}_0^2) \tag{398}$$

$$\mathring{\tilde{\mathbf{h}}}_0^2 = \mathrm{G}(\mathring{\mathbf{h}}_0^2, \mathring{\mathbf{s}}_0) \tag{399}$$

$$Z^{h_0^{3,k}} = Z^{W_0^{3,k} h_0^1} \text{ for every expert } k, \tag{400}$$

$$Z^{h_0^3} = \sum_{k=1}^{m} \mathring{\tilde{h}}^2_{0,k} Z^{h_0^{3,k}} \tag{401}$$

$$\mathring{\mathbf{f}}_0 = \mathbf{0}. \tag{402}$$

Here, $Z^{W_0^1}$ is standard Gaussian by construction. The gating function G is either a Softmax or Sigmoid. $Z^{W_0^{3,k} h_0^1} = \hat{Z}^{W_0^{3,k} h_0^1}$ from the Z MatMul rule. Finally $\mathring{\mathbf{f}}_0 = \mathbf{0}$ as $\mathring{f}_0 = \mathbb{E}[Z^{W_0^4}]\mathbb{E}[Z^{h_0^3}] = 0$.

First Backward Pass:

$$Z^{\mathrm{d}h_0^3} = Z^{\overline{W_0^4}}, \tag{403}$$

$$\mathring{\mathrm{d}\tilde{h}}^2_{0,k} = 0 \text{ for every expert } k, \tag{404}$$

$$\mathring{\mathrm{d}h}^2{}_{0,k} = 0 \text{ for every expert } k, \tag{405}$$

$$Z^{\mathrm{d}h_0^{3,k}} = \tilde{h}^2_{0,k} Z^{\mathrm{d}h_0^3} \text{ for every expert } k, \tag{406}$$

$$Z^{\mathrm{d}h_0^1} = \sum_{k=1}^{m} Z^{(W_0^{3,k})^T \mathrm{d}h_0^{3,k}} \tag{407}$$

$$\tag{408}$$

$Z^{\overline{W_0^4}}$ is a standard Gaussian by construction. $\mathring{\mathrm{d}\tilde{h}}^2_{0,k} = \mathbb{E}[Z^{h_0^{3,k}}]\mathbb{E}[\overline{W_0^4}] = 0$. From this, $\mathring{\mathrm{d}h}^2{}_{0,k} = 0$ follows. $Z^{(W_0^{3,k})^T \mathrm{d}h_0^3} = \hat{Z}^{(W_0^{3,k})^T \mathrm{d}h_0^3}$ from the Z MatMul rule.

First Feature Update:

$$Z^{\delta h_1^1} = -\eta \sum_{\beta=1}^{d_{\text{out}}} Z^{(\mathrm{d}h_0^1)_\beta} (\mathring{\boldsymbol{\chi}}_0)_\beta \boldsymbol{\xi}^T \boldsymbol{\xi}, \tag{409}$$

$$\delta \mathring{h}_{1,k}^2 = \mathring{\theta}_{\delta h_1^1} \mathbb{E}[Z^{(\overline{W_0^2})_{k,:}} Z^{\delta h_1^1}] - \mathring{\theta}'_{\delta W_1^2} \eta \sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathring{h}_{0,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_0)_\beta \mathbb{E}[Z^{h_0^1} Z^{h_1^1}] - \eta \mathring{\theta}'_{W_0^4} \sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathring{h}_{0,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_0)_\beta, \tag{410}$$

$$\delta \mathring{\mathbf{s}}_1^2 = \mathrm{T}(\mathring{\mathbf{h}}_1^2) - \mathrm{T}(\mathring{\mathbf{h}}_0^2) \text{ if } \mathring{\theta}_{\delta h_1^2} = 1, \tag{411}$$

$$\delta \mathring{\tilde{\mathbf{h}}}_1^2 = \mathrm{G}(\mathring{\mathbf{h}}_1^2, \mathring{\mathbf{s}}_1) - \mathrm{G}(\mathring{\mathbf{h}}_0^2, \mathring{\mathbf{s}}_0) \text{ if } \mathring{\theta}_{\delta h_1^2} = 1, \tag{412}$$

$$Z^{\delta h_1^{3,k}} = \mathring{\theta}_{\delta h_1^1 / \delta h_1^{3,k}} Z^{W_0^{3,k} \delta h_1^1} - \mathring{\theta}'_{\delta W_1^3 / \delta h_1^{3,k}} \eta \sum_{\beta=1}^{d_{\text{out}}} Z^{(\mathrm{d}h^{3,k})_\beta} (\mathring{\boldsymbol{\chi}}_0)_\beta \mathbb{E}[Z^{h_0^1} Z^{h_1^1}] \text{ for every expert } k, \tag{413}$$

$$Z^{\delta h_1^3} = \sum_{k=1}^{m} \mathring{\theta}_{\delta h_1^{3,k} / \delta h_1^3} \mathring{\tilde{h}}_{0,k}^2 Z^{\delta h_1^{3,k}} + \mathring{\theta}_{\delta \tilde{h}_1^2} \mathring{\theta}_{h_1^{3,k} / \delta h^3} \delta \mathring{\tilde{h}}_{1,k}^2 Z^{h_1^{3,k}}, \tag{414}$$

$$\delta \mathring{f}_1 = \mathring{\theta}_{\delta h_1^3} \mathbb{E}[Z^{\overline{W_0^4}} Z^{\delta h_1^3}] - \mathring{\theta}'_{\delta W_1^4} \eta \mathring{\chi}_0 \mathbb{E}[Z^{h_0^3} Z^{h_1^3}], \tag{415}$$

where G is either softmax or sigmoid and G′ is the derivative. Further, $Z^{W_0^{3,k} \delta h_1^1}$ is given by the Z MatMul rule,

$$Z^{W_0^{3,k} \delta h_1^1} = \hat{Z}^{W_0^{3,k} \delta h_1^1} + \dot{Z}^{W_0^{3,k} \delta h_1^1} \tag{416}$$

$$= \hat{Z}^{W_0^{3,k} \delta h_1^1} + Z^{\mathrm{d}h_0^{3,k}} \mathbb{E}\left[\frac{\partial Z^{\delta h_1^1}}{\partial \hat{Z}^{(W_0^{3,k})^T \mathrm{d}h_0^{3,k}}} \right]. \tag{417}$$

We write the limit of each vector in the program as

$$Z^{h_t} = Z^{h_0} + \mathring{\theta}_{\delta h_1} Z^{\delta h_1} + \ldots + \mathring{\theta}_{\delta h_t} Z^{\delta h_t}. \tag{418}$$

Weight updates: The distribution of certain weight updates are required in the backward pass and thus the forward passes. Note that $\mathbf{b}$ is a scalar in the program ($\in \mathbb{R}^m$) and hence does not have a limit.

$$Z^{\overline{\delta W_t^4}} = -\eta \mathring{\chi}_{t-1} Z^{h_{t-1}^3}, \tag{419}$$

$$Z^{(\overline{\delta W_t^2})_{k,:}} = -\eta \sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathring{h}_{t-1,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_{t-1})_\beta Z^{h_{t-1}^1}, \tag{420}$$

$$\mathbf{b}_t^2 - \mathbf{b}_{t-1}^2 = -\eta \mathring{\theta}'_{W_{t-1}^4} \sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathring{h}_{t-1,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_{t-1})_\beta. \tag{421}$$

tth Backward Pass: Let $\mathcal{S} = \mathrm{TopK}(\mathbf{h}_t^2)$ be the set of active experts,

$$Z^{\mathrm{d}h_t^3} = Z^{(\overline{W_t^4})^T}, \tag{422}$$

$$\mathrm{d}\mathring{\tilde{h}}_{t,k}^2 = \mathbb{E}[Z^{h_t^{3,k}} Z^{\mathrm{d}h_t^3}], \tag{423}$$

$$\mathrm{d}\mathring{h}_{t,k}^2 = \mathring{s}_{t,k} \sum_{j=1}^{m} J(\mathring{h}_t^2)_{kj} \mathrm{d}\mathring{\tilde{h}}_{t,j}^2, \tag{424}$$

$$Z^{\mathrm{d}h_t^{3,k}} = \mathring{\tilde{h}}_{t,k}^2 Z^{\mathrm{d}h_t^3}, \tag{425}$$

$$Z^{\mathrm{d}h_t^1} = \sum_{k=1}^{m} Z^{(\overline{W_t^2})_{k,:}} \mathrm{d}\mathring{h}_{t,k}^2 + Z^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} - \eta \sum_{s=0}^{t-1} Z^{h_s^1} \mathring{\chi}_s \mathbb{E}[Z^{\mathrm{d}h_s^{3,k}} Z^{\mathrm{d}h_t^{3,k}}], \tag{426}$$

where $Z^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}}$ is given by the Z MatMul rule,

$$Z^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} = \hat{Z}^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} + \dot{Z}^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} \tag{427}$$

$$= \hat{Z}^{(W_0^{3,k})^T \mathrm{d}h_t^{3,k}} + \sum_{v \in \mathcal{V}: W_0^{3,k} v \in \mathcal{V}} Z^v \mathbb{E}\left[\frac{\partial Z^{\mathrm{d}h_t^{3,k}}}{\partial \hat{Z}^{W_0^{3,k} v}}\right], \tag{428}$$

where $\mathcal{V} := \{v \text{ for some vector in the program } v\}$ is the set of all vectors in the program.

tth Forward Pass:

$$Z^{\delta h_t^1} = -\eta \sum_{\beta=1}^{d_{\text{out}}} Z^{(\mathrm{d}\mathbf{h}_{t-1}^1)_\beta} (\mathring{\boldsymbol{\chi}}_{t-1})_\beta \boldsymbol{\xi}^T \boldsymbol{\xi}, \tag{429}$$

$$\delta \mathring{h}_{t,k}^2 = \mathring{\theta}'_{W_{t-1}^2} \mathring{\theta}_{\delta h_t^1} \mathbb{E}[Z^{(\overline{W_{t-1}^2})_{k,:}} Z^{\delta h_t^1}] - \mathring{\theta}'_{\delta W_t^2} \eta \sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathring{h}_{t-1,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_{t-1})_\beta \mathbb{E}[Z^{h_{t-1}^1} Z^{h_t^1}] \tag{430}$$

$$- \eta \mathring{\theta}'_{W_{t-1}^4} \sum_{\beta=1}^{d_{\text{out}}} (\mathrm{d}\mathring{h}_{t-1,k}^2)_\beta (\mathring{\boldsymbol{\chi}}_{t-1})_\beta, \tag{431}$$

$$\delta \mathring{\mathbf{s}}_t^2 = \mathrm{T}(\mathring{\mathbf{h}}_t^2) - \mathrm{T}(\mathring{\mathbf{h}}_{t-1}^2) \text{ if } \mathring{\theta}_{\delta h_t^2} = 1, \tag{432}$$

$$\delta \mathring{\tilde{\mathbf{h}}}_t^2 = \mathrm{G}(\mathring{\mathbf{h}}_t^2, \mathbf{s}_t) - \mathrm{G}(\mathring{\mathbf{h}}_{t-1}^2, \mathbf{s}_{t-1}) \text{ if } \mathring{\theta}_{\delta h_t^2} = 1, \tag{433}$$

$$Z^{\delta h_t^{3,k}} = \mathring{\theta}_{\delta h_t^1 / \delta h_t^{3,k}} Z^{W_0^{3,k} \delta h_t^1} - \sum_{s=0}^{t-2} \mathring{\theta}'_{\delta W_{s+1}^{3,k} \delta h_t^1 / \delta h_t^{3,k}} \eta \sum_{\beta}^{d_{\text{out}}} Z^{(\mathrm{d}h_s^{3,k})_\beta} (\mathring{\boldsymbol{\chi}}_s)_\beta \mathbb{E}[Z^{h_s^1} Z^{\delta h_t^1}] \tag{434}$$

$$- \mathring{\theta}'_{\delta W_t^{3,k} / \delta h_t^{3,k}} \eta \sum_{\beta}^{d_{\text{out}}} Z^{(\mathrm{d}h_{t-1}^{3,k})_\beta} (\mathring{\boldsymbol{\chi}}_{t-1})_\beta \mathbb{E}[Z^{h_{t-1}^1} Z^{h_t^1}], \tag{435}$$

$$Z^{\delta h_t^3} = \sum_{k=1}^{m} \mathring{\theta}_{\delta h_t^{3,k} / \delta h_t^3} \mathring{\tilde{h}}_{k,t-1}^2 Z^{\delta h_t^{3,k}} + \mathring{\theta}_{\delta \tilde{h}_t^2 / \delta h_t^3} \delta \mathring{\tilde{h}}_{t,k}^2 Z^{h_t^{3,k}}, \tag{436}$$

$$\delta \mathring{f}_t = \mathring{\theta}'_{W_{t-1}^4} \mathring{\theta}_{\delta h_t^3} \mathbb{E}[Z^{\overline{W_{t-1}^4}} Z^{\delta h_t^3}] - \mathring{\theta}'_{\delta W_t^4} \eta \mathring{\chi}_{t-1} \mathbb{E}[Z^{h_{t-1}^3} Z^{h_t^3}], \tag{437}$$

where G is either softmax or sigmoid and G$'$ is the derivative, and

$$Z^{W_0^{3,k} \delta h_t^1} = \hat{Z}^{W_0^{3,k} \delta h_t^1} + \dot{Z}^{W_0^{3,k} \delta h_t^1} \tag{438}$$

$$= \hat{Z}^{W_0^{3,k} \delta h_t^1} + \sum_{s=0}^{t-1} Z^{\mathrm{d}h_s^{3,k}} \mathbb{E}\left[\frac{\partial Z^{\delta h_t^1}}{\partial \hat{Z}^{(W_0^{3,k})^T \mathrm{d}h_s^{3,k}}}\right] \tag{439}$$

Remark D.4 (μP-MoE gives feature learning and stability). As the infinite-width limit does not change much for the Top-K case. It is easy to see that the μP-MoE values in Table 1 for SGD and Table 2 for Adam hold for Top-K MoEs.

E Expressing softmax and sigmoid over a multi-scalar as a moment operation in TP

We can express the sum of l scalars $\theta_1, \ldots, \theta_k \in \mathbb{R}$ using the **Moment** operations with no vectors ($k = 0$):

$$\phi : \mathbb{R}^0 \times \mathbb{R}^l \to \mathbb{R}, \qquad \phi(-; \theta_1, \ldots, \theta_k) = \sum_{i=1}^{k} \theta_i.$$

Then

$$\frac{1}{n} \sum_{\alpha=1}^{n} \phi(-; \theta_1, \ldots, \theta_l) = \frac{1}{n} \sum_{\alpha=1}^{n} \Big(\sum_{i=1}^{l} \theta_i \Big) = \sum_{i=1}^{l} \theta_i,$$

since the expression inside the sum is constant with respect to α.

Softmax: We can do the same with a softmax. Simply define the function ϕ as the softmax function over the scalars.

$$\phi : \mathbb{R}^0 \times \mathbb{R}^l \to \mathbb{R}, \qquad \phi(-; \theta_1, \ldots, \theta_k) = \frac{\exp \theta_i}{\sum_{i=1}^k \exp \theta_i}.$$

Then

$$\frac{1}{n} \sum_{\alpha=1}^{n} \phi(-; \theta_1, \ldots, \theta_l) = \frac{1}{n} \sum_{\alpha=1}^{n} \frac{\exp \theta_i}{\sum_{i=1}^k \exp \theta_i} = \frac{\exp \theta_i}{\sum_{i=1}^k \exp \theta_i},$$

so each of these operations generates a valid scalar. **Sanity check.** Note that if a sequence of random vectors (multi-scalars in TP language)

$$\theta^{(n)} = (\theta_1^{(n)}, \theta_2^{(n)}, \cdots \theta_l^{(n)}) \longrightarrow \theta^{(*)} = (\theta_1^{(*)}, \theta_2^{(*)}, \cdots \theta_l^{(*)})$$

where the convergence may be in probability, distribution, or almost sure, then the continuous mapping theorem states that if $\phi : \mathbb{R}^l \to \mathbb{R}$ is continuous at $\theta^{(*)}$, then

$$\phi(\theta^{(n)}) = \phi(\theta_1^{(n)}, \theta_2^{(n)}, \cdots \theta_l^{(n)}) \longrightarrow \phi(\theta^{(*)}) = \phi(\theta_1^{(*)}, \theta_2^{(*)}, \cdots \theta_l^{(*)})$$

in the same mode of convergence (probability, distribution, a.s). So continuity at the limit alone suffices for a.s convergence to hold.

TP requires that the non-linearities have polynomially bounded derivatives which are of course much stronger and clearly satisfied by the softmax.

Sigmoid: We can simply express the sigmoid of a scalar with the function ϕ defined as

$$\phi : \mathbb{R}^0 \times \mathbb{R} \to \mathbb{R}, \qquad \phi(-; \theta_1) = \frac{1}{1 + \exp(-\theta_1)}.$$

Then

$$\frac{1}{n} \sum_{\alpha=1}^{n} \phi(-; \theta_1) = \frac{1}{1 + \exp(-\theta_1)}.$$

F Additional Experiments and Details

F.1 Details for Figures

Figures in the main paper: All figures in the main paper are created by training for $t = 20$ steps on Fashion-MNIST. The network used is the same as Definition 2.1 but with non-linear experts with ReLU activation. The models contained 4 experts, and $K = 2$ were executed in the Top-K case. Hyperparameters of the model such as multiplier (for μP-MoE) and learning rate were tuned so that the figures showed effective learning. The router and function output are zero-initialised to show limiting behaviour [43].

HP Transfer in Section 5.2: HP transfer of learning rate in Section 5.2 was carried out using a Mixtral style transformer language model [21] trained on Wikitext-103-v1 for 1 epoch. The model was made smaller due to compute constraints. The model contained three transformer layers with feedforward width of 4 times the width dimension. A default load balancing loss coefficient of 0.01 was fixed. Hyperparameters such as the multipliers (for output, router, inputs) and learning rate were randomly sampled 100 times and were tuned based on the validation loss. The model consisted of 8 experts and 2 experts were executed per token. QK Layernorm was implemented as it has been shown to be more stable for HP transfer [16].

Compute Resources: The experiments for this paper (including the Mixtral run) were run on a single H100 GPU.

F.2 Coordinate Checks for Soft Routing

We show additional coordinate checks in for sigmoid soft in Figure 6 and softmax soft in Figure 5.

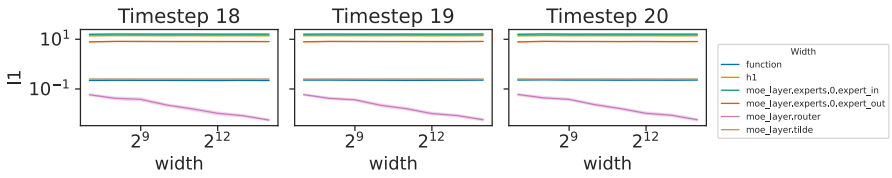


Figure 5: Coordinate check for Softmax-Soft MoE. Shows that the router feature scales $n^{-0.5}$ as predicted in Remark B.10. Tilde signifies the router probability.

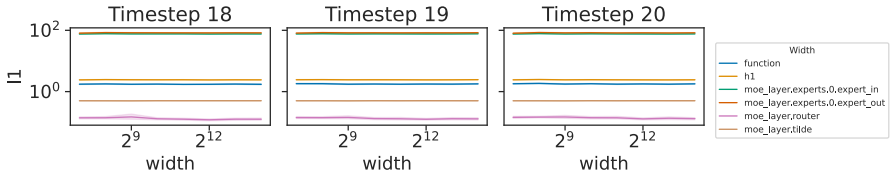


Figure 6: Coordinate check for Sigmoid-Soft MoE. Shows that the router feature learns in comparison with Softmax gating. Tilde signifies the router probability.

F.3 Coordinate Checks for Top-K Routing

We show additional coordinate checks for sigmoid Top-K in Figure 8 and Figure 7. We additionally verify that μP-MoE scaling (Figure 10) allows for feature learning in Mixtral style transformer MoEs compare to SP-MoE (Figure 9).

F.4 Additional HP Transfer Experiments

We show in Algorithm 1 how to transfer hyperparameters conditional on noise terms/bias.

We carry out additional HP transfer with a similar network to Definition 2.1, but with non-linear experts and Top-2 out of 4 expert activation. In Figure 11 we show HP transfer with and without the bias term on tinyimagenet. In Figure 12 we show HP transfer with and without the bias term on emnist.

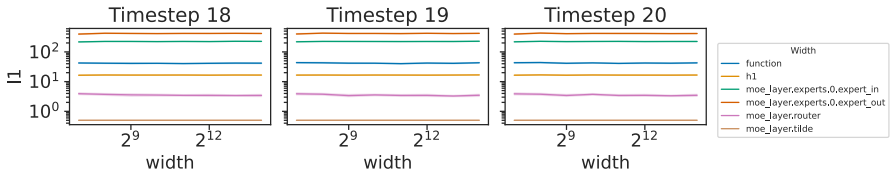


Figure 7: Coordinate check for Softmax-Top-K MoE with 2/4 active experts per forward pass. Shows that in comparison to the soft routing case, the Top-K operation can allow for expert specialisation and hence route feature learning. Tilde signifies the router probability.

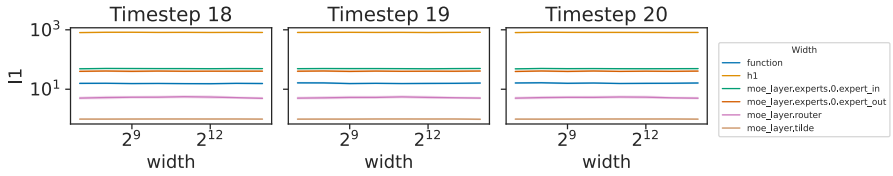


Figure 8: Coordinate check for Sigmoid-Top-K MoE with 2/4 active experts per forward pass. Shows that like in the soft routing case, the router exhibits feature learning. Tilde signifies the router probability.

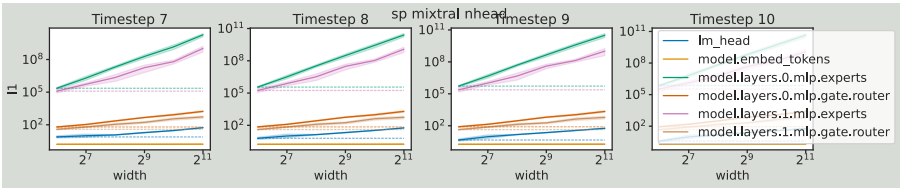


Figure 9: Coordinate check for Top-K Mixtral MoE with SP. Shows that the network diverges with increasing width.

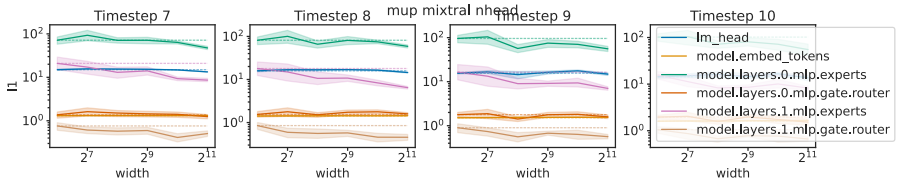
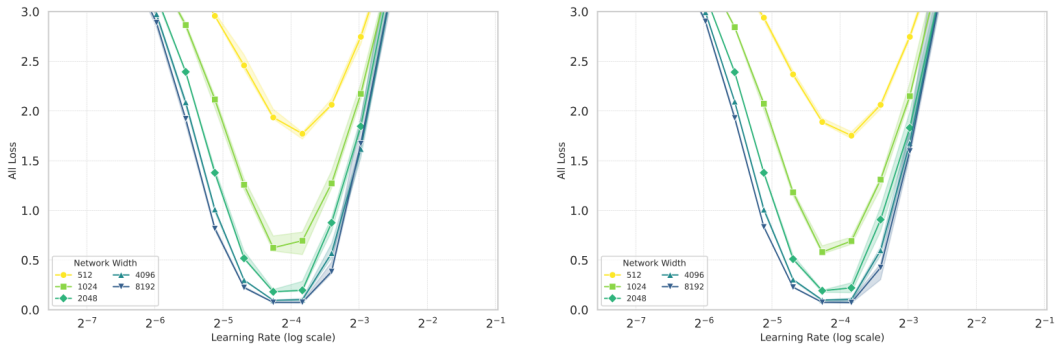


Figure 10: Coordinate check for Top-K Mixtral MoE with μP-MoE. Shows that the network fearture learns as with increases.

Algorithm 1 μP HP-transfer for Top-K MoE with a shared router-bias schedule

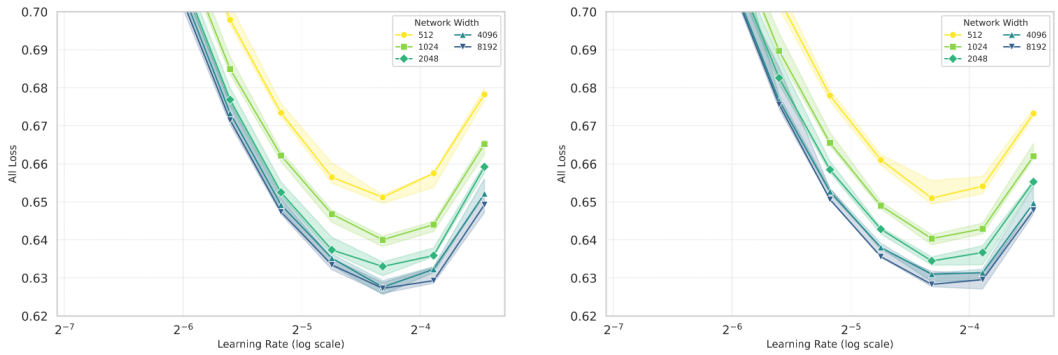
Require: Base width n_0, target width n, steps T, experts m, Top-K, bias scale σ_{b^2}

1: **Bias schedule (fix once, reuse for all widths):** sample $\{b_t^2\}_{t=0}^{T}$ with $b_t^2 \sim \mathcal{N}(0, \sigma_{b^2} I_m)$ using a fixed seed.
2: **Training loop (given HPs $\mathcal{H}(\cdot)$) at width $w \in \{n_0, n\}$:**
3: **for** $t = 0$ to T **do**
4: $\quad h_t^2 \leftarrow W_t^2 h_t^1 + b_t^2; \quad I_t \leftarrow \mathrm{TopK}(h_t^2)$
5: $\quad$ Route to experts in I_t; update parameters.
6: **end for**
7: **Tune (width n_0):** run the training loop with μP-heuristic scaling $\Rightarrow$ tuned HPs $\mathcal{H}(n_0)$.
8: **Transfer (width n):** initialise width-n model with μP-heuristic and $\mathcal{H}(n)$; run the same training loop.



(a) With Bias (b) Without Bias

Figure 11



(a) With Bias

(b) Without Bias

Figure 12

G Extension to MoE Transformer Block and Non-linear Experts

Our analysis in this paper is carried out for the MoE network with linear experts (Definition 2.1) for ease of exposition. We show here that our results hold when the MoE block is a part of standard μP transformer architectures [43], and the experts are non-linear.

G.1 MoE Transformer Block

For our results to hold when the MoE block (Definition 2.10) is a part of a standard transformer architecture, we require two properties:

1. The MoE block in the transformer retains the same Tensor Program structure and alignment as the MoE block in the MoE network.
2. Inputs into the MoE block scale $\Theta(1)$ (h^1 in Definition 2.10 scales $\Theta(1)$), and gradients into the MoE block scale $\Theta(n^{-1})$ ($\frac{\partial \mathcal{L}}{\partial \mathbf{h}^3}$ in Definition 2.10 scales $\Theta(n^{-1})$).

We first show that the embedded block retains the single-block alignment structure, and then use this to establish the coordinate scales at the block interfaces (inputs and outputs). Consider a transformer with L layers of the form

$$\begin{aligned} u^\ell &= x^\ell + \mathrm{Attn}^\ell\left(\mathrm{Norm}_1^\ell(x^\ell)\right), \\ \mathbf{h}^{1,(\ell)} &= \mathrm{Norm}_2^\ell(u^\ell), \qquad \mathbf{h}^{3,(\ell)} = \mathrm{MoE}^\ell(\mathbf{h}^{1,(\ell)}), \\ x^{\ell+1} &= u^\ell + \mathbf{h}^{3,(\ell)}, \qquad \ell = 1, \dots, L. \end{aligned} \tag{440}$$

Here L, the number of experts, and all other non-width dimensions are fixed as $n \to \infty$. The attention, normalisation, input embedding, and readout use μP, while each MoE block uses the μP-MoE parametrisation in Appendix B.2. We work under the regularity assumptions of the single-block analysis and assume no leading-order cancellation in residual sums or their backward counterparts. For Top-K routing, we additionally impose the limiting-mask conditions of Remark D.3 at each block.

Proposition G.1 (Tensor-Program structure of an embedded MoE block)**.** *The transformer embedded MoE block retains the single-block forward and backward pass Tensor-Program structure as the MoE block in the MoE network (Definition 2.1).*

Proof. Let $\Delta_t v := v_t - v_{t-1}$ and $\mathbf{g}_t^{2,(\ell)} := \nabla_{\mathbf{h}_t^{2,(\ell)}} \mathcal{L}_t$. The router-logit update in block ℓ is

$$\Delta_t \mathbf{h}^{2,(\ell)} = \mathbf{W}_{t-1}^{2,(\ell)} \Delta_t \mathbf{h}^{1,(\ell)} + \Delta_t \mathbf{W}^{2,(\ell)} \mathbf{h}_t^{1,(\ell)} + \Delta_t \mathbf{b}^{2,(\ell)}. \tag{441}$$

For SGD, suppressing the width-dependent prefactors already specified by the parametrisation,

$$\begin{aligned} \Delta_t \mathbf{W}^{2,(\ell)} &= -\eta \mathbf{g}_{t-1}^{2,(\ell)} (\mathbf{h}_{t-1}^{1,(\ell)})^\top, \\ \mathbf{W}_{t-1}^{2,(\ell)} &= \mathbf{W}_0^{2,(\ell)} - \eta \sum_{s=0}^{t-2} \mathbf{g}_s^{2,(\ell)} (\mathbf{h}_s^{1,(\ell)})^\top. \end{aligned} \tag{442}$$

Substituting these expressions into Equation (441) gives

$$\begin{aligned} \Delta_t \mathbf{h}^{2,(\ell)} = \mathbf{W}_0^{2,(\ell)} \Delta_t \mathbf{h}^{1,(\ell)} \\ - \eta \sum_{s=0}^{t-2} \mathbf{g}_s^{2,(\ell)} \left\langle \mathbf{h}_s^{1,(\ell)}, \Delta_t \mathbf{h}^{1,(\ell)} \right\rangle \\ - \eta \mathbf{g}_{t-1}^{2,(\ell)} \left\langle \mathbf{h}_{t-1}^{1,(\ell)}, \mathbf{h}_t^{1,(\ell)} \right\rangle + \Delta_t \mathbf{b}^{2,(\ell)}. \end{aligned} \tag{443}$$

Thus, the rows of the router weight update are explicitly aligned with $\mathbf{h}_{t-1}^{1,(\ell)}$, giving the effective update in the third term. Likewise, the learned part of the propagating update is aligned with the previous block inputs $\{\mathbf{h}_s^{1,(\ell)}\}_{s=0}^{t-2}$, giving the second term.

Using the same normalisation and normalised feature update $\delta \mathbf{h}_t^{1,(\ell)}$ as in Appendix A.1, the corresponding inner products give the same Tensor-Program `Moment` operations:

$$\mathbb{E}\left[Z^{h_s^{1,(\ell)}} Z^{\delta h_t^{1,(\ell)}}\right], \qquad \mathbb{E}\left[Z^{h_{t-1}^{1,(\ell)}} Z^{h_t^{1,(\ell)}}\right],$$
$$\mathbb{E}\left[Z^{(\overline{W_0^{2,(\ell)}})_{k,:}} Z^{\delta h_t^{1,(\ell)}}\right], \tag{444}$$

where the last expression is the initial-router contribution.

The surrounding attention, normalisation, residual, and preceding MoE subprograms generate $Z^{\delta h_t^{1,(\ell)}}$ jointly with the local router variables, but do not change the algebraic form of the router update. Hence, the transformer Tensor Program tracks the relevant alignments directly. Under the same conditions used in the single-block μP analysis, the propagating and effective contributions retain their single-block asymptotic orders. The Adam case is analogous, using the Adam Tensor-Program updates in Appendix A.3.

For the backward pass, suppress ℓ and write $\mathbf{g}_t^j := \nabla_{\mathbf{h}_t^j} \mathcal{L}_t$. Let $d_t := \nabla_{\mathbf{h}_t^3} \mathcal{L}_t = \nabla_{x_t^{\ell+1}} \mathcal{L}_t$ be the incoming gradient, $q_t^k := \nabla_{\mathbf{h}_t^{3,k}} \mathcal{L}_t$ the gradient into expert k, and $r_t := \nabla_{\tilde{\mathbf{h}}_t^2} \mathcal{L}_t$ the gradient into the gating weights. The chain rule gives

$$q_t^k = \tilde{h}_{t,k}^2 d_t, \qquad r_{t,k} = \langle \mathbf{h}_t^{3,k}, d_t \rangle,$$
$$\mathbf{g}_t^2 = J_{\mathrm{G}}(\mathbf{h}_t^2)^\top r_t, \qquad \mathbf{g}_t^1 = (\mathbf{W}_t^2)^\top \mathbf{g}_t^2 + \sum_{k=1}^m (\mathbf{W}_t^{3,k})^\top q_t^k. \tag{445}$$

Thus the gradient into the block input has a router path and an expert path. Under SGD, restoring the width-dependent learning-rate factors,

$$\mathbf{W}_t^2 = \mathbf{W}_0^2 - \eta n^{-c_2} \sum_{s=0}^{t-1} \mathbf{g}_s^2 (\mathbf{h}_s^1)^\top,$$
$$\mathbf{W}_t^{3,k} = \mathbf{W}_0^{3,k} - \eta n^{-c_3} \sum_{s=0}^{t-1} q_s^k (\mathbf{h}_s^1)^\top. \tag{446}$$

Substituting into Equation (445) gives

$$\mathbf{g}_t^1 = (\mathbf{W}_0^2)^\top \mathbf{g}_t^2 + \sum_{k=1}^m (\mathbf{W}_0^{3,k})^\top q_t^k$$
$$- \eta n^{-c_2} \sum_{s=0}^{t-1} \mathbf{h}_s^1 \langle \mathbf{g}_s^2, \mathbf{g}_t^2 \rangle - \eta n^{-c_3} \sum_{k=1}^m \sum_{s=0}^{t-1} \mathbf{h}_s^1 \langle q_s^k, q_t^k \rangle. \tag{447}$$

The learned terms are therefore explicitly aligned with the previous block inputs $\mathbf{h}_s^1$, just as in the single-block case. With the single-block normalisations $\widehat{d}_t := n d_t$ and $\widehat{q}_t^k := n q_t^k$, the inner products become

$$r_{t,k} = \frac{1}{n} \sum_\alpha h_{t,\alpha}^{3,k} \widehat{d}_{t,\alpha}, \qquad \langle q_s^k, q_t^k \rangle = \frac{1}{n} \left(\frac{1}{n} \sum_\alpha \widehat{q}_{s,\alpha}^k \widehat{q}_{t,\alpha}^k \right). \tag{448}$$

These give the same Tensor-Program `Moment` operations as in the single-block backward pass:

$$\mathbb{E}\left[Z^{h_t^{3,k}} Z^{\widehat{d}_t}\right], \qquad \mathbb{E}\left[Z^{\widehat{q}_s^k} Z^{\widehat{q}_t^k}\right]. \tag{449}$$

The initial expert-weight terms are `MatMul` operations, while the router terms sum over the fixed number of experts. Hence the backward pass retains the same Tensor-Program operations and alignment structure as the single-block case. □

Remark G.2 (Forward and backward scaling of μP transformer modules). At any fixed training step, μP-parametrised attention and normalisation modules map $\Theta(1)$ inputs to $\Theta(1)$ outputs. Analogously, in the backward pass, they map incoming gradients of coordinate scale $\Theta(n^{-1})$ to input gradients of coordinate scale $\Theta(n^{-1})$. At fixed depth, the residual additions preserve these scales under the assumptions above.

We now use the alignment structure in Proposition G.1 and the scaling of μP modules in Remark G.2 to apply the single-block scaling analysis at every depth. In the next two propositions, all quantities are evaluated at a fixed training step t, where the subscript is suppressed for clarity.

Proposition G.3 (Forward scaling through depth)**.** *Suppose for μP modules and μP-MoE blocks, the coordinates of x^1 have scale $\Theta(1)$. Then, for every $\ell \in \{1, \dots, L\}$, the coordinates of u^ℓ, $\mathbf{h}^{1,(\ell)}$, $\mathbf{h}^{3,(\ell)}$, and $x^{\ell+1}$ have scale $\Theta(1)$. Thus each MoE block has the same forward interface scales as in the single-block analysis of Appendix B.2.*

Proof. We induct on depth ℓ. For $\ell = 1$, the input x^1 has coordinate scale $\Theta(1)$ by assumption. By Remark G.2, the μP attention and normalisation modules map $\Theta(1)$ inputs to $\Theta(1)$ outputs, so the first residual addition gives $u^1 = \Theta(1)$ coordinatewise. Normalisation therefore gives $\mathbf{h}^{1,(1)} = \Theta(1)$. By the alignment result in Proposition G.1 and the single-block μP-MoE scaling analysis in Appendix B.2, this implies $\mathbf{h}^{3,(1)} = \Theta(1)$, and hence $x^2 = u^1 + \mathbf{h}^{3,(1)} = \Theta(1)$. Both residual additions use the non-cancellation assumption.

Now suppose that $x^\ell = \Theta(1)$ coordinatewise. Applying the same argument to layer ℓ gives, successively, $u^\ell = \Theta(1)$, $\mathbf{h}^{1,(\ell)} = \Theta(1)$, $\mathbf{h}^{3,(\ell)} = \Theta(1)$, and $x^{\ell+1} = \Theta(1)$. This completes the induction. Since L is fixed independently of n, composition over depth introduces no additional width factor. □

Proposition G.4 (Backward scaling through depth)**.** *Under the hypotheses of Proposition G.3, suppose $\partial\mathcal{L}/\partial x_\alpha^{L+1} = \Theta(n^{-1})$. Then, for every $\ell \in \{1, \dots, L\}$,*

$$\frac{\partial \mathcal{L}}{\partial h_\alpha^{3,(\ell)}}, \quad \frac{\partial \mathcal{L}}{\partial h_\alpha^{1,(\ell)}}, \quad \frac{\partial \mathcal{L}}{\partial u_\alpha^{\ell}}, \quad \frac{\partial \mathcal{L}}{\partial x_\alpha^{\ell}} \quad \textit{all have scale } \Theta(n^{-1}). \tag{450}$$

In particular, each MoE block has the same backward interface scales as in the single-block analysis.

Proof. We induct backwards from layer L. The base case is the assumed terminal gradient $\partial\mathcal{L}/\partial x_\alpha^{L+1} = \Theta(n^{-1})$, supplied by a μP readout with a non-degenerate loss gradient. Suppose now that $\partial\mathcal{L}/\partial x_\alpha^{\ell+1} = \Theta(n^{-1})$. Since $\mathbf{h}^{3,(\ell)}$ feeds into the rest of the network only through the addition in Equation (440),

$$\frac{\partial \mathcal{L}}{\partial \mathbf{h}^{3,(\ell)}} = \frac{\partial \mathcal{L}}{\partial x^{\ell+1}}. \tag{451}$$

The alignment result in Proposition G.1 and the single-block backward scaling in Appendix B.2 therefore give $\partial\mathcal{L}/\partial h_\alpha^{1,(\ell)} = \Theta(n^{-1})$. Writing $J_F(z)$ for the Jacobian of a map F at z, the gradient at the preceding residual stream is

$$\frac{\partial \mathcal{L}}{\partial u_\alpha^{\ell}} = \underbrace{\frac{\partial \mathcal{L}}{\partial x_\alpha^{\ell+1}}}_{\text{residual: } \Theta(n^{-1})} + \underbrace{\left[J_{\mathrm{Norm}_2^\ell}(u^\ell)^\top \frac{\partial \mathcal{L}}{\partial \mathbf{h}^{1,(\ell)}} \right]_\alpha}_{\text{sublayer: } \Theta(n^{-1})} = \Theta(n^{-1}). \tag{452}$$

Here we use Remark G.2 and the non-cancellation assumption. Defining $A^\ell = \mathrm{Attn}^\ell \circ \mathrm{Norm}_1^\ell$, the same chain-rule decomposition at the attention residual gives

$$\frac{\partial \mathcal{L}}{\partial x_\alpha^{\ell}} = \frac{\partial \mathcal{L}}{\partial u_\alpha^{\ell}} + \left[J_{A^\ell}(x^\ell)^\top \frac{\partial \mathcal{L}}{\partial u^\ell} \right]_\alpha = \Theta(n^{-1}), \tag{453}$$

since μP attention and normalisation preserve this backward scale. This establishes the inductive step. The fixed depth ensures that only a width-independent number of residual contributions is accumulated. □

Proposition G.5 (Preservation of single-block scaling during training)**.** *Suppose that x_0^1 has coordinate scale $\Theta(1)$, and train the transformer with SGD or Adam using the corresponding μP and μP-MoE scalings. Assume that the readout loss gradients remain non-degenerate and of order one, and that the regularity and non-degeneracy conditions above hold over the training steps considered. Then, at every fixed training step t, each MoE block obeys the single-block coordinate-scaling rules for its forward pass, backward pass, and parameter and feature updates. In particular,*

$$h_{t,\alpha}^{1,(\ell)},\ h_{t,\alpha}^{3,(\ell)} = \Theta(1), \qquad \frac{\partial \mathcal{L}}{\partial h_{t,\alpha}^{3,(\ell)}},\ \frac{\partial \mathcal{L}}{\partial h_{t,\alpha}^{1,(\ell)}} = \Theta(n^{-1}) \qquad (\ell = 1, \dots, L). \tag{454}$$

Proof. We induct on the training step, retaining the scaling statements for all earlier steps in the induction hypothesis. At initialisation, the prescribed parameter scales and $x_0^1 = \Theta(1)$ allow us to apply Proposition G.3. The μP readout gives $\partial\mathcal{L}/\partial x_{0,\alpha}^{L+1} = \Theta(n^{-1})$, so Proposition G.4 supplies the single-block backward interface scales at every layer.

Suppose these scales hold through step t. By Proposition G.1, the parameter-gradient and update expressions retain the single-block alignment structure. Substituting their interface scales into the single-block formulas therefore leaves the width factors in the update rules unchanged. For SGD, multiplying by the learning rates in Table 1 gives the same parameter-update scales as in the single-block analysis. For Adam, the induction hypothesis also covers the gradient history: the scaled gradients $n^{d_j} g_s$, $s \le t$, entering each parameter group's update function Q_t^j have the same scales as in Appendix A.3. Using the learning-rate and epsilon scalings in Table 2 thus preserves the single-block Adam update scales as well.

The resulting parameter and propagated feature updates preserve the stable scaling regime by the single-block analysis in Appendix B.2 and the corresponding μP stability of the remaining transformer modules. In particular, the input embedding again supplies $x_{t+1}^1 = \Theta(1)$. Applying Propositions G.3 and G.4 at step $t+1$ recovers the same interface scales and closes the induction. □

Remark G.6 (Soft routing in MoE transformers). Under the assumptions of Proposition 4.2, including the absence of router bias or noise, a soft-routed MoE block within a multi-layer μP-MoE transformer does not have expert specialisation. The preceding propositions preserve the single-block scaling and alignment. The expert weights are initialised i.i.d. across experts, and the downstream network sees their outputs only through

$$\mathbf{h}_t^{3,(\ell)} = \sum_{k=1}^{m} \tilde{h}_{t,k}^{2,(\ell)} \mathbf{h}_t^{3,k,(\ell)}. \tag{455}$$

Permuting the experts together with their router rows leaves this aggregate unchanged, so it also leaves the incoming gradient $d_t^{(\ell)} := \nabla_{\mathbf{h}_t^{3,(\ell)}} \mathcal{L}_t$ unchanged. The same SGD or Adam rule is applied to every expert, so this joint symmetry is preserved during training. In particular, the joint law of an expert output and the shared incoming gradient is the same for every expert. Writing $\widehat{d}_t^{(\ell)} := n d_t^{(\ell)}$, the gating gradients therefore satisfy

$$\frac{\partial \mathcal{L}_t}{\partial \tilde{h}_{t,k}^{2,(\ell)}} = \frac{1}{n} \sum_{\alpha} h_{t,\alpha}^{3,k,(\ell)} \widehat{d}_{t,\alpha}^{(\ell)} \longrightarrow \mathbb{E}\left[Z^{h_t^{3,k,(\ell)}} Z^{\widehat{d}_t^{(\ell)}} \right] = C_t^{(\ell)}, \tag{456}$$

with the same deterministic limit for every k. Since the router logits start at zero, the induction in Proposition 4.2 applies: the scores remain symmetric and the experts remain identically distributed at every fixed training step. For softmax, the common gating gradient is annihilated by the softmax Jacobian; for sigmoid, the scores can evolve but remain equal across experts.

G.2 MoE Non-Linear Experts

Proposition G.7 (Non-linear experts preserve the parametrisation)**.** *Non-linear experts do not change the scaling parametrisation of μP-MoE. This holds for experts of fixed depth with coordinatewise activations ϕ, where ϕ and ϕ' are pseudo-Lipschitz, under the same non-degeneracy assumptions on the forward and backward limits as in the linear case.*

Proof. Take a two-layer expert

$$\mathbf{z}_t^k = \mathbf{U}_t^k \mathbf{h}_t^1, \qquad \mathbf{a}_t^k = \phi(\mathbf{z}_t^k), \qquad \mathbf{h}_t^{3,k} = \mathbf{V}_t^k \mathbf{a}_t^k, \tag{457}$$

with $\mathbf{U}_t^k, \mathbf{V}_t^k \in \mathbb{R}^{n \times n}$ scaled as expert matrices in μP-MoE (Tables 1 and 2). It suffices to show that the extra layer preserves the forward scale $\Theta(1)$ and backward scale $\Theta(n^{-1})$ used in our derivation.

Forward. The linear map gives $\mathbf{z}_t^k$ coordinate scale $\Theta(1)$. Since ϕ is an admissible nonlinearity, the Master Theorem gives

$$Z^{a_t^k} = \phi(Z^{z_t^k}), \qquad \frac{1}{n} \|\mathbf{a}_t^k\|_2^2 \longrightarrow \mathbb{E}\left[\phi(Z^{z_t^k})^2 \right]. \tag{458}$$

Pseudo-Lipschitz ϕ applied to the finite-moment limit $Z^{z_t^k}$ has finite moments, so $\mathbf{a}_t^k$ has coordinate scale $\mathcal{O}(1)$, and $\Theta(1)$ when the limiting second moment is non-zero. Hence $\mathbf{h}_t^{3,k} = \mathbf{V}_t^k \mathbf{a}_t^k$ has $\Theta(1)$ coordinates, exactly as in the linear case.

Backward. The linear map $\mathbf{V}_t^k$ sends an incoming gradient of coordinate scale $\Theta(n^{-1})$ to $\nabla_{\mathbf{a}_t^k}\mathcal{L}_t$ of the same scale. Coordinatewise,

$$\frac{\partial \mathcal{L}_t}{\partial z_{t,\alpha}^k} = \phi'(z_{t,\alpha}^k)\frac{\partial \mathcal{L}_t}{\partial a_{t,\alpha}^k}. \tag{459}$$

Write $\widehat{\mathbf{g}}_t^a := n\nabla_{\mathbf{a}_t^k}\mathcal{L}_t$ and $\widehat{\mathbf{g}}_t^z := n\nabla_{\mathbf{z}_t^k}\mathcal{L}_t$. Since ϕ' is pseudo-Lipschitz, the backward operation $(z, g) \mapsto \phi'(z)g$ is also admissible, with joint coordinate limit

$$Z^{\widehat{g}_t^z} = \phi'(Z^{z_t^k})Z^{\widehat{g}_t^a}. \tag{460}$$

This limit has finite moments, so multiplication by ϕ' introduces no power of n. Under the non-degeneracy assumption, $\nabla_{\mathbf{z}_t^k}\mathcal{L}_t$ therefore has coordinate scale $\Theta(n^{-1})$, and backpropagation through $\mathbf{U}_t^k$ preserves this scale at the expert input.

Both interface scales are unchanged. Each weight gradient is still an outer product of a $\Theta(n^{-1})$ backward factor and a $\Theta(1)$ forward factor, so the same SGD and Adam expert-layer scalings apply. The argument iterates unchanged through any fixed number of expert layers. □

Remark G.8 (Soft routing with non-linear experts). Proposition 4.2 also holds for the non-linear experts in Proposition G.7, with the same architecture and activation for every expert, i.i.d. initial expert weights, and no router bias or noise. The gating gradient depends on each expert only through its output:

$$\frac{\partial \mathcal{L}_t}{\partial \tilde{h}_{t,k}^2} = \left\langle \mathbf{h}_t^{3,k}, \nabla_{\mathbf{h}_t^3}\mathcal{L}_t \right\rangle. \tag{461}$$

Thus it suffices to preserve the joint symmetry of the expert outputs and the shared incoming gradient. At initialisation, the shared input and identically distributed $\mathbf{U}_0^k$ give identical preactivation laws across experts. The Master Theorem gives $Z^{a_0^k} = \phi(Z^{z_0^k})$, so the activation laws are also identical; applying the independently initialised $\mathbf{V}_0^k$ then gives identical expert-output laws. The router still converges to symmetric scores at initialisation.

For the inductive step, suppose the expert histories are jointly symmetric through step t. As in Remark G.6, the limiting gating gradients are then identical across experts, so the router scores remain symmetric. Writing $q_t^k := \tilde{h}_{t,k}^2 \nabla_{\mathbf{h}_t^3}\mathcal{L}_t$, backpropagation through each expert gives

$$\nabla_{\mathbf{a}_t^k}\mathcal{L}_t = (\mathbf{V}_t^k)^\top q_t^k, \qquad \nabla_{\mathbf{z}_t^k}\mathcal{L}_t = \phi'(\mathbf{z}_t^k) \odot \nabla_{\mathbf{a}_t^k}\mathcal{L}_t. \tag{462}$$

Since ϕ, ϕ', and the SGD or Adam update rule are identical across experts, both expert layers preserve this joint symmetry at the next step. Hence the expert outputs remain identically distributed, and the rest of the proof of Proposition 4.2 is unchanged. The same lack of specialisation follows for both softmax and sigmoid routing, with a vanishing limiting router gradient in the softmax case.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: The abstract and introduction state the scope of fixed-expert infinite-width limits for soft and Top-K MoEs trained with SGD/Adam, and the claimed failures and empirical validations match the later theoretical and experimental sections.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: All the claims made are within the scope of our study, including the fixed-number-of-experts regime, fixed other scaling dimensions. These are listed in Setting 2.3. We do not make any assumptions on the data distribution (similar to all TP works).

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

Answer: [Yes]

Justification: The paper includes theoretical results with stated settings and an appendix roadmap covering Tensor Program derivations, soft-routing proofs, Top-K proofs, and infinite-width limits.

Guidelines:

- The answer [N/A] means that the paper does not include theoretical results.
- All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
- All assumptions should be clearly stated or referenced in the statement of any theorems.
- The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
- Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
- Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

Answer: [Yes]

Justification: The experimental details for the main figures in the paper are listed in Appendix F.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
- If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
- Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
- While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
  (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
  (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
  (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).
  (d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in

some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We will publish our code. We include it in supplementary material.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: These are in Appendix F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Our paper is mainly theoretical but for our larger scale experiemnt, we run over multiple runs with different seeds.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.

- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: Compute used are listed in Appendix F.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: The work is theoretical and uses standard public benchmark datasets for validation; no human-subject, private-data, or high-risk deployment setting is described.

   Guidelines:

   - The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [N/A]

    Justification: The paper focuses on technical MoE scaling dynamics and does not broader impacts except for those with advancement of ML systems.

    Guidelines:

- The answer [N/A] means that there is no societal impact of the work performed.
- If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release a high-risk model, image generator, scraped dataset, or other artifact requiring misuse safeguards; the experiments use benchmark datasets and an open source Mixtral-style model.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
    - Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
    - We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: We use open source models and datasets that have been cited.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.

- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [N/A]

    Justification: The paper does not release new datasets, code packages, or model checkpoints as new assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or research with human subjects; the experiments are benchmark/model-training experiments.

    Guidelines:

    - The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [N/A]

    Justification: The paper does not involve crowdsourcing or human-subject research, so IRB approval or equivalent review is not applicable.

    Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

    Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

    Answer: [N/A]

    Justification: The core method is Tensor Program/infinite-width analysis of MoE dynamics, not an important, original, or non-standard use of LLMs.

    Guidelines:

    - The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
    - Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.