MomentumSMoE: Integrating Momentum into Sparse Mixture of Experts

Rachel S.Y. Teo
Department of Mathematics
National University of Singapore
rachel.tsy@u.nus.edu

Tan M. Nguyen
Department of Mathematics
National University of Singapore
tanmn@nus.edu.sg

Abstract

Sparse Mixture of Experts (SMoE) has become the key to unlocking unparalleled scalability in deep learning. SMoE has the potential to exponentially increase in parameter count while maintaining the efficiency of the model by only activating a small subset of these parameters for a given sample. However, it has been observed that SMoE suffers from unstable training and has difficulty adapting to new distributions, leading to the model's lack of robustness to data contamination. To overcome these limitations, we first establish a connection between the dynamics of the expert representations in SMoEs and gradient descent on a multi-objective optimization problem. Leveraging our framework, we then integrate momentum into SMoE and propose a new family of SMoEs, named MomentumSMoE. We theoretically prove and numerically demonstrate that MomentumSMoE is more stable and robust than SMoE. In particular, we verify the advantages of MomentumSMoE over SMoE on a variety of practical tasks including ImageNet-1K object recognition and WikiText-103 language modeling. We demonstrate the applicability of MomentumSMoE to many types of SMoE models, including those in the Sparse MoE model for vision (V-MoE) and the Generalist Language Model (GLaM). We also show that other advanced momentum-based optimization methods, such as Adam, can be easily incorporated into the MomentumSMoE framework for designing new SMoE models with even better performance, almost negligible additional computation cost, and simple implementations. The code is publicly available at https://github.com/rachtsy/MomentumSMoE.

1 Introduction

Scaling up deep models has demonstrated significant potential for enhancing the model's performance on a wide range of cognitive and machine learning tasks, ranging from large language model pretraining [13, 58, 59, 30, 5, 51, 70] and vision understanding [15, 2, 3, 39, 1, 40] to reinforcement learning [6, 28] and scientific applications [66, 74]. However, increasing the model's size requires a higher computational budget, which can be often challenging to meet. As a result, Sparse Mixture of Experts (SMoE) has been recently studied as an efficient approach to effectively scale up deep models. By modularizing the network and activating only subsets of experts for each input, SMoE maintains constant computational costs while increasing model complexity. This approach enables the development of billion-parameter models and achieves significant success in various applications, including machine translation [35], image classification [61], and speech recognition [34].

1.1 Sparse Mixture of Experts

A MoE replaces a component in the layer of the model, for example, a feed-forward or convolutional layer, by a set of networks termed experts. This approach largely scales up the model but increases

Please correspond to: rachel.tsy@u.nus.edu

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

the computational cost. A SMoE inherits the extended model capacity from MoE but preserves the computational overhead by taking advantage of conditional computation. In particular, a SMoE consists of a router and E expert networks, u_i, $i = 1, 2, \ldots, E$. For each input token $\boldsymbol{x}_t \in \mathbb{R}^D$ at layer t, the SMoE's router computes the affinity scores between $\boldsymbol{x}_t$ and each expert as $g_i(\boldsymbol{x}_t)$, $i = 1, 2, \ldots, E$. In practice, we often choose the router $g(\boldsymbol{x}_t) = [g_1(\boldsymbol{x}_t), g_2(\boldsymbol{x}_t), \ldots, g_E(\boldsymbol{x}_t)]^\top = \boldsymbol{W}\boldsymbol{x} + \boldsymbol{b}$, where $\boldsymbol{W} \in \mathbb{R}^{E\times D}$ and $\boldsymbol{b} \in \mathbb{R}^E$. Then, a sparse gating function TopK is applied to select only K experts with the greatest affinity scores. Here, we define the TopK function as:

$$\mathrm{TopK}(g_i) := \begin{cases} g_i, & \text{if } g_i \text{ is in the } K \text{ largest elements of } g \\ -\infty, & \text{otherwise.} \end{cases} \tag{1}$$

The outputs from K expert networks chosen by the router are then linearly combined as

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \sum_{i=1}^{E} \mathrm{softmax}(\mathrm{TopK}(g_i(\boldsymbol{x}_t))u_i(\boldsymbol{x}_t) = \boldsymbol{x}_t + u(\boldsymbol{x}_t), \tag{2}$$

where $\mathrm{softmax}(g_i) := \exp(g_i)/\sum_{j=1}^{E}\exp(g_j)$. We often set $K = 2$, i.e., top-2 routing, as this configuration has been shown to provide the best trade-off between training efficiency and testing performance [35, 16, 76].

Limitations of SMoE. Despite their remarkable success, SMoE suffers from unstable training [11, 78] and difficulty in adapting to new distributions, leading to the model's lack of robustness to data contamination [55, 75]. These limitations impede the application of SMoE to many important large-scale tasks.

1.2 Contribution

In this paper, we explore the role of the residual connection in SMoE and show that simple modifications of this residual connection can help enhance the stability and robustness of SMoE. In particular, we develop a gradient descent (GD) analogy of the SMoE, showing that the dynamics of the expert representations in SMoE is associated with a gradient descent step toward the optimal solution of a multi-objective optimization problem. We then propose to integrate heavy-ball momentum into the dynamics of SMoE, which results in the Momentum Sparse Mixture-of-Experts (MomentumSMoE). At the core of MomentumSMoE is the use of momentum to stabilize and robustify the model. The architecture of MomentumSMoE is depicted in Fig. 1. MomentumSMoE can be extended beyond heavy-ball momentum to integrate well with other advanced momentum-accelerated methods such as AdamW [33, 42] and Robust Momentum [10]. Our contribution is three-fold:

1. We incorporate heavy-ball momentum in SMoE to improve the model's stability and robustness.
2. We theoretically prove that the spectrum of MomentumSMoE is better-structured than SMoE, leading to the model's stability enhancement.
3. We show that the design principle of MomentumSMoE can be generalized to other advanced momentum-based optimization methods, proposing AdamSMoE and Robust MomentumSMoE.

Our experimental results validate that our momentum-based SMoEs improve over the baseline SMoE in terms of accuracy and robustness on a variety of practical benchmarks, including WikiText-103 language modeling and ImageNet-1K object recognition. We also empirically demonstrate that our momentum-based design framework is universally applicable to many existing SMoE models, including the Sparse MoE model for vision (V-MoE) [61] and the Generalist Language Model (GLaM) [16], just by changing a few lines of the baseline SMoE code.

Organization. We structure this paper as follows: In Section 2, we establish the connection between SMoE and gradient descent and derive our MomentumSMoE. In Section 3, we theoretically prove the stability advantage of MomentumSMoE over SMoE. In Section 4, we introduce AdamSMoE and Robust MomentumSMoE. In Section 5, we present our experimental results to justify the advantages of our momentum-based SMoE models over the traditional SMoE and other SMoE baselines. In Section 6, we empirically analyze our MomentumSMoE. We discuss related works in Section 7. The paper ends with concluding remarks. More experimental details are provided in the Appendix.

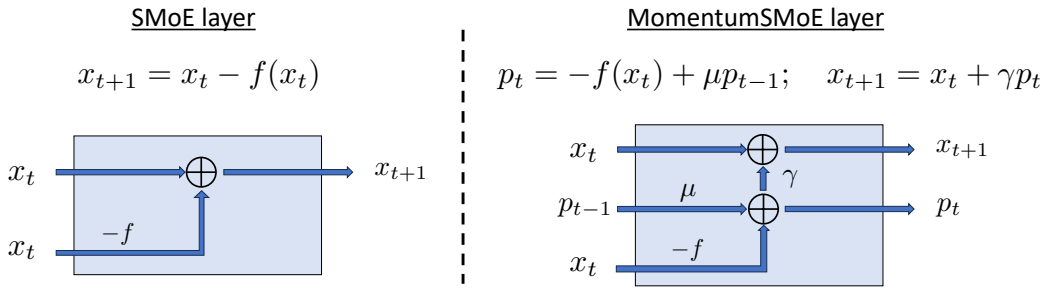

Figure 1: Illustration of SMoE (Left) and MomentumSMoE layer (Right). We establish a connection between Multiple-Gradient Descent and SMoE to introduce momentum into the model, leading to better accuracy, enhanced robustness, and faster convergence.

2 Momentum Sparse Mixture of Experts

2.1 Background: Multiple-Gradient Descent Algorithm for Multi-objective Optimization

A multi-objective optimization problem comprises of the concurrent optimization of E objective functions, $F_i(x)$, $i = 1, 2, \ldots, E$, which might be formulated as the following minimization problem

$$\min_{x \in D} F(x) := \sum_{i=1}^{E} c_i F_i(x) \tag{3}$$

where D is the feasible region and $c_i \in \mathbb{R}$ are weights representing the importance of each objective function. The optimal solution to the multi-objective optimization problem above is a Pareto-optimal point such that there is no other solution that can decrease at least one of the objective functions without increasing any other objective functions. [12] shows that a necessary condition for a solution to be Pareto-optimal is for it to be Pareto-stationary, which is defined as:

Definition 1 (Pareto-stationary) *Let x be in the interior of the feasible region, D, in which the E objective functions, F_i, are smooth, and $f_i(x) = \nabla_x F_i(x)$ be the local gradients for $i = 1, \ldots, E$. x is said to be Pareto-stationary if there exists a vector $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_E]^\top \in \mathbb{R}^E$ such that $\alpha_i \geq 0$, $\sum_{i=1}^{E} \alpha_i = 1$ and $\sum_{i=1}^{E} \alpha_i f_i(x) = 0$. That is, there exists a convex combination of the gradient-vectors $f_i(x)$ that is equal to 0.*

Therefore, it would be intuitive to extend the steepest descent algorithm to a multi-objective setting by finding a descent direction, that is common to all objectives, in the convex hull of the normalized local gradients $\tilde{f}_i(x) = f_i(x)/\|f_i(x)\|$. We denote such a set as $\bar{U} = \{v \in \mathbb{R}^N | v = \sum_{i=1}^{E} \alpha_i \tilde{f}_i(x); \alpha_i \geq 0, \forall i; \sum_{i=1}^{E} \alpha_i = 1\}$. Indeed, [12] developed the Multiple-Gradient Descent Algorithm (MGDA) from such an understanding, proving that there does exist such a descent direction in $\bar{U}$, which is the direction with the smallest norm in the set. Then, the update rule of MGDA is

$$x_{t+1} = x_t - \gamma \sum_{i=1}^{E} \alpha_i^* \tilde{f}_i(x_t) \tag{4}$$

where $\alpha^* = (\alpha_1^*, \ldots, \alpha_E^*)$ minimizes $\{\|v\| | v \in \bar{U}\}$.

2.2 Background: Momentum Acceleration for Gradient-Based Optimization

Among the simplest learning algorithms is gradient descent, also termed the steepest descent method. It typically starts with an objective function $F(x)$ whose minima we aim to find by modifying our iterate x_t at each time step t through its gradient $f(x_t) = \nabla_x F(x_t)$, scaled by a step size $\gamma > 0$:

$$x_{t+1} = x_t - \gamma \nabla_x F(x_t) = x_t - \gamma f(x_t). \tag{5}$$

However, following this update rule might result in slow convergence. A classical acceleration method to speed up the steepest descent, known as the heavy-ball method [53, 67], includes a momentum term in the algorithm. This takes the form of

$$p_t = -f(x_t) + \mu p_{t-1}; \quad x_{t+1} = x_t + \gamma p_t, \tag{6}$$

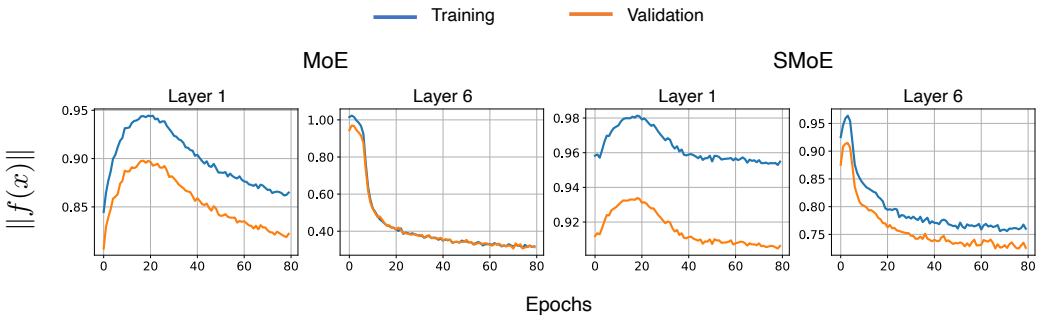

Figure 2: Average output norms at layers 1 and 6 of the MoE/SMoE during 80 training epochs on WikiText-103.

Figure 3: Average output norm at each layer across 1K train/validation samples of the (S)MoE trained on WikiText-103.

where $\gamma > 0$ is the step size, and $\mu \geq 0$ is the momentum constant. Eqn. 6 can then be rewritten as:

$$x_{t+1} = x_t + \gamma[-f(x_t) + \mu p_{t-1}] = x_t - \gamma f(x_t) + \mu(x_t - x_{t-1}). \tag{7}$$

By incorporating the past gradients in each update, the descent path can become smoother with fewer oscillations, resulting in a faster convergence [53, 21, 71].

2.3 (S)MoE as (Stochastic) Gradient Descent on Multi-objective Optimization and MomentumSMoE

We will now consider the SMoE model from the multi-objective optimization perspective.

MoE as Gradient Descent. In viewing each expert in an SMoE as specializing in optimizing an objective function, we are going to establish a connection between MoE and GD, and further leverage momentum to enhance MoE and SMoE. We rewrite Eqn. 2 of MoE as follows:

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \gamma \sum_{i=1}^{E} \text{softmax}(g_i(\boldsymbol{x}_t))[-u_i(\boldsymbol{x}_t)]. \tag{8}$$

If we regard $-u_i(\boldsymbol{x}_t)$ as the local "gradient" $\nabla_{\boldsymbol{x}} F_i(\boldsymbol{x}_t)$ at the t-th iteration and $\text{softmax}(g_i(\boldsymbol{x}_t))$ as to be learned to approximate α_i^*, then we can consider the MoE in Eqn. 2 and 8 as the dynamical system which updates $\boldsymbol{x}_t$ using the MGDA to minimize the multi-objective optimization in Eqn. 3.

SMoE as Stochastic Gradient Descent. Given the analogy between MoE and GD, SMoE can be interpreted as a stochastic version of MoE, which corresponds to an SGD algorithm applied to the multi-objective optimization problem in Eqn. 3. SMoE is then reformulated as:

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \gamma \sum_{i=1}^{K} \text{softmax}(\text{TopK}(g_i(\boldsymbol{x}_t)))[-u_i(\boldsymbol{x}_t)] = \boldsymbol{x}_t - \gamma f(\boldsymbol{x}_t). \tag{9}$$

Here, $-f(\boldsymbol{x}_t) = \gamma \sum_{i=1}^{K} \text{softmax}(\text{TopK}(g_i(\boldsymbol{x}_t)))u_i(\boldsymbol{x}_t)$ is the SMoE output.

Empirical Evidences for the Gradient Descent Analogy of (S)MoE. We provide empirical justification for the connection between (S)MoE and (S)GD in Fig. 2 and 3.

Gradient norm $\|f(\boldsymbol{x}_t)\|$ decreases when t increases: As shown in Eqn. 8 and 9 above, the norm of the MoE and SMoE output corresponds to the gradient norm $\|f(\boldsymbol{x}_t)\| = \nabla_{\boldsymbol{x}} F(\boldsymbol{x}_t)$, respectively. It is expected that this gradient norm decreases when t increases or equivalently when the number of layers in an (S)MoE model increases. Fig. 3 confirms this expectation by showing that the norm of the (S)MoE output decreases over layers in a 6-layer (S)MoE model trained on the WikiText-103 language modeling task. At the last layer, the norm increases might be due to overshooting, a common phenomenon that can occur when using gradient descent.

Gradient norm $\|f(\boldsymbol{x}_t)\|$ at each layer t decreases during training: According to the update rule of MGDA in Eqn. 7, the coefficient α_i^* minimizes the norm $\|\sum_{i=1}^E \alpha_i \tilde{f}_i\|$. In Eqn. 8 and 9, as discussed above, softmax$(g_i(\boldsymbol{x}_t))$ and softmax(TopK$(g_i(\boldsymbol{x}_t))$) try to learn α_i^*, respectively. Thus, it is expected that these two terms learn to reduce the corresponding $\|\sum_{i=1}^E \alpha_i \tilde{f}_i\|$ in Eqn. 8 and 9, which is the norm of the SMoE output at layer t. Fig. 2 verifies this expectation by showing that each MoE and SMoE layer learns to reduce its output norm during training, suggesting that the routers softmax$(g_i(\boldsymbol{x}_t))$ and softmax(TopK$(g_i(\boldsymbol{x}_t))$) learn to approximate α_i^*. We provide the full plots for all layers in Appendix C, Fig. 6 and 7.

2.4 MomentumSMoE

We propose the new *MomentumSMoE* layer, depicted in Fig. 1, to accelerate the dynamics of 8, which is principled by the accelerated gradient descent theory (see Section 2.2):

$$\boldsymbol{p}_t = -f(\boldsymbol{x}_t) + \mu \boldsymbol{p}_{t-1}; \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \gamma \boldsymbol{p}_t, \tag{10}$$

where $\mu \geq 0$ and $\gamma > 0$ are hyperparameters corresponding to the momentum coefficient and step size in the momentum-accelerated GD, respectively. The formulation of MomentumSMoE can be applied to MoE to derive the MomentumMoE.

3 Stability Analysis: MomentumSMoE vs. SMoE

In this section, we demonstrate the theoretical advantages of MomentumSMoE over SMoE. In particular, we show that the spectrum of MomentumSMoE is better-structured than that of SMoE, thus MomentumSMoE is more stable than SMoE. We rewrite MomentumSMoE using the equivalent form of momentum acceleration given in Eqn. 7 as follows:

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \gamma f(\boldsymbol{x}_t) + \mu(\boldsymbol{x}_t - \boldsymbol{x}_{t-1}). \tag{11}$$

Taking inspiration from [57], we then expand $f(\boldsymbol{x}_t)$ around the Pareto-stationary solution $\boldsymbol{x}^*$ at which $f(\boldsymbol{x}^*) = 0$ (see Definition 1) using Taylor expansion to obtain an approximation of $f(\boldsymbol{x}_t)$:

$$f(\boldsymbol{x}_t) \approx f(\boldsymbol{x}^*) + \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*)(\boldsymbol{x}_t - \boldsymbol{x}^*) = \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*)(\boldsymbol{x}_t - \boldsymbol{x}^*). \tag{12}$$

Substituting the Taylor expansion of $f(\boldsymbol{x}_t)$ in Eqn. 12 into Eqn. 11, we attain

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \gamma \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*)(\boldsymbol{x}_t - \boldsymbol{x}^*) + \mu(\boldsymbol{x}_t - \boldsymbol{x}_{t-1}).$$

Without loss of generality, we further let $\boldsymbol{x}^* = 0$ as we can always replace $\boldsymbol{x}_t$ with $\boldsymbol{x}_t + \boldsymbol{x}^*$. The formula of MomentumSMoE can then be simplified as

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \gamma \nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*)\boldsymbol{x}_t + \mu(\boldsymbol{x}_t - \boldsymbol{x}_{t-1}). \tag{13}$$

Suppose that $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*)$ does not have any defective eigenvalues and hence is diagonalizable. Then, $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*) = \boldsymbol{Q}\boldsymbol{\Sigma}\boldsymbol{Q}^{-1}$, for some invertible matrix $\boldsymbol{Q}$ and the diagonal matrix $\boldsymbol{\Sigma}$ with diagonal entries being the eigenvalues $\sigma(n)$, $n = 1, 2, \ldots, N$, of $\nabla_{\boldsymbol{x}} f(\boldsymbol{x}^*)$. We can then rewrite Eqn. 13 as

$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \gamma \boldsymbol{\Sigma} \boldsymbol{x}_t + \mu(\boldsymbol{x}_t - \boldsymbol{x}_{t-1}). \tag{14}$$

Since we have decoupled the N features in $\boldsymbol{x}_t$, we can consider each feature $\boldsymbol{x}_t(n)$, separately. Introducing a dummy equation $\boldsymbol{x}_t = \boldsymbol{x}_t$, we rewrite Eqn. 14 as follows:

$$\begin{pmatrix} \boldsymbol{x}_t(n) \\ \boldsymbol{x}_{t+1}(n) \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\mu & (1+\mu) - \gamma\sigma(n) \end{pmatrix} \begin{pmatrix} \boldsymbol{x}_{t-1}(n) \\ \boldsymbol{x}_t(n) \end{pmatrix} = \boldsymbol{A} \begin{pmatrix} \boldsymbol{x}_{t-1}(n) \\ \boldsymbol{x}_t(n) \end{pmatrix} \tag{15}$$

The convergence of $\boldsymbol{x}_t(n)$ then depends on the eigenvalues $\lambda_1(\boldsymbol{A})$ and $\lambda_2(\boldsymbol{A})$ of $\boldsymbol{A}$. In particular, we require $\max\{|\lambda_1(\boldsymbol{A})|, |\lambda_2(\boldsymbol{A})|\} < 1$. It should be noted that omitting the momentum parameter, i.e., $\mu = 0$, recovers the standard, unaccelerated SMoE layer.

Lemma 1 *Given the matrix* $\boldsymbol{A} = \begin{pmatrix} 0 & 1 \\ -\mu & (1+\mu) - \gamma\sigma(n) \end{pmatrix}$ *and* $\lambda_1(\boldsymbol{A})$, $\lambda_2(\boldsymbol{A})$ *are eigenvalues of* A, $\max\{|\lambda_1(\boldsymbol{A})|, |\lambda_2(\boldsymbol{A})|\} < 1$ *if and only if* $\mu \in (-1, 1)$ *and* $\gamma\sigma(n) \in (0, 2 + 2\mu)$.

Proposition 1 (Convergence of MomentumSMoE) *The MomentumSMoE defined in Eqn. 10 converges if and only if* $\mu \in (-1, 1)$ *and* $\gamma\sigma(n) \in (0, 2 + 2\mu)$.

The proofs of the results above are provided in Appendix A. It is worth noting that in both the MomentumSMoE and standard SMoE, the convergence of $\boldsymbol{x}_t$ depends on the eigenvalues of the Jacobian $\nabla_{\boldsymbol{x}} f$ of the SMoE layer. Since the step size $\gamma > 0$, we require that $\nabla_{\boldsymbol{x}} f$ to be positive definite for its eigenvalues $\sigma(n)$ to be positive. Furthermore, even though among the convergence conditions of MomentumSMoE is that $\mu \in (-1, 1)$, in practice, μ is chosen to be positive.

Proposition 1 implies that the spectrum of MomentumSMoE is better-structured than that of SMoE. Thus, MomentumSMoE is more stable than SMoE. We summarize this finding in Corollary 1 below.

Corollary 1 (MomentumSMoE is more stable than SMoE) *Without momentum,* $\mu = 0$*, the range of values that* $\gamma\sigma(n)$ *can take for the system to be stable is limited to* $0 < \gamma\sigma(n) < 2$*. The addition of momentum expands this margin, almost doubling it, providing a larger parameter range for the network to converge stably to a good output in the forward pass.*

4 Beyond Heavy-ball Momentum: AdamSMoE and Robust MomentumSMoE

In addition to heavy-ball momentum, there are several advanced momentum-based algorithms in optimization that can be utilized for SMoE design. In this subsection, we propose two additional variants of MomentumSMoE, AdamSMoE and Robust MomentumSMoE, which are derived from the AdamW [33, 42] and Robust Momentum [10], respectively.

4.1 Adam Sparse Mixture of Experts (AdamSMoE)

Adam [33] accelerates the gradient dynamics by utilizing the moving average of historical gradients and element-wise squared gradients. Adam with a decoupled weight decay regularization (AdamW) is more commonly used in practice thanks to its better generalization over Adam. We employ AdamW to derive the *AdamSMoE* as follows:

$$\boldsymbol{p}_t = \mu\boldsymbol{p}_{t-1} + (1-\mu)[-f(\boldsymbol{x}_t)]; \quad \boldsymbol{m}_t = \beta\boldsymbol{m}_{t-1} + (1-\beta)f(\boldsymbol{x}_t) \odot f(\boldsymbol{x}_t)$$
$$\boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \frac{\gamma}{\sqrt{\boldsymbol{m}_t} + \epsilon}\boldsymbol{p}_t - \kappa\boldsymbol{x}_t$$

where ϵ is a small constant to prevent numerical instability, κ the weight decay parameter, γ the step size, and μ and β are the decay parameters for the moment estimates.

4.2 Robust Momentum Sparse Mixture of Experts (Robust MomentumSMoE)

Deep learning models, including SMoE, are known to not be robust to distribution shifts and data distortions [60, 19, 14]. Utilizing the connection between (S)MoE and (S)GD in Section 2.3, we develop the new *Robust MomentumSMoE* from the Robust Momentum Method [10].

The Robust Momentum Method proposed by [10] has the following update rule

$$y_t = x_t + \alpha(x_t - x_{t-1}); \quad x_{t+1} = x_t - \gamma f(y_t) + \mu(x_t - x_{t-1}), \tag{16}$$

where γ, μ and α are parameterized by an additional hyperparameter p as follows:

$$\gamma = \frac{k(1-p)^2(1+p)}{L}; \quad \mu = \frac{kp^3}{k-1}; \quad \alpha = \frac{p^3}{(k-1)(1-p)^2(1+p)}. \tag{17}$$

Here, $k = L/m$ is a condition ratio of the objective function assuming that it is m-strongly convex and L-smooth. Compared with the heavy-ball momentum in Eqn. 7, there is an additional variable y_t that can be interpreted as a feedback signal to steer the x_t toward a robust solution. The parameters γ, μ, and α are designed such that the new system is robust.

Table 1: Perplexity (PPL) of momentum-based SMoE vs. SMoE baseline on clean/attacked WikiText-103.

Model/Metric	Parameters	Clean WikiText-103		Attacked WikiText-103	
		Valid PPL ↓	Test PPL ↓	Valid PPL ↓	Test PPL ↓
SMoE-medium (baseline)	216M	33.76	35.55	42.24	44.19
MomentumSMoE-medium	216M	32.29	33.46	40.94	42.33
AdamSMoE-medium	216M	**31.59**	**33.25**	**39.27**	**41.11**
SMoE-large (baseline)	388M	29.31	30.33	36.77	37.83
MomentumSMoE-large	388M	**27.58**	**29.03**	**35.21**	**36.78**
GLaM-medium (baseline)	220M	36.37	37.71	45.83	47.61
MomentumGLaM-medium	220M	33.87	35.29	42.15	43.64
AdamGLaM-medium	220M	**32.99**	**34.32**	**41.09**	**42.81**

We incorporate the Robust Momentum Method above in our SMoE optimization framework developed in Section 2.3 and formulate the novel Robust MomentumSMoE as follows:

$$\boldsymbol{y}_t = \boldsymbol{x}_t + \alpha(\boldsymbol{x}_t - \boldsymbol{x}_{t-1}); \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \gamma f(\boldsymbol{y}_t) + \mu(\boldsymbol{x}_t - \boldsymbol{x}_{t-1}), \tag{18}$$

where γ, μ and α are as defined in Eqn. 17, and $-f(\boldsymbol{y}_t) = \gamma \sum_{i=1}^{K} \text{softmax}(\text{TopK}(g_i(\boldsymbol{y}_t)))u_i(\boldsymbol{y}_t)$ is the SMoE output given the input $\boldsymbol{y}_t$. Equivalently, at each layer t, we update the input $\boldsymbol{x}_t$ and momentum vector $\boldsymbol{p}_t$ as

$$\boldsymbol{y}_t = \boldsymbol{x}_t + \alpha\gamma\boldsymbol{p}_{t-1}; \quad \boldsymbol{p}_t = -f(\boldsymbol{y}_t) + \mu\boldsymbol{p}_{t-1}; \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \gamma\boldsymbol{p}_t.$$

Remark 1 *We provide an interpretation of robust momentum in Appendix B.*

5 Experimental Results

In this section, we numerically justify the advantages of our momentum-based SMoE over the baseline SMoE on both WikiText-103 language modeling and ImageNet-1k object recognition tasks. We aim to show that: (i) MomentumSMoE improves model performance across both language and vision tasks; (ii) AdamSMoE significantly outperforms the baseline and accelerates convergence in language models, even surpassing MomentumSMoE; (iii) Robust MomentumSMoE is highly effective at improving robustness of vision models to data corruption; (iv) MomentumSMoE is universal and can be easily integrated into many state-of-the-art SMoE and MoE models.

Throughout the experiments, we compare our momentum-based SMoE with the baseline SMoE of the same configuration, replacing SMoE layers with our momentum-based SMoE. For Adam-based SMoE models, we use AdamSMoE in the first layer of the model and MomentumSMoE for the subsequent layers. We provide an explanation for this implementation in Appendix D.1. We find that implementing AdamSMoE in the first layer is enough to significantly improve the model's overall performance and accelerate its convergence. Our results are averaged over 5 runs. Details on datasets, models, and training are provided in Appendix D, along with Table 4 summarizing the momentum methods implemented on SMoE/MoE models for different tasks in our experiments. More results can be found in Appendix E. All experiments are conducted on a server with 8 A100 GPUs.

5.1 WikiText-103 Language Modeling

We use the Switch Transformer [18], referred to as SMoE in our tables and figures below, and GLaM [16] baselines. We consider 2 configurations: medium (6 layers) and large (12 layers). We report the perplexity (PPL) of MomentumSMoE and AdamSMoE in comparison with the baseline SMoE on word-level WikiText-103 validation and test datasets for both model sizes in Table 1. We also include experiments on the medium-sized GLaM. A lower PPL indicates a better performance of the model. To further demonstrate the robustness of our method, we test the models on word swap attacked WikiText-103 data and present their results. Across all metrics, AdamSMoE and AdamGLaM outperform the baseline by a significant margin, verifying the strength of our method. Additionally, in Figure 4(Left), we provide the training and validation PPL during the first 5 training epochs of MomentumSMoE and AdamSMoE compared to the baseline SMoE to illustrate the accelerated convergence of our momentum-based models.

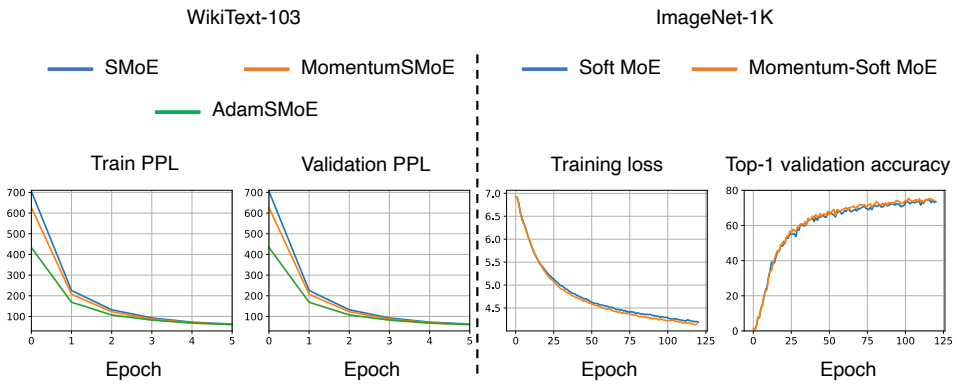

Figure 4: **Left:** WikiText-103 train/validation perplexity (PPL) curves during the first 5 training epochs for MomentumSMoE, AdamSMoE, and SMoE. AdamSMoE has significantly faster convergence compared to SMoE. **Right:** Training loss/top-1 accuracy (%) of Momentum-Soft MoE vs. Soft MoE baseline on ImageNet-1K across 120 epochs of training. Momentum-Soft MoE has faster convergence and improved accuracy.

Table 2: Top-1 accuracy (%) and mean corruption error (mCE) of MomentumV-MoE and Robust MomentumV-MoE vs. the V-MoE baseline on ImageNet-1K and popular robustness benchmarks for image classification.

Model	Params	Train IN-1K		Valid IN-1K		IN-R	IN-A	IN-C	
		Top-1 ↑	Top-5 ↑	Top-1 ↑	Top-5 ↑	Top-1 ↑	Top-1 ↑	Top-1 ↑	mCE ↓
V-MoE (baseline)	297M	76.49	**92.27**	73.16	**90.42**	36.10	5.25	46.98	67.14
MomentumV-MoE	297M	**76.92**	92.19	**73.26**	90.30	37.45	**6.48**	48.11	65.77
Robust MomentumV-MoE	297M	76.66	**92.27**	73.20	90.36	**37.57**	6.37	**48.82**	**64.92**

An important advantage of MomentumSMoE is its simplicity, which allows easy implementation with negligible computational overhead. We provide a comparison of the run time/sample, memory, number of parameters, and computation time between models in Table 11 and 12 in Appendix E.5.

5.2 ImageNet-1K Object Recognition Task

In this section, we investigate our momentum-based models on two popular vision models, Vision MoE (V-MoE) [61] and Soft MoE [56] on the ImagetNet-1K (IN-1K) object recognition task. We focus on i) improving the robustness of V-MoE using Robust MomentumSMoE (18) and ii) demonstrating that our momentum method is not limited to sparse models but can be generalized to MoE models such as Soft MoE. To benchmark robustness to data corruptions, we use the standard datasets, ImageNet-R (IN-R) [24], ImageNet-A (IN-A) [26], and ImageNet-C (IN-C) [25].

Vision Mixture of Experts (V-MoE). We use a V-MoE (small) model as the baseline. This V-MoE consists of 8 Vision Transformer (ViT) blocks [15] with every odd block's MLP being replaced by a SMoE layer. In Table 2, we provide the top-1 accuracy (%) on the training and validation set of IN-1K, IN-R, IN-A, and IN-C, as well as the mean Corruption Error (mCE) for IN-C. While MomentumV-MoE and Robust MomentumV-MoE have marginal gains on clean IN-1K data, we see significant improvement on IN-R, IN-A, and IN-C with at least a 1% increase in accuracy across these metrics. Specifically, Robust MomentumV-MoE has an almost 2% increase and 2 mCE decrease on IN-C, justifying the advantage of our method. Furthermore, we visualize the top-1 accuracy and mCE across increasing severity of two corruption types in Fig. 14 in Appendix E.3 to illustrate the increasing effectiveness of our method with escalating data corruption. The results of Robust MomentumSMoE on WikiText-103 can also be found in Appendix E.4, Table 9.

Soft Mixture of Experts (Soft MoE). We use the Soft MoE-tiny with 12 layers. The first 6 layers consist of standard ViT blocks, and the last 6 layers replace the MLP in those blocks with a Soft MoE layer. We train a Momentum-Soft MoE and a baseline Soft MoE model on ImageNet-1K and present their results in Table 3. In addition, we plot the training loss and top-1 accuracy of both models for 120 training epochs in Fig. 4(Right). Notably, there is a considerable increase in the accuracy of Momentum-Soft MoE over the baseline, as well as a clear acceleration to a good solution during

Table 3: Top-1/top-5 accuracy (%) of Momentum-Soft MoE vs. Soft MoE baseline on ImageNet-1K (IN-1K).

Model	Params	Valid IN-1K Top-1 ↑	Valid IN-1K Top-5 ↑
Soft MoE (baseline)	231M	73.52	90.94
Momentum-Soft MoE	231M	**75.47**	**92.34**

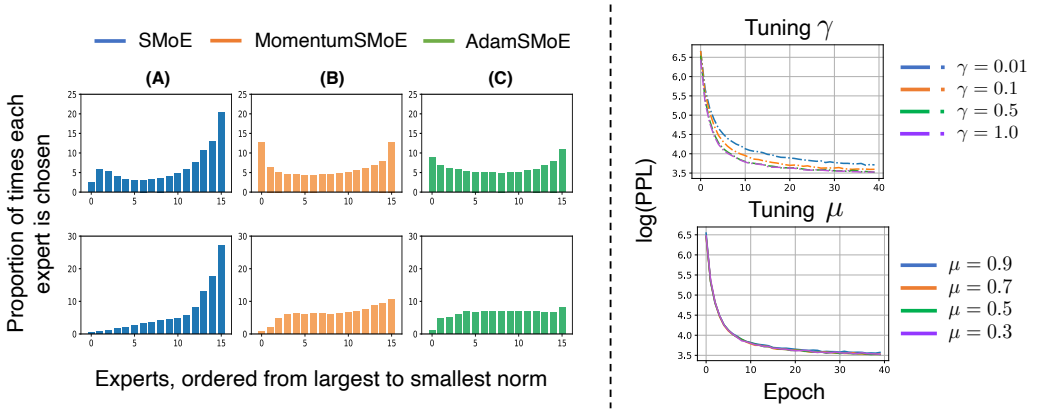

Figure 5: **Left:** Proportion of each expert chosen, ordered from the largest norm of each expert output to the smallest norm, in layers 3 and 5 of SMoE, MomentumSMoE, and Adam SMoE, averaged over the WikiText-103 validation set. **Right:** Log validation perplexity (PPL) during the finetuning of hyperparameters, μ and γ, for 40 training epochs in MomentumSMoE. When tuning γ, we keep $\mu = 0.7$ and vice versa with $\gamma = 1.0$.

training. These findings justify the benefits and universality of our momentum-based method, that extends beyond SMoE to MoE.

6 Empirical Analysis

We conduct empirical analysis based on the SMoE-medium trained on WikiText-103 in Section 5.1.

Norm-based Load Imbalance. Load imbalance in SMoE occurs when only a small subset of experts are consistently selected [63, 77]. Numerous methods have been developed to counter this common phenomenon, such as introducing a buffer capacity for each expert and a load balancing loss [65, 18]. Orthogonal to these, in line with our GD framework for SMoEs, we examine the choice of experts determined by the size of the norm of their outputs, $\|f_i(x)\|$.

From a multi-objective optimization perspective, the optimal descent direction is one that minimizes the norm in the convex hull of the normalized gradients $\bar{U}$ (see Section 2.1). If the gradients are not normalized, the minimum norm direction is then expected to be mainly influenced by the gradients with the smallest norms [12]. From our GD analogy of SMoE in Section 2.3, the gradients correspond to the experts, whose outputs are not normalized. We then visualize the proportion of times the experts in a SMoE are chosen according to their norms during inference in Fig. 5(Left, A). We exactly observe the corresponding phenomenon in the SMoE, further empirically justifying our connection between SMoE and GD. The full plots for all layers and all models are provided in Appendix C.

The direction with the smallest norms are frequently related to the objectives that have already had a substantial degree of convergence and is insufficient for a balanced minimization of the multi-objective criteria. In this light, an ideal SMoE output would have a norm-based balanced choice of experts and should translate to improved model performance. Indeed, in Section 5.1, we established the superior performance of MomentumSMoE and AdamSMoE on large-scale WikiText-103 language modeling task, and in Figure 5(Left, B, C), this directly correlates with a significantly more balanced selection of experts with respect to their norms.

Ablation Study on Momentum μ and Step Size γ. To better understand the effects of the momentum parameter and step size on the performance of the trained MomentumSMoE models, we do an ablation

study on these two hyperparameters and include results in Fig. 5(Right), which contains a plot of log validation PPL during 40 training epochs. We notice that MomentumSMoE is robust to the choice of μ, and we select $\mu = 0.7$ for the final comparison with the baseline SMoE. On the other hand, when the value of γ is too small, there is an adverse effect on the model. Hence, we select $\gamma = 1.0$.

Making Momentum μ and Step Size γ Learnable. We note that additional time and effort are required to tune the momentum parameter and step size. Thus, we explore different methods to circumvent this limitation. We discuss the results of one such method, making γ and μ learnable parameters, in this section and include results in Table 5 in Appendix E.1. We leave the discussion on other methods to Appendix E.1. As shown in Table 5 in Appendix E.1, MomentumSMoE (ii), with learnable γ and fixed μ, significantly outperforms the baseline for both clean validation and test data by at least 1.5 PPL, even surpassing the tuned model in Table 1. On the attacked data, the benefits of MomentumSMoE are further enhanced with more than 2 PPL improvements. These results confirm that the benefits of our model can be leveraged with minimal effort. Since the MomentumSMoE is robust to the choice of μ, we do not consider the setting of fixing γ and making μ learnable.

Other Optimizers. We study the integration of other advanced momentum and optimization methods, such as the Nesterov accelerated gradient [48], RMSProp [68], and sharpness-aware minimization [64], into our MomentumSMoE framework in Appendix E.4.

7 Related Work

Sparse Mixture of Experts. SMoE has been extensively studied to enhance the training efficiency of large language models (LLMs), with various stable routing strategies proposed, including (i) allowing tokens to select the top-k experts [35, 18, 79, 11], (ii) allowing experts to select the top-k tokens [77], and (iii) globally determining expert assignment [36, 9]. Recent works have also tried to enhance the robustness of SMoE. [55] study the robustness of SMoE for ViTs [15] while [75] investigates the robustness of SMoE for CNNs. Furthemore, [37] explores the potential of SMoE for domain generalization, and [22] employs SMoE for robust multi-task learning. Various works have also focused on addressing load imbalance in SMoE, including [63, 77, 7]. Our momentum-based SMoE can be easily incorporated into these methods above to further improve their performance.

Deep Learning Models with Momentum. Momentum has been utilized in the design of deep neural network (DNN) architectures [72, 38]. [23] applies momentum to create large and consistent dictionaries for unsupervised learning using a contrastive loss, with a momentum-based moving average of the queue encoder at the core of this approach. Many DNN-based methods for sparse coding have been designed by unfolding classical optimization algorithms, such as FISTA [4], where momentum is used in the underlying optimizer [45]. In addition, [38] introduces momentum into ResNet and DenseNet, [73, 49] integrate momentum into neural differential equations, [52] incorporates momentum into transformers, and [50] designs RNNs using momentum-accelerated first-order optimization algorithms.

8 Concluding Remarks

In this paper, we propose MomentumSMoE, a new class of SMoE that utilizes heavy-ball momentum to stabilize and robustify SMoE. We theoretically justify the stability of our MomentumSMoE models compared to the SMoE baseline. Furthermore, we demonstrate that our momentum-based design framework for SMoE is universal and can incorporate advanced momentum-based optimization methods, including AdamW [33, 42] and Robust Momentum [10], into many existing SMoE models. We empirically validate the advantage of our momentum-based SMoE over the standard SMoE baseline on WikiText-103 and ImageNet-1K. As shown in Table 7 in Appendix E.2, momentum has no positive effect on the small SMoE of only 3 layers but attains an increasing improvement with the medium and large models of 6 and 12 layers, respectively. This is expected as each layer represents an iteration of GD. A limitation of MomentumSMoE is that while beneficial for larger models, for models that have few layers, MomentumSMoE has little effect. From a theoretical perspective, it would be intriguing to develop a theory to explain the enhanced robustness of MomentumSMoE. Furthermore, MomentumSMoE can be analyzed as a fixed-point iteration. We leave these theoretical developments as future work.

Acknowledgments and Disclosure of Funding

This research / project is supported by the National Research Foundation Singapore under the AI Singapore Programme (AISG Award No: AISG2-TC-2023-012-SGIL). This research / project is supported by the Ministry of Education, Singapore, under the Academic Research Fund Tier 1 (FY2023) (A-8002040-00-00, A-8002039-00-00). This research / project is also supported by the NUS Presidential Young Professorship Award (A-0009807-01-00).

References

[1] Yutong Bai, Xinyang Geng, Karttikeya Mangalam, Amir Bar, Alan L Yuille, Trevor Darrell, Jitendra Malik, and Alexei A Efros. Sequential modeling enables scalable learning for large vision models. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22861–22872, 2024.

[2] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. BEit: BERT pre-training of image transformers. In *International Conference on Learning Representations*, 2022.

[3] Hangbo Bao, Wenhui Wang, Li Dong, Qiang Liu, Owais Khan Mohammed, Kriti Aggarwal, Subhojit Som, Songhao Piao, and Furu Wei. Vlmo: Unified vision-language pre-training with mixture-of-modality-experts. *Advances in Neural Information Processing Systems*, 35:32897–32912, 2022.

[4] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

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

[6] Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in neural information processing systems*, 34:15084–15097, 2021.

[7] 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*, 35:34600–34613, 2022.

[8] Aakanksha Chowdhery, Sharan Narang, Jacob Devlin, Maarten Bosma, Gaurav Mishra, Adam Roberts, Paul Barham, Hyung Won Chung, Charles Sutton, Sebastian Gehrmann, et al. Palm: Scaling language modeling with pathways. *Journal of Machine Learning Research*, 24(240):1–113, 2023.

[9] Aidan Clark, Diego de Las Casas, Aurelia Guy, Arthur Mensch, Michela Paganini, Jordan Hoffmann, Bogdan Damoc, Blake Hechtman, Trevor Cai, Sebastian Borgeaud, et al. Unified scaling laws for routed language models. In *International conference on machine learning*, pages 4057–4086. PMLR, 2022.

[10] Saman Cyrus, Bin Hu, Bryan Van Scoy, and Laurent Lessard. A robust accelerated optimization algorithm for strongly convex functions. In *2018 Annual American Control Conference (ACC)*, pages 1376–1381. IEEE, 2018.

[11] Damai Dai, Li Dong, Shuming Ma, Bo Zheng, Zhifang Sui, Baobao Chang, and Furu Wei. StableMoE: Stable routing strategy for mixture of experts. In Smaranda Muresan, Preslav Nakov, and Aline Villavicencio, editors, *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7085–7095, Dublin, Ireland, May 2022. Association for Computational Linguistics.

[12] Jean-Antoine Désidéri. Multiple-gradient descent algorithm (mgda) for multiobjective optimization. *Comptes Rendus Mathematique*, 350(5):313–318, 2012.

[13] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics.

[14] Samuel Dodge and Lina Karam. A study and comparison of human and deep learning recognition performance under visual distortions. In *2017 26th international conference on computer communication and networks (ICCCN)*, pages 1–7. IEEE, 2017.

[15] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *International Conference on Learning Representations*, 2021.

[16] Nan Du, Yanping Huang, Andrew M Dai, Simon Tong, Dmitry Lepikhin, Yuanzhong Xu, Maxim Krikun, Yanqi Zhou, Adams Wei Yu, Orhan Firat, Barret Zoph, Liam Fedus, Maarten P Bosma, Zongwei Zhou, Tao Wang, Emma Wang, Kellie Webster, Marie Pellat, Kevin Robinson, Kathleen Meier-Hellstern, Toju Duke, Lucas Dixon, Kun Zhang, Quoc Le, Yonghui Wu, Zhifeng Chen, and Claire Cui. GLaM: Efficient scaling of language models with mixture-of-experts. In Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato, editors, *Proceedings of the 39th International Conference on Machine Learning*, volume 162 of *Proceedings of Machine Learning Research*, pages 5547–5569. PMLR, 17–23 Jul 2022.

[17] Gintare Karolina Dziugaite and Daniel M. Roy. Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data. In *Proceedings of the 33rd Annual Conference on Uncertainty in Artificial Intelligence (UAI)*, 2017.

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

[19] Robert Geirhos, Kantharaju Narayanappa, Benjamin Mitzkus, Tizian Thieringer, Matthias Bethge, Felix A Wichmann, and Wieland Brendel. Partial success in closing the gap between human and machine vision. *Advances in Neural Information Processing Systems*, 34:23885–23899, 2021.

[20] Gauthier Gidel, Reyhane Askari Hemmat, Mohammad Pezeshki, Rémi Le Priol, Gabriel Huang, Simon Lacoste-Julien, and Ioannis Mitliagkas. Negative momentum for improved game dynamics. In Kamalika Chaudhuri and Masashi Sugiyama, editors, *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 1802–1811. PMLR, 16–18 Apr 2019.

[21] Gabriel Goh. Why momentum really works. *Distill*, 2017.

[22] Shashank Gupta, Subhabrata Mukherjee, Krishan Subudhi, Eduardo Gonzalez, Damien Jose, Ahmed H Awadallah, and Jianfeng Gao. Sparsely activated mixture-of-experts are robust multi-task learners. *arXiv preprint arXiv:2204.07689*, 2022.

[23] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 9729–9738, 2020.

[24] Dan Hendrycks, Steven Basart, Norman Mu, Saurav Kadavath, Frank Wang, Evan Dorundo, Rahul Desai, Tyler Zhu, Samyak Parajuli, Mike Guo, et al. The many faces of robustness: A critical analysis of out-of-distribution generalization. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 8340–8349, 2021.

[25] Dan Hendrycks and Thomas Dietterich. Benchmarking neural network robustness to common corruptions and perturbations. In *International Conference on Learning Representations*, 2019.

[26] Dan Hendrycks, Kevin Zhao, Steven Basart, Jacob Steinhardt, and Dawn Song. Natural adversarial examples. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15262–15271, 2021.

[27] Geoffrey Hinton. Neural networks for machine learning lecture 6e, 2014.

[28] Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. *Advances in neural information processing systems*, 34:1273–1286, 2021.

[29] Yiding Jiang*, Behnam Neyshabur*, Hossein Mobahi, Dilip Krishnan, and Samy Bengio. Fantastic generalization measures and where to find them. In *International Conference on Learning Representations*, 2020.

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

[31] Nitish Shirish Keskar, Dheevatsa Mudigere, Jorge Nocedal, Mikhail Smelyanskiy, and Ping Tak Peter Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *International Conference on Learning Representations*, 2017.

[32] Nitish Shirish Keskar and Richard Socher. Improving generalization performance by switching from adam to sgd. *arXiv preprint arXiv:1712.07628*, 2017.

[33] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

[34] Kenichi Kumatani, Robert Gmyr, Felipe Cruz Salinas, Linquan Liu, Wei Zuo, Devang Patel, Eric Sun, and Yu Shi. Building a great multi-lingual teacher with sparsely-gated mixture of experts for speech recognition. *arXiv preprint arXiv:2112.05820*, 2021.

[35] Dmitry Lepikhin, HyoukJoong Lee, Yuanzhong Xu, Dehao Chen, Orhan Firat, Yanping Huang, Maxim Krikun, Noam Shazeer, and Zhifeng Chen. {GS}hard: Scaling giant models with conditional computation and automatic sharding. In *International Conference on Learning Representations*, 2021.

[36] Mike Lewis, Shruti Bhosale, Tim Dettmers, Naman Goyal, and Luke Zettlemoyer. Base layers: Simplifying training of large, sparse models. In *International Conference on Machine Learning*, pages 6265–6274. PMLR, 2021.

[37] Bo Li, Yifei Shen, Jingkang Yang, Yezhen Wang, Jiawei Ren, Tong Che, Jun Zhang, and Ziwei Liu. Sparse mixture-of-experts are domain generalizable learners. In *The Eleventh International Conference on Learning Representations*, 2023.

[38] Huan Li, Yibo Yang, Dongmin Chen, and Zhouchen Lin. Optimization algorithm inspired deep neural network structure design. In *Asian Conference on Machine Learning*, pages 614–629. PMLR, 2018.

[39] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

[40] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[41] Jonathan P. Lorraine, David Acuna, Paul Vicol, and David Duvenaud. Complex momentum for optimization in games. In Gustau Camps-Valls, Francisco J. R. Ruiz, and Isabel Valera, editors, *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*, volume 151 of *Proceedings of Machine Learning Research*, pages 7742–7765. PMLR, 28–30 Mar 2022.

[42] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[43] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models. In *International Conference on Learning Representations*, 2017.

[44] Igor Molybog, Peter Albert, Moya Chen, Zachary DeVito, David Esiobu, Naman Goyal, Punit Singh Koura, Sharan Narang, Andrew Poulton, Ruan Silva, et al. A theory on adam instability in large-scale machine learning. *arXiv preprint arXiv:2304.09871*, 2023.

[45] Thomas Moreau and Joan Bruna. Understanding the learned iterative soft thresholding algorithm with matrix factorization. *arXiv preprint arXiv:1706.01338*, 2017.

[46] John Morris, Eli Lifland, Jin Yong Yoo, Jake Grigsby, Di Jin, and Yanjun Qi. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126, 2020.

[47] Yu. Nesterov. Gradient methods for minimizing composite functions. *Mathematical Programming*, 140(1):125–161, 2013.

[48] Yurii Nesterov. A method for solving the convex programming problem with convergence rate $o(1/k^2)$. *Proceedings of the USSR Academy of Sciences*, 269:543–547, 1983.

[49] Nghia Nguyen, Tan Minh Nguyen, Huyen K. Vo, Stanley Osher, and Thieu Vo. Improving neural ordinary differential equations with nesterov's accelerated gradient method. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[50] Tan Nguyen, Richard Baraniuk, Andrea Bertozzi, Stanley Osher, and Bao Wang. Momentum-rnn: Integrating momentum into recurrent neural networks. *Advances in Neural Information Processing Systems*, 33:1924–1936, 2020.

[51] Tan Nguyen, Tam Nguyen, Hai Do, Khai Nguyen, Vishwanath Saragadam, Minh Pham, Khuong Duy Nguyen, Nhat Ho, and Stanley Osher. Improving transformer with an admixture of attention heads. *Advances in neural information processing systems*, 35:27937–27952, 2022.

[52] Tan Minh Nguyen, Richard Baraniuk, Robert Kirby, Stanley Osher, and Bao Wang. Momentum transformer: Closing the performance gap between self-attention and its linearization. In *Mathematical and Scientific Machine Learning*, pages 189–204. PMLR, 2022.

[53] Boris T Polyak. Some methods of speeding up the convergence of iteration methods. *Ussr computational mathematics and mathematical physics*, 4(5):1–17, 1964.

[54] Ofir Press, Noah A. Smith, and Omer Levy. Improving transformer models by reordering their sublayers. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2996–3005, Online, July 2020. Association for Computational Linguistics.

[55] Joan Puigcerver, Rodolphe Jenatton, Carlos Riquelme, Pranjal Awasthi, and Srinadh Bhojanapalli. On the adversarial robustness of mixture of experts. *Advances in Neural Information Processing Systems*, 35:9660–9671, 2022.

[56] 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.

[57] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural Networks*, 12(1):145–151, 1999.

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

[59] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of machine learning research*, 21(140):1–67, 2020.

[60] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do imagenet classifiers generalize to imagenet? In *International conference on machine learning*, pages 5389–5400. PMLR, 2019.

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

[62] Carlos Riquelme, Joan Puigcerver, Basil Mustafa, Maxim Neumann, Rodolphe Jenatton, André Susano Pinto, Daniel Keysers, and Neil Houlsby. Scaling vision with sparse mixture of experts. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 8583–8595. Curran Associates, Inc., 2021.

[63] Clemens Rosenbaum, Ignacio Cases, Matthew Riemer, and Tim Klinger. Routing networks and the challenges of modular and compositional computation. *arXiv preprint arXiv:1904.12774*, 2019.

[64] Vincent Roulet and Alexandre d'Aspremont. Sharpness, restart and acceleration. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[65] 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.

[66] Shashank Subramanian, Peter Harrington, Kurt Keutzer, Wahid Bhimji, Dmitriy Morozov, Michael W Mahoney, and Amir Gholami. Towards foundation models for scientific machine learning: Characterizing scaling and transfer behavior. *Advances in Neural Information Processing Systems*, 36, 2024.

[67] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *International conference on machine learning*, pages 1139–1147. PMLR, 2013.

[68] Tijmen Tieleman and Geoffrey Hinton. Rmsprop: Divide the gradient by a running average of its recent magnitude. coursera: Neural networks for machine learning. *COURSERA Neural Networks Mach. Learn*, 17, 2012.

[69] Hugo Touvron, Matthieu Cord, Matthijs Douze, Francisco Massa, Alexandre Sablayrolles, and Herve Jegou. Training data-efficient image transformers & distillation through attention. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 10347–10357. PMLR, 18–24 Jul 2021.

[70] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[71] Bao Wang, Tan Nguyen, Tao Sun, Andrea L Bertozzi, Richard G Baraniuk, and Stanley J Osher. Scheduled restart momentum for accelerated stochastic gradient descent. *SIAM Journal on Imaging Sciences*, 15(2):738–761, 2022.

[72] Bao Wang, Hedi Xia, Tan Nguyen, and Stanley Osher. How does momentum benefit deep neural networks architecture design? a few case studies. *Research in the Mathematical Sciences*, 9(3):57, 2022.

[73] Hedi Xia, Vai Suliafu, Hangjie Ji, Tan Nguyen, Andrea Bertozzi, Stanley Osher, and Bao Wang. Heavy ball neural ordinary differential equations. *Advances in Neural Information Processing Systems*, 34:18646–18659, 2021.

[74] Liu Yang, Siting Liu, Tingwei Meng, and Stanley J Osher. In-context operator learning with data prompts for differential equation problems. *Proceedings of the National Academy of Sciences*, 120(39):e2310142120, 2023.

[75] Yihua Zhang, Ruisi Cai, Tianlong Chen, Guanhua Zhang, Huan Zhang, Pin-Yu Chen, Shiyu Chang, Zhangyang Wang, and Sijia Liu. Robust mixture-of-expert training for convolutional neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 90–101, 2023.

[76] Yanqi Zhou, Nan Du, Yanping Huang, Daiyi Peng, Chang Lan, Da Huang, Siamak Shakeri, David So, Andrew M. Dai, Yifeng Lu, Zhifeng Chen, Quoc V Le, Claire Cui, James Laudon, and Jeff Dean. Brainformers: Trading simplicity for efficiency. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202, pages 42531–42542. PMLR, 23–29 Jul 2023.

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

[78] 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.

[79] Simiao Zuo, Xiaodong Liu, Jian Jiao, Young Jin Kim, Hany Hassan, Ruofei Zhang, Jianfeng Gao, and Tuo Zhao. Taming sparsely activated transformer with stochastic experts. In *International Conference on Learning Representations*, 2022.

Supplement to "MomentumSMoE: Integrating Momentum into Sparse Mixture of Experts"

Table of Contents

A Proof of Lemma 1

We can find the eigenvalues of matrix A explicitly as

$$\lambda_{\{2\}^{1}} = \frac{(1+\mu-\gamma\sigma(n)) \pm \sqrt{(1+\mu-\gamma\sigma(n))^2 - 4\mu}}{2}$$

If $\gamma\sigma(n) \leq 0$,

$$\begin{aligned}
\lambda_1 &= \frac{(1+\mu-\gamma\sigma(n)) + \sqrt{(1+\mu-\gamma\sigma(n))^2 - 4\mu}}{2} \\
&\geq \frac{(1+\mu) + \sqrt{(1+\mu)^2 - 4\mu}}{2} \\
&= \frac{(1+\mu) + |1-\mu|}{2} \\
&= \max\{1, \mu\} \\
&\geq 1
\end{aligned}$$

Then, we must have $\mu\sigma(n) > 0$. Letting the expression in the square root be Δ, expanding it,

$$\begin{aligned}
\Delta &:= (1+\mu-\gamma\sigma(n))^2 - 4\mu \\
&= (1-\gamma\sigma(n))^2 + 2(1-\gamma\sigma(n))\mu + \mu^2 - 4\mu \\
&= \mu^2 - 2(1-\gamma\sigma(n))\mu + (1-\gamma\sigma(n))^2
\end{aligned}$$

and finding the roots of the equation yields, $\mu = (1 \pm \sqrt{\gamma\sigma(n)})^2$. Then, we consider two cases.

Case 1: $\Delta \geq 0$ when $\mu \geq (1+\sqrt{\gamma\sigma(n)})^2$ or $\mu \leq (1-\sqrt{\gamma\sigma(n)})^2$

1a: If $\mu \geq (1+\sqrt{\gamma\sigma(n)})^2$, then $1+\mu-\gamma\sigma(n) \geq 1+(1+\sqrt{\gamma\sigma(n)})^2-\gamma\sigma(n) \geq 2+2\sqrt{\gamma\sigma(n)} \geq 0$. Then, $\lambda_{1,2} > 0$ and $\lambda_1 \geq \lambda_2$. For the system to converge, we need

$$\lambda_1 = \frac{(1+\mu-\gamma\sigma(n)) + \sqrt{\Delta}}{2} < 1 \iff \sqrt{\Delta} < 1-\mu+\gamma\sigma(n) \tag{19}$$

However, by assumption, we have a contradiction

$$0 \leq \sqrt{\Delta} < 1-\mu+\gamma\sigma(n) \leq 1-(1+\sqrt{\gamma\sigma(n)})^2+\gamma\sigma(n) = -2\sqrt{\gamma\sigma(n)} < 0.$$

Hence, we cannot have $\mu \geq (1+\sqrt{\gamma\sigma(n)})^2$.

1b: If $\mu \leq (1-\sqrt{\gamma\sigma(n)})^2$, we further divide into two more subcases,

1bi: If $1+\mu-\gamma\sigma(n) \geq 0$, again we have, $\lambda_{1,2} > 0$ and $\lambda_1 \geq \lambda_2$. By Eqn. 19, we require $0 \leq \sqrt{\Delta} < 1-\mu+\gamma\sigma(n)$. As $1-\mu+\gamma\sigma(n) \geq 1-(1-\sqrt{\gamma\sigma(n)})^2+\gamma\sigma(n) = 2\sqrt{\gamma\sigma(n)} > 0$, it is enough to check that $\Delta < (1-\mu+\gamma\sigma(n))^2$ can be satisfied.

$$\begin{aligned}
\Delta < (1-\mu+\gamma\sigma(n))^2 &\iff (1+\mu-\gamma\sigma(n))^2 - 4\mu < (1-\mu+\gamma\sigma(n))^2 \\
&\iff (1+\mu-\gamma\sigma(n))^2 - (1-\mu+\gamma\sigma(n))^2 < 4\mu \\
&\iff (1+\mu-\gamma\sigma(n)+1-\mu+\gamma\sigma(n)) \\
&\qquad\qquad (1+\mu-\gamma\sigma(n)-1+\mu-\gamma\sigma(n)) < 4\mu \\
&\iff 4(\mu-\gamma\sigma(n)) < 4\mu \\
&\iff 0 < \gamma\sigma(n)
\end{aligned}$$

Therefore, the conditions for convergence are $i)$ $\mu \leq (1-\sqrt{\gamma\sigma(n)})^2$, $ii)$ $1+\mu-\gamma\sigma(n) \geq 0$ and $iii)$ $\gamma\sigma(n) > 0$. Then by $ii)$, $\mu \geq \gamma\sigma(n) - 1 \geq -1$. Suppose for a contradiction that $\mu \geq 1$, then by $i)$, $1 \leq \mu \leq (1-\sqrt{\gamma\sigma(n)})^2$ which implies that $\gamma\sigma(n) \geq 2\sqrt{\gamma\sigma(n)}$ and hence, $\gamma\sigma(n) \geq 4$. Combining this with condition $i)$, we have $\sqrt{\mu} \leq \sqrt{\gamma\sigma(n)} - 1$ which leads to the following contradiction with $ii)$

$$\begin{aligned}
\mu \leq (1-\sqrt{\gamma\sigma(n)})^2 &\implies \mu \leq 1-2\sqrt{\gamma\sigma(n)}+\gamma\sigma(n) \\
&\implies \mu + 2(\sqrt{\gamma\sigma(n)}-1) \leq \gamma\sigma(n) - 1 \\
&\implies \mu + 2\sqrt{\mu} \leq \gamma\sigma(n) - 1 \\
&\implies 1+\mu-\gamma\sigma(n)+2\sqrt{\mu} \leq 0.
\end{aligned}$$

Then, we must have a last condition for convergence, $iv)$ $|\mu| < 1$.

1bii: If $1+\mu-\gamma\sigma(n) \leq 0$, then $|\lambda_2| \geq |\lambda_1|$ and

$$\lambda_2 = \frac{(1+\mu-\gamma\sigma(n)) - \sqrt{\Delta}}{2} < 0$$

For the system to be stable, we need $\lambda_2 > -1 \iff 3+\mu-\gamma\sigma(n) > \sqrt{\Delta} \geq 0$. Hence, we require $i)$ $\mu \leq (1-\sqrt{\gamma\sigma(n)})^2$, $ii)$ $1+\mu-\gamma\sigma(n) \leq 0$, $iii)$ $3+\mu-\gamma\sigma(n) > 0$ and $iv)$

$$\begin{aligned}
(3+\mu-\gamma\sigma(n))^2 > \Delta &\iff (3+\mu-\gamma\sigma(n))^2 > (1+\mu-\gamma\sigma(n))^2 - 4\mu \\
&\iff 4\mu > (1+\mu-\gamma\sigma(n))^2 - (3+\mu-\gamma\sigma(n))^2 \\
&\iff 4\mu > (1+\mu-\gamma\sigma(n)+3 \qquad\qquad +\mu-\gamma\sigma(n)) \\
&\qquad\qquad (1+\mu-\gamma\sigma(n)-3-\mu+\gamma\sigma(n)) \\
&\iff 4\mu > -4(2+\mu-\gamma\sigma(n)) \\
&\iff 2+2\mu-\gamma\sigma(n) > 0
\end{aligned}$$

Combining $ii)$ and $iv)$, we have, $2+2\mu-\mu-1 > 0 \implies \mu > -1$. Similarly, we assume for a contradiction that $\mu \geq 1$. Then, $\gamma\sigma(n) \geq 1+\mu \geq 2$ and by condition $i)$, $\sqrt{\mu} \leq \sqrt{\gamma\sigma(n)}-1$

from which follows the contradiction

$$\begin{aligned} 1 \leq \sqrt{\gamma\sigma(n)} - \sqrt{\mu} &\implies \sqrt{\gamma\sigma(n)} + \sqrt{\mu} \leq (\sqrt{\gamma\sigma(n)} - \sqrt{\mu})(\sqrt{\gamma\sigma(n)} + \sqrt{\mu}) = \gamma\sigma(n) - \mu \\ &\implies \sqrt{\gamma\sigma(n)} + \sqrt{\mu} \leq \gamma\sigma(n) - \mu < 3 \quad \text{(by } iii\text{)} \\ &\implies \sqrt{\mu} + \sqrt{\gamma\sigma(n)} - 1 < 2 \\ &\implies 2\sqrt{\mu} < 2 \quad \text{(by } i\text{) again).} \end{aligned}$$

Therefore, we need $v)$ $|\mu| < 1$ for convergence as well.

To summarize, for **Case 1**, in order to have a stable system, we require:

$$\begin{aligned} &a) \quad \mu \leq (1 - \sqrt{\gamma\sigma(n)})^2 \\ &b) \quad \text{If } 1 + \mu - \gamma\sigma(n) \geq 0, \\ &\quad\; i) \quad \gamma\sigma(n) > 0 \\ &\quad ii) \quad |\mu| < 1 \\ &c) \quad \text{If } 1 + \mu - \gamma\sigma(n) \leq 0, \\ &\quad\; i) \quad 2 + 2\mu - \gamma\sigma(n) > 0 \\ &\quad ii) \quad |\mu| < 1 \end{aligned} \implies \begin{cases} i) \quad 3 + \mu - \gamma\sigma(n) > 0 \\ ii) \quad 2 + 2\mu - \gamma\sigma(n) > 0 \\ iii) \quad |\mu| < 1 \end{cases}$$

Case 2: $\Delta < 0$ when $(1 + \sqrt{\gamma\sigma(n)})^2 > \mu > (1 - \sqrt{\gamma\sigma(n)})^2$

Then, the eigenvalues are complex and given by

$$\lambda_{\{2\}}^{\{1\}} = \frac{(1 + \mu - \gamma\sigma(n)) \pm i\sqrt{4\mu - (1 + \mu - \gamma\sigma(n))^2}}{2}.$$

For convergence, we require $|\lambda_{1,2}| = \sqrt{\mu} < 1$ or equivalently, $\mu < 1$. Hence, the necessary condition becomes $1 > \mu > (1 - \sqrt{\gamma\sigma(n)})^2$ and to avoid a contradiction, we also require $1 > (1 - \sqrt{\gamma\sigma(n)})^2$. This is satisfied when $\gamma\sigma(n) < 2 + 2\mu$ and $|\mu| < 1$.

Therefore, from all the considered cases, to have a stable system, we require $0 < \gamma\sigma(n) < 2 + 2\mu$ and $|\mu| < 1$, concluding the proof.

B Interpretation of Robust Momentum

To provide intuition behind robust momentum, we follow [10] and view the update rule in Eqn. 16 as a Lur'e feedback control system. For simplicity, we consider the case where, $x_t \in \mathbb{R}$ and write the update in matrix form

$$\begin{pmatrix} x_t \\ x_{t+1} \end{pmatrix} = \begin{pmatrix} 0 & 1 \\ -\mu & 1 + \mu \end{pmatrix} \begin{pmatrix} x_{t-1} \\ x_t \end{pmatrix} + \begin{pmatrix} 0 \\ -\gamma \end{pmatrix} f(y_t); \quad y_t = \begin{pmatrix} -\alpha & 1 + \alpha \end{pmatrix} \begin{pmatrix} x_{t-1} \\ x_t \end{pmatrix} \tag{20}$$

In the frequency domain, in order for the system 20 to be stable, we require that for all $|z| = 1$,

$$\text{Re}((1 - pz^{-1})((k - 1)\tilde{G}(pz) - 1)) < 0,$$

where $\tilde{G}(pz)$ is a transformed transfer function, and p, a scaling factor. The imaginary axis, where $\text{Re} = 0$, is the stability boundary, and the specific construction of the parameters γ, μ and α pushes this boundary into the negative real axis to $-v = (1 + p)(1 - k + 2kp - kp^2)/2p$, hence achieving robust stability through the design of γ, μ and α.

C Empirical Evidence

In this section, we provide the full plots presented in Section 2.3 and 6. We visualize the average norm of the SMoE and MoE outputs respectively, for 80 epochs of training in all 6 layers in Fig. 6 and 7. Notably, in all plots, less a minor exception in layer 3 of SMoE, there is a decrease in the norm of each SMoE and MoE layer output throughout training. This is consistent with our expectation that the gate learns the optimal α^* in a multi-objective optimization problem (Eqn. 8 and 9).

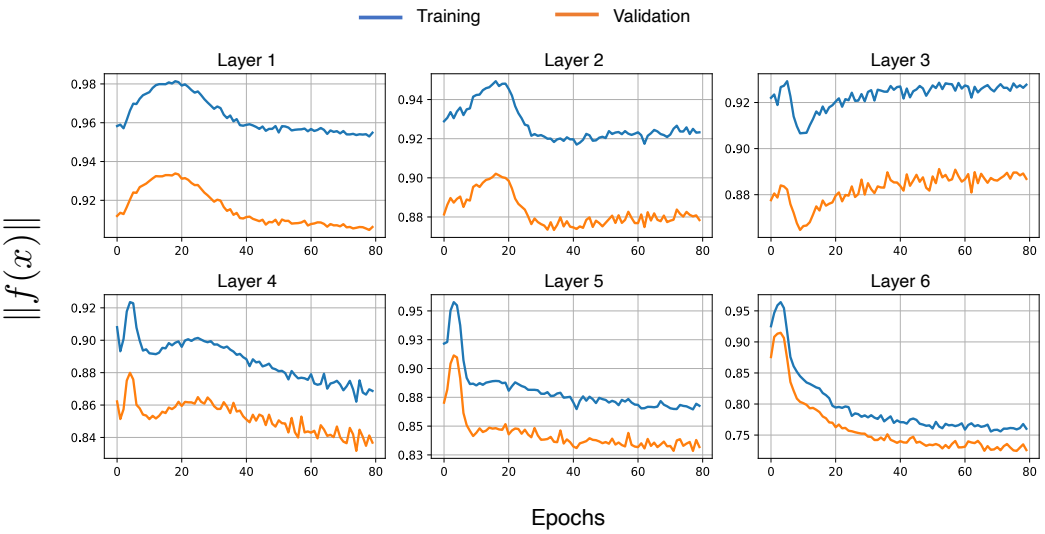

Figure 6: Average norm of the SMoE outputs at all layers during 80 epochs of training for the baseline SMoE.

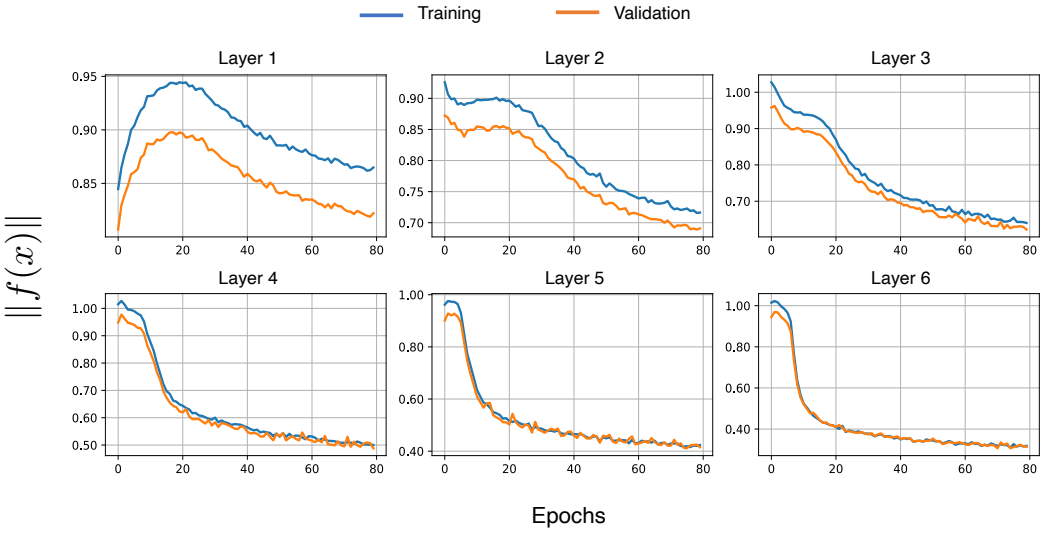

Figure 7: Average norm of the MoE outputs at all layers during 80 epochs of training for the baseline MoE.

In Fig. 8, 9 and 10, we present the proportion of time each expert is chosen in decreasing magnitude of the norm of the expert outputs for SMoE, MomentumSMoE and AdamSMoE. As discussed in Section 6, in accordance with our connection between GD and SMoE, a more even distribution of expert selection based on the size of the norm of the expert outputs should yield improved performance of the model. We demonstrate in Section 5.1 that both MomentumSMoE and AdamSMoE significantly exceed the baseline performance and correspondingly, flattens the normed-based load distribution among experts as observed in Fig. 8, 9 and 10. These strong empirical evidences serve to reinforce our optimization framework for SMoE.

D Experiment Details

For clarity, we summarize the new models developed in this paper for each task in Table 4 and address the lacking implementations here. First, as the introduction of AdamSMoE into V-MoE leads to unstable training, we defer this challenge to future work. Second, our primary goal when studying the Momentum-Soft MoE model is to showcase the universality of our MomentumSMoE method

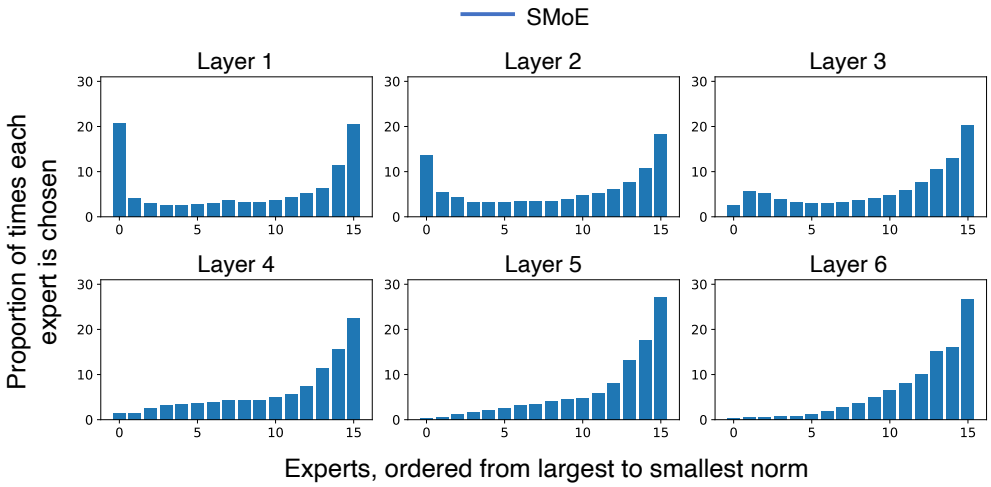

Figure 8: Proportion of each expert chosen, ordered from the largest norm of each expert output to the smallest norm, in all layers of baseline SMoE.

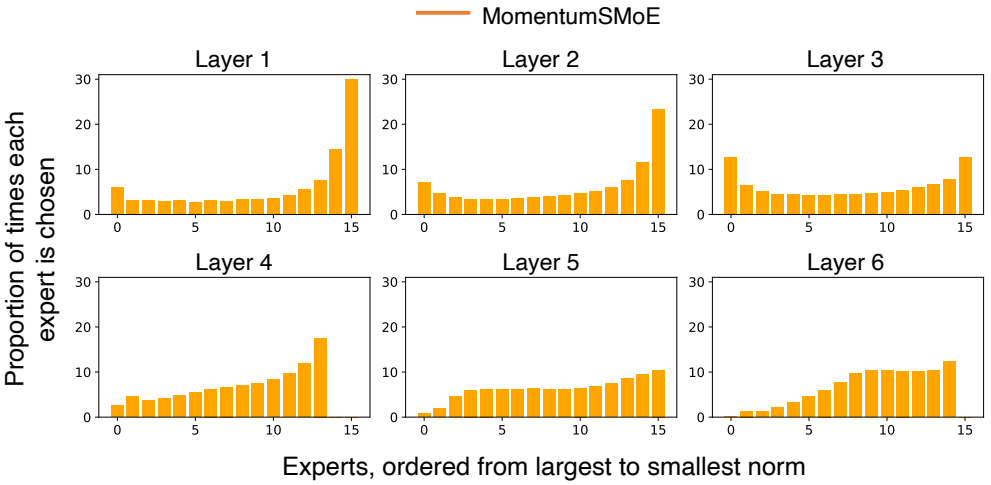

Figure 9: Proportion of each expert chosen, ordered from the largest norm of each expert output to the smallest norm, in all layers of MomentumSMoE.

Table 4: A summary of the new models developed for WikiText-103 language modeling and ImageNet-1K object recognition tasks.

Task	Model/Method	MomentumSMoE	AdamSMoE	Robust MomentumSMoE
WikiText-103	SMoE GLaM	✓	✓	✓
ImageNet-1K ImageNet-R ImageNet-A ImageNet-C	V-MoE	✓	×	✓
ImageNet-1K	Soft MoE	✓	×	×

across both SMoE and MoE, hence we did not integrate AdamW and Robust Momentum into Soft MoE. We leave these implementations for future work.

D.1 WikiText-103 Language Modeling

Dataset: The WikiText-103 dataset [43] is derived from Wikipedia articles and is designed to capture long-range contextual dependencies. The training set contains about 28,000 articles, with a

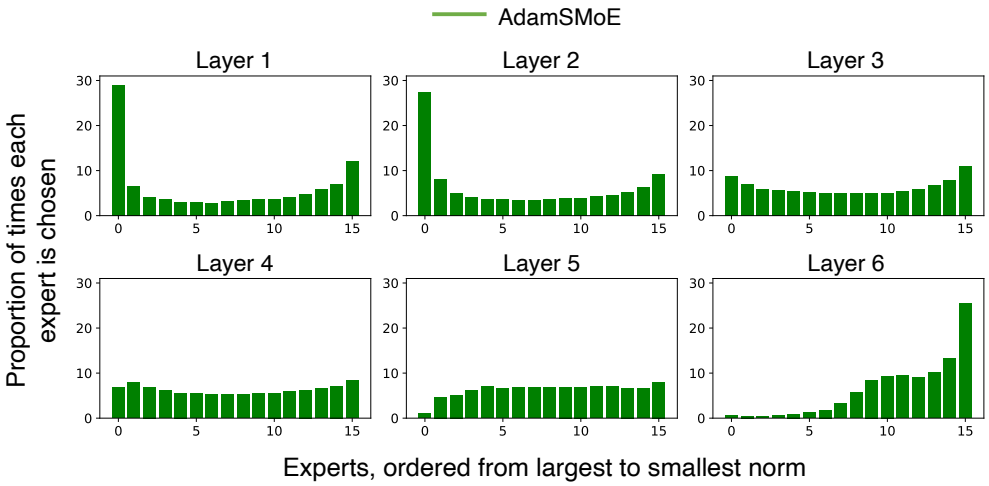

Figure 10: Proportion of each expert chosen, ordered from the largest norm of each expert output to the smallest norm, in all layers of AdamSMoE.

total of 103 million words. Each article is divided into text blocks with approximately 3,600 words. The validation and test sets have 218,000 and 246,000 words, respectively, with both sets comprising 60 articles and totaling about 268,000 words. On the attacked dataset, we corrupt the both validation and test data to demonstrate the robustness of MomentumSMoE and AdamSMoE using TextAttack's word swap attack [46]. This adversarial attack randomly replaces words in the dataset with a generic "AAA" for evaluation making it difficult for the model to predict the next word in the sequence correctly.

Model and baselines: We use the Switch Transformer [18], referred to as SMoE in our tables and figures, and GLaM [16] baselines, which replaces each multilayer perceptron (MLP) layer and every other MLP layer in a vanilla language modeling transformer with a SMoE layer, respectively. For MomentumSMoE, we replace each MLP layer with a MomentumSMoE layer and initialise each momentum vector, p_0 at 0. We do the same for MomentumGLaM with every other MLP layer.

For consistency, we define the number of layers in each model as the number of SMoE layers. The default model used in each experiment is medium sized with 6 layers, but we include a comparison between a smaller model with 3 layers as well as a larger one with 12 layers in Appendix E.2. Each model has 16 experts in every SMoE, MomentumSMoE and AdamSMoE layer and selects 2 experts ($K = 2$) per input. All models use the same sparse router function consisting of a linear network receiving the input data followed by the TopK, then the Softmax function. The small models train for 60 epochs, the medium and large SMoE models train for 80 epochs and the GLaM models train for 120 epochs without any additional load balancing loss. Our implementation is based on the code base developed by [54], publicly available at https://github.com/ofirpress/sandwich_transformer and https://github.com/giangdip2410/CompeteSMoE/tree/main.

AdamSMoE/AdamGLaM: It is observed that Adam has certain divergent behavior during large-scale training leading to unstable loss curves [8, 44]. In line with this observation, during the initial implementations of AdamSMoE, we experience similar instability. A widely used solution to this, proposed by [32], is to switch from Adam to gradient descent at a suitable point during training. We follow suit and use AdamSMoE in the first layer of the model and MomentumSMoE for the subsequent layers. We find that implementing AdamSMoE in the first layer is enough to significantly improve the model's overall performance and accelerate its convergence.

D.2 ImageNet-1K Object Recognition

Datasets: We use the ImageNet-1K dataset that contains 1.28M training images and 50K validation images. There are 1000 classes of images and the model learns an object recognition task. For robustness to common corruptions, we use ImageNet-C (IN-C) [25] which consists of 15 different types of corruptions applied to the ImageNet-1K validation set with 5 levels of severity. We provide a breakdown of our results on each corruption type and the mean Corruption Error (mCE) across

```
Hyperparameters: mu

def MomentumSMoE(x, momentum):
    momentum = - SMoE(x) + mu * momentum
    x = x + gamma * momentum
    return x
```

Figure 11: Pseudocode for MomentumSMoE implementation in python with 1 hyperparameter μ.

```
Hyperparameters: mu, beta, eps = 1e-8

def AdamSMoE(x, gradient, squared_gradient):
    gradient = - (1 - mu) * SMoE(x) + mu * momentum
    squared_gradient = beta * squared_gradient + (1 - beta) * SMoE(x) ** 2
    x = x + gamma / (torch.sqrt(squared_gradient) + eps) * gradient - k * x
    return x
```

Figure 12: Pseudocode for AdamSMoE implementation in python with 2 hyperparameters μ and β.

escalating levels of severity for two corruption types in Appendix E.3. To test robustness to input data distribution shifts, we use ImageNet-A (IN-A) [26]. IN-A contains a 200 class subset of ImageNet-1K classes with adversarially filtered images. Finally, we test our model on ImageNet-R (IN-R) [24] which contains various artistic renditions of images. This evaluates the model's generalization ability to abstract visual renditions.

Metrics: On ImageNet-1K, ImageNet-A and ImageNet-R, we report the top-1 accuracies for all experiments. On ImageNet-C, the standard metric for evaluation is the mCE. To calculate this, we average the top-1 error rate for each corruption type across the 5 levels of severity and divide them by AlexNet's average errors, then take the final average across all corruption types. The direction of increasing or decreasing values of these metrics signifying greater robustness will be indicated in the table with an arrow.

Model and baselines: We use a small Vision Mixture of Experts (V-MoE) [62] model as the SMoE baseline for ImageNet-1K object recognition task as well as the standard robustness benchmarks. This variant of V-MoE consists of 8 Vision Transformer (ViT) blocks [15] with every odd block's MLP being replaced by a SMoE layer. In turn, in MomentumV-MoE, we replace every other MLP layer with a MomentumSMoE layer and similarly for Robust MomentumV-MoE. For all vision SMoE models, we select 2 experts ($K = 2$) at every SMoE layer for each patch. We follow the training configurations and setting as in [62] and their code base is available here https://github.com/google-research/vmoe/.

For our MoE baseline on clean ImageNet-1K data, we use Soft Mixture of Experts (Soft MoE) [56]. A Soft MoE model is designed to side step the challenging discrete optimization problem of assigning each token to an expert, as in SMoE, through a soft token assignment. Instead of each token being routed to one expert, in a Soft MoE layer, each expert is assigned a certain number of slots and each slot is allocated a weighted average of all tokens. These weights are dependant on both the tokens and the experts. In this case, Soft MoE is not considered as a SMoE, but instead a MoE. We use the smallest model, Soft MoE-tiny with 12 layers. The first 6 layers consists of standard ViT blocks and the last 6 layers replace the MLP in those blocks with a Soft MoE layer. In Momentum-Soft MoE, we implement the momentum parameter into each Soft-MoE layer as in a SMoE layer.

As there were no training details provided for Soft MoE on ImageNet-1K, we follow the training procedure in [69] for 120 epochs, using their published code base https://github.com/facebookresearch/deit. We train Momentum-Soft MoE and the baseline model using the PyTorch implementation of Soft MoE at https://github.com/bwconrad/soft-moe.

D.3 Pseudocode

In this section, we provide the pseudocode as written in python for MomentumSMoE, AdamSMoE, and Robust MomentumSMoE for clarification on our implementation. These are found in Figures 11, 12 and 13 respectively.

```
Hyperparameters: p, L, m

def RobustMomentumSMoE(x, momentum):
    k = L / m
    gamma = k * ((1 - p) ** 2) * (1 + p) / L
    mu = k * p ** 3 / (k - 1)
    alpha = p ** 3 / ((k - 1) * ((1 - p) ** 2) * (1 + p))
    y = x + alpha * gamma * momentum
    momentum = - SMoE(y) + mu * momentum
    x = x + gamma * momentum
    return x
```

Figure 13: Pseudocode for Robust MomentumSMoE implementation in python with 3 hyperparameters p, L and m.

Table 5: Perplexity (PPL) results on clean and attacked WikiText-103 validation and test data for standard MomentumSMoE with tuned hyperparameters μ and γ and MomentumSMoE trained with different learning settings for μ and γ that do not require tuning: *i)* Both μ and γ are scalar learnable parameters in Pytorch. *ii)* Only γ is learned with fixed $\mu = 0.7$.

Model	μ	γ	WikiText-103	Valid PPL ↓	Test PPL ↓
SMoE (baseline)	-	-	Clean	33.76	35.55
			Attacked	42.24	44.19
MomentumSMoE	0.7	1.0	Clean	32.29	**33.46**
			Attacked	40.94	42.33
MomentumSMoE *(i)*	Learnable	Learnable	Clean	**32.28**	33.87
			Attacked	40.41	42.32
MomentumSMoE *(ii)*	0.7	Learnable	Clean	**32.28**	33.69
			Attacked	**39.96**	**41.84**

E Additional Experimental Results

E.1 Hyperparameters

In this section, we discuss two additional methods to avoid tuning MomentumSMoE hyperparameters for ease of implementation and efficiency.

Time-varying momentum: Another form of the classical momentum proposed by [48] replaces the constant momentum parameter with a time-varying one $t-1/t+2$, which removes the need to choose an appropriate momentum hyperparameter. We adopt this modification into MomentumSMoE to replace μ while keeping γ fixed at 1.0, and the MomentumSMoE time-varying (TV) layer is as follows

$$\boldsymbol{p}_t = -f(\boldsymbol{x}_t) + \frac{t-1}{t+2}\boldsymbol{p}_{t-1}; \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \gamma \boldsymbol{p}_t$$

Zero-order hold μ and γ: Recall Eqn. 10, the proposed accelerated MomentumSMoE layer

$$\boldsymbol{p}_t = -f(\boldsymbol{x}_t) + \mu \boldsymbol{p}_{t-1}; \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \gamma \boldsymbol{p}_t.$$

When we replace MLP layers with MomentumSMoE layers, as we do in our language model MomentumSMoE, each expert function is a linear network. Explicitly expressing the MomentumSMoE layer results in the following expression

$$\boldsymbol{p}_t = (-\sum_{i=1}^{E} g(\boldsymbol{x}_t)_i \boldsymbol{W}_i^\top)\boldsymbol{x}_t + \mu \boldsymbol{p}_{t-1} = \bar{\boldsymbol{B}}\boldsymbol{x}_t + \mu \boldsymbol{p}_{t-1}; \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \gamma \boldsymbol{p}_t$$

and can be interpreted as a state space representation with constant state and output matrices, μ and γ. From this perspective, we experiment with learning a discretized μ_t and γ_t by applying the Zero-order hold (ZOH) such that they become adaptive parameters as is a common practise in optimization algorithms. μ and γ are then parameterized as follows

$$\mu_t = e^{\Delta\mu}; \quad \gamma_t = (\Delta\mu)^{-1}(e^{\Delta\mu} - 1)\Delta\gamma$$

Table 6: Perplexity (PPL) results on clean and attacked WikiText-103 validation and test data for standard MomentumSMoE with tuned hyperparameters μ and γ and MomentumSMoE trained with different learning settings for μ and γ that do not require tuning: *i)* Both μ and γ are scalar learnable parameters in Pytorch. *ii)* Only γ is learned with fixed $\mu = 0.7$. *iii)* Only γ is learned as a linear network then composed with a sigmoid function with fixed μ. Included are the time-varying MomentumSMoE (TV) and Zero-order hold MomentumSMoE (ZOH).

Model	μ	γ	WikiText-103	Valid PPL ↓	Test PPL ↓
SMoE (baseline)	-	-	Clean	33.76	35.55
			Attacked	42.24	44.19
MomentumSMoE	0.7	1.0	Clean	32.29	**33.46**
			Attacked	40.94	42.33
MomentumSMoE *(i)*	Learnable	Learnable	Clean	32.28	33.87
			Attacked	40.41	42.32
MomentumSMoE *(ii)*	0.7	Learnable	Clean	32.28	33.69
			Attacked	**39.96**	**41.84**
MomentumSMoE *(iii)*	0.7	Linear network	Clean	32.30	33.96
			Attacked	40.35	42.39
MomentumSMoE (TV)	$t-1/t+2$	1.0	Clean	33.02	35.08
			Attacked	41.44	43.97
MomentumSMoE (ZOH)	$e^{\Delta\mu}$	$(e^{\Delta\mu}-1)\Delta\gamma/\Delta\mu$	Clean	**32.21**	34.00
			Attacked	40.74	42.70

where Δ is the Softplus function of a learned scalar parameter, μ a learned scalar parameter and γ a linear network with scalar outputs.

Results: We report the PPL for MomentumSMoE (TV) and MomentumSMoE (ZOH) on clean and word swap attacked WikiText-103 validation and test data in Table 6, an extended version of Table 5. We also include an additional setting of learning γ as the sigmoid of a linear network. In this setting, every input has a different learning rate. While these models do improve over the baseline, there is no advantage over the standard MomentumSMoE and the other learnable μ and γ settings. Hence, we do not recommend implementations in this section over those and keep the results here for reference.

E.2 WikiText-103 Language Modeling

We present the results of all three sizes of SMoE and MomentumSMoE models in Table 7 and observe that with increasing model depth, the effectiveness of the momentum parameter in improving model performance increases as well. This aligns with the analogy of each layer being a GD step and with a higher number of iterations, the momentum term becomes more effective. While beneficial for large models, as implementing MomentumSMoE is computationally cost efficient, and hence, minimally affected by model depth, we note that there is an adverse effect in small models such as SMoE-small with only 3 layers. We hypothesize that the primary reason for this are the insufficient layers for the momentum term to make a positive impact and is a limitation of our method.

E.3 ImageNet-C Full Results

We provide the full results of the top-1 accuracy and mCE of all 15 corruption types in ImageNet-C for V-MoE, MomentumV-MoE, Robust MomentumV-MoE and Sharpness Aware MomentumV-MoE (SAM-V-MoE), developed in Appendix E.4, in Table 8. Included in Figure 14 is a plot of the mCE with escalating severity of impulse noise and gaussian noise corruption, which illustrates the advantages of our methods with increasing data corruption. We observe that across all 15 corruption types, except for motion blur, Robust MomentumV-MoE outperforms the baseline V-MoE, with as high as a 6.5% increase in top-1 accuracy and 8 mCE decrease on fog corruption.

E.4 Further Extensions Beyond Adam and Robust Momentum

The advantage of an optimization perspective for SMoEs are the countably many descent algorithms available that can be used to improve the model. In this section, we elaborate on six more extensions

Table 7: Perplexity (PPL) of baseline SMoE-small/medium/large, MomentumSMoE-small/medium/large, AdamSMoE-medium, baseline GLaM-medium, MomentumGLaM-medium and AdamGLaM-medium on clean and attacked WikiText-103 validation and test data.

Model/Metric	Clean WikiText-103		Attacked WikiText-103	
	Valid PPL ↓	Test PPL ↓	Valid PPL ↓	Test PPL ↓
SMoE-small (baseline)	**84.26**	**84.81**	**98.60**	**99.29**
MomentumSMoE-small	85.71	86.65	100.26	101.18
SMoE-medium (baseline)	33.76	35.55	42.24	44.19
MomentumSMoE-medium	32.29	33.46	40.94	42.33
AdamSMoE-medium	**31.59**	**33.25**	**39.27**	**41.11**
GLaM-medium (baseline)	36.37	37.71	45.83	47.61
MomentumGLaM-medium	33.87	35.29	42.15	43.64
AdamGLaM-medium	**32.99**	**34.32**	**41.09**	**42.81**
SMoE-large (baseline)	29.31	30.33	36.77	37.83
MomentumSMoE-large	**27.58**	**29.03**	**35.21**	**36.78**

Table 8: Top-1 accuracy (%) and mean corruption error (mCE) of V-MoE, MomentumV-MoE, Robust MomentumV-MoE and SAM-V-MoE on each corruption type in ImageNet-C.

Corruption Type	Model/ Metric	*V-MoE (Baseline)*	MomentumV-MoE	Robust MomentumV-MoE	SAM-V-MoE
Brightness	Top-1 ↑	67.25	67.77	**67.79**	67.34
	mCE ↓	58.01	57.08	**57.06**	57.85
Contrast	Top-1 ↑	36.94	40.94	**42.71**	41.64
	mCE ↓	73.91	69.22	**67.15**	68.41
Defocus Blur	Top-1 ↑	39.89	41.01	**41.02**	40.26
	mCE ↓	73.31	71.95	**71.94**	72.86
Elastic Transform	Top-1 ↑	53.14	53.08	**53.19**	52.89
	mCE ↓	72.53	72.63	**72.45**	72.92
Fog	Top-1 ↑	38.23	42.00	**44.85**	44.03
	mCE ↓	75.39	70.79	**67.31**	68.31
Frost	Top-1 ↑	50.12	51.51	**51.83**	51.45
	mCE ↓	60.35	58.66	**58.27**	58.73
Gaussian Noise	Top-1 ↑	49.38	50.92	**52.08**	50.92
	mCE ↓	57.11	55.37	**54.06**	55.36
Glass Blur	Top-1 ↑	36.65	36.74	**37.30**	36.43
	mCE ↓	76.67	76.56	**75.88**	76.93
Impulse Noise	Top-1 ↑	47.72	49.30	**50.59**	49.15
	mCE ↓	56.66	54.95	**53.56**	55.11
JPEG Compression	Top-1 ↑	60.21	60.44	**60.53**	60.23
	mCE ↓	65.61	65.23	**65.08**	65.58
Motion Blur	Top-1 ↑	**43.59**	43.10	43.35	42.08
	mCE ↓	**71.77**	72.40	72.08	73.69
Pixelate	Top-1 ↑	61.66	62.61	**63.14**	62.30
	mCE ↓	53.41	52.09	**51.35**	52.52
Shot Noise	Top-1 ↑	47.96	49.03	**50.51**	49.66
	mCE ↓	58.18	56.98	**55.33**	56.28
Snow	Top-1 ↑	36.91	37.31	**37.32**	37.12
	mCE ↓	72.78	72.32	**72.31**	72.54
Zoom Blur	Top-1 ↑	35.03	35.88	**36.14**	35.20
	mCE ↓	81.38	80.31	**79.99**	81.17

to MomentumSMoE, namely, Nesterov accelerated gradient [48], time-varying momentum with scheduled restart [64, 47], RMSprop [27], sharpness aware minimization [64] (SAM), negative momentum [20], and complex momentum [41]. We report their results on clean and word swap attacked WikiText-103 language modeling task in Table 9. The nature of SAM is to find local minima where the geometry of the loss landscape is flat to improve the generalization ability of the model. This aligns with the objective of improving the robustness of models to distribution shifts. Hence, we implement SAM in V-MoE as well and provide a comparison with V-MoE, MomentumV-MoE

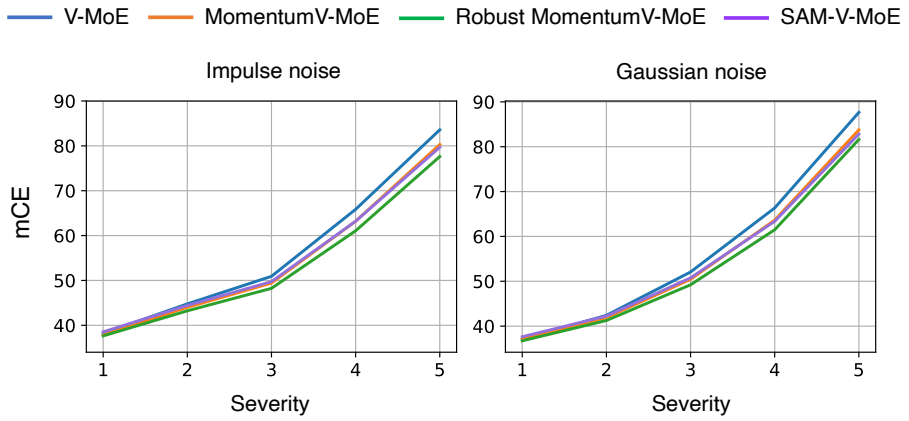

Figure 14: mCE (lower is better) of baseline V-MoE, MomentumV-MoE, Robust MomentumV-MoE and SAM-V-MoE on increasing severities of impulse noise and gaussian noise corruption. As the severity increases, the effect of momentum, robust momentum and SAM becomes increasingly apparent.

Table 9: Perplexity (PPL) results on clean and word swap attacked WikiText-103 validation and test data for baseline SMoE, NAG-SMoE, SR-SMoE, rms-SMoE, SAM-SMoE, Robust MomentumSMoE, Negative-MomentumSMoE, and Complex-MomentumSMoE.

Model/Metric	Clean WikiText-103		Attacked WikiText-103	
	Valid PPL ↓	Test PPL ↓	Valid PPL ↓	Test PPL ↓
SMoE (baseline)	33.76	35.55	42.24	44.19
NAG-SMoE	33.83	35.46	41.94	43.97
SR-SMoE	32.96	35.01	41.21	43.72
rms-SMoE	32.43	34.25	40.58	42.60
SAM-SMoE	33.39	35.05	41.47	43.44
Robust MomentumSMoE	33.22	34.45	41.49	42.78
Negative-MomentumSMoE	33.48	35.09	41.68	43.62
Complex-MomentumSMoE	**32.08**	**33.34**	**40.24**	**41.66**

and Robust MomentumV-MoE in Table 10. In all models, we replace all SMoE layers with their corresponding newly derived layer unless stated otherwise.

Nesterov accelerated gradient (NAG): Nesterov accelerated gradient (NAG) [48] takes the momentum method a step further by looking ahead to where the momentum term will move the parameters and providing a correction. The foresight of the update prevents the parameters from moving in an undesirable direction too quickly, preventing large oscillations especially when the learning rate is high. We implement NAG in our SMoE gradient framework as

$$\boldsymbol{p}_t = -\gamma f(\boldsymbol{x}_t - \mu \boldsymbol{p}_{t-1}) + \mu \boldsymbol{p}_{t-1}; \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \boldsymbol{p}_t$$

where $f(\boldsymbol{x}_t)$ is the SMoE output and $\mu \in (0, 1)$ and $\gamma > 0$ are two hyperparameters corresponding to the momentum coefficient and step size respectively. We refer to this implementation as a NAG-SMoE.

Time-varying momentum with scheduled restart: An enhancement of the time-varying momentum covered in Section E.1 is to include a scheduled restart for the momentum parameter, $t - 1/t + 2$. Such a modification can help to recover an optimal convergence rate as accelerated methods do not necessarily maintain a monotonic decrease in objective value [64, 47]. As our model has 6 layers, we choose to restart after layer 3 and the RS-SMoE update is

$$\boldsymbol{p}_t = -f(\boldsymbol{x}_t) + \frac{t \mod 3}{t \mod 3 + 3} \boldsymbol{p}_{t-1}; \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \gamma \boldsymbol{p}_t$$

where $f(\boldsymbol{x}_t)$ is the SMoE output and $\gamma > 0$ is the step size.

RMSprop: RMSprop is an adaptive step size algorithm that scales the gradient by an exponentially decaying average of the squared gradients [27]. The algorithm replaces a global learning rate, which

Table 10: Top-1 accuracy (%) and mean corruption error (mCE) of V-MoE, MomentumV-MoE, Robust MomentumV-MoE and SAM-V-MoE on validation and test ImageNet-1K data and popular standard robustness benchmarks for image classification.

Model	Valid IN-1K Top-1 ↑	Test IN-1K Top-1 ↑	IN-R Top-1 ↑	IN-A Top-1 ↑	IN-C Top-1 ↑	IN-C mCE ↓
V-MoE (baseline)	76.49	73.16	36.10	5.25	46.98	67.14
MomentumV-MoE	**76.92**	**73.26**	37.45	**6.48**	48.11	65.77
Robust MomentumV-MoE	76.66	73.20	**37.57**	6.37	**48.82**	**64.92**
SAM-V-MoE	76.26	72.84	36.64	6.27	48.05	65.88

is difficult to tune due to the widely differing magnitudes of the gradients of each parameter, with one that adapts to the individual parameter gradients. In addition, by keeping a running average of the squared gradients, we avoid extreme fluctuations in the step size due to stochastic batch updates. Incorporating rmsprop into the SMoE optimization perspective, we have the following update in our rms-SMoE layer,

$$\boldsymbol{p}_t = \mu \boldsymbol{p}_{t-1} + (1-\mu) f(\boldsymbol{x}_t)^2; \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \frac{\gamma}{\sqrt{\boldsymbol{p}_t + \epsilon}} f(\boldsymbol{x}_t)$$

where $f(\boldsymbol{x}_t)$ is the SMoE output, $\mu \in (0,1)$ is the moving average parameter and, $\gamma > 0$ the step size. Similar to AdamSMoE, we experience some instability when using rms-SMoE to replace all SMoE layers. Hence, we follow suit, and only replace the first layer with rms-SMoE and replace subsequent SMoE layers with MomentumSMoE.

Sharpness-Aware Minimization: In a standard machine learning approach, training models with the usual optimization algorithms minimize the training empirical loss, which is typically non-convex. This results in multiple local minima that may have similar loss values but lead to vastly different generalization abilities. As such, models that perform well on training data, may still have inferior validation performance. From studies relating the geometry of the loss landscape to generalization, specifically, and intuitively, a flatter minima should yield greater generalization ability [17, 29, 31]. Further, this would increase the model's robustness to distribution shifts in input data or labels.

The sharpness-aware minimization (SAM) algorithm [64] aims to take advantage of this relationship and seek not just any local minimum during training, but a minima whose neighbourhood also has uniformly low loss values, in other words, lower sharpness, to improve robustness. Implementing the algorithm with the l_2-norm in the SMoE optimization framework yields the SAM-SMoE layer

$$\hat{\epsilon}(\boldsymbol{x}_t) = \rho f(\boldsymbol{x}_t)/\|f(\boldsymbol{x}_t)\|_2; \quad \boldsymbol{p}_t = f(\boldsymbol{x}_t + \hat{\epsilon}(\boldsymbol{x}_t)); \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t - \gamma \boldsymbol{p}_t$$

where $f(\boldsymbol{x}_t)$ is the SMoE output, $\rho > 0$ is a hyperparameter controlling the size of the neighbourhood and, $\gamma > 0$ the step size.

Complex and Negative Momentum: Classic momentum works well in a gradient-based optimization when the Jacobian of our update step, as a fixed-point operator, has real eigenvalues. However, this might not always be the case. Negative momentum, which simply chooses negative momentum parameters, is preferred by operators with complex eigenvalues [20]. In situations where the spectrum is purely imaginary or has mixtures of complex and imaginary eigenvalues, complex momentum is more robust than both classic and negative momentum [41]. This is due to its oscillations at fixed frequencies between adding and subtracting the momentum term. We oscillate the sign of the momentum term by choosing a complex momentum parameter and then updating our weights by only the real part of the momentum term. This translates to the following update in the Complex-MomentumSMoE layer:

$$\boldsymbol{p}_t = -f(\boldsymbol{x}_t) + \mu \boldsymbol{p}_{t-1}; \quad \boldsymbol{x}_{t+1} = \boldsymbol{x}_t + \gamma \Re(\boldsymbol{p}_t), \tag{21}$$

where $\mu \in \mathbb{C}$ and $\gamma > 0$ are hyperparameters corresponding to the momentum coefficient and step size, respectively and $\Re$ extracts the real component of the momentum term.

Results: On the language modeling task, though all models, except NAG-SMoE on clean WikiText-103 validation data, do outperform the baseline across all metrics, most have only marginal gains that fall short of the performance gap achieved with MomentumSMoE and AdamSMoE. This is

Table 11: Run time per sample, memory and number of parameters of MomentumSMoE and AdamSMoE as compared to the baseline SMoE during training and test time.

Model	Sec/Sample (Training)	Sec/Sample (Test)	Memory (Training)	Memory (Test)	Parameters
SMoE (baseline)	0.0315	0.0303	22168MB	17618MB	216M
MomentumSMoE	0.0317	0.0304	22168MB	17618MB	216M
AdamSMoE	0.0321	0.0307	22168MB	17618MB	216M

Table 12: Total computation time for SMoE, MomentumSMoE and AdamSMoE to reach 38 PPL on WikiText-103 validation data.

Model	Time (minutes)
SMoE (baseline)	85.56
MomentumSMoE	**81.77**
AdamSMoE	84.23

with the exception of Complex-MomentumSMoE, which has an even greater improvement over the baseline than MomentumSMoE. Complex-MomentumSMoE's enhanced results further verify the power and promise of our momentum-based framework in designing SMoE. On the computer vision task, we find that SAM-V-MoE does perform relatively well on corruption benchmarks, exceeding the baseline by more than 1% on ImageNet-A and ImageNet-C.

E.5 Comparison of Computational Efficiency

A common consideration when introducing modified layers into deep models is the potential increase in computational overhead. We aim to alleviate that concern by providing the run time per sample, memory and number of parameters of MomentumSMoE and AdamSMoE as compared to the baseline SMoE during both training and test time in Table 11. We also provide the total computation time required for all models to reach the same PPL level on WikiText-103 validation data in Table 12. We observe that MomentumSMoE and AdamSMoE are comparable to the baseline SMoE across all metrics at both training and test time with negligible computational cost.

F Broader Impacts

Our research enhances both clean data handling and robust performance, particularly in socially impactful domains. Notably, we demonstrate improved results in object recognition, benefiting self-driving cars, and language modeling, enhancing AI chatbot assistants. We show significant advancements in resisting data perturbation, aiming to protect critical AI systems from malicious actors. Furthermore, we achieve competitive performance in language modeling with contaminated data, reflecting real-world scenarios where data is often imperfect. While the potential for AI misuse exists, our work provides substantial improvements in fundamental architectures and theory, which we hope will lead to further socially beneficial outcomes.

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 claims made in the abstract and introduction are clearly stated in the **Contribution** in the Introduction. These claims accurately reflect the paper's contributions and scope.

   Guidelines:

   - The answer NA 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 NA 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: The limitations are discussed in the Conclusion.

   Guidelines:

   - The answer NA 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: All theoretical results in the paper are given together with the full set of assumptions and complete/correct proofs (See Section 3 and Appendix A in our manuscript).

Guidelines:

- The answer NA 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: We provide the experiment details in the Experiment Details Section in the Appendix of our manuscript. We also provide the source code so that the results in the paper can be easily reproduced.

Guidelines:

- The answer NA 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 provide the source code so that the results in the paper can be easily reproduced. We verify our proposed methods using public benchmarks (See the Experimental Results Section in our manuscript)

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (https://nips.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://nips.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, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: We specify all the training and test details necessary to understand the results in the Experiment Details Section in the Appendix of our manuscript.

   Guidelines:

   - The answer NA 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: We report error bars suitably and correctly defined of the experiments.

   Guidelines:

   - The answer NA 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: We provide sufficient information on the computer resources for all experiments in our Experimental Results Section.

   Guidelines:

   - The answer NA 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 research conducted in the paper conforms, in every respect, with the NeurIPS Code of Ethics.

   Guidelines:

   - The answer NA 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: [Yes]

    Justification: We discuss broader impacts in Appendix F.

    Guidelines:

    - The answer NA means that there is no societal impact of the work performed.

- If the authors answer NA 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., pretrained language models, image generators, or scraped datasets)?

    Answer: [NA]

    Justification: The paper poses no such risks.

    Guidelines:

    - The answer NA 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 cite the githubs we use and the baselines we compare with in our manuscript.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: The paper does not release new assets.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: The paper does not involve crowdsourcing or research with human subjects.

    Guidelines:

    - The answer NA 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: [NA]

    Justification: The paper does not involve research with human subjects.

    Guidelines:

    - The answer NA 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.