Noether's Learning Dynamics: The Role of Symmetry Breaking in Deep Learning

Anonymous Author(s)
Affiliation
Address
email

Abstract

Recent work has identified a myriad of symmetries in the architecture and loss function of deep learning systems. This raises a fundamental question, is such symmetry beneficial, harmful, or irrelevant to the success of learning? Here, we reveal that the efficiency and stability in modern deep learning is in part due to *broken* symmetries. To build this understanding, we model the discrete learning dynamics of gradient descent using a continuous-time Lagrangian formulation, in which the learning rule corresponds to the kinetic energy and the loss function corresponds to the potential energy. We identify *kinetic asymmetry*, the condition when the kinetic energy does not observe the same symmetry as the potential function unique to learning systems. We generalize Noether's theorem known in physics to explicitly take into account this kinetic asymmetry and derive the resulting motion of the Noether charge. Finally, we apply our theory to modern deep networks with normalization layers and reveal a mechanism of implicit adaptive optimization induced by the kinetic symmetry breaking. Through the lens of variational formulation of mechanics, we identify that *kinetic symmetry breaking* is a key design principle to the success of a learning system.

With the rapid increase in available data and computational power, machine learning has become an essential tool both in science and engineering. To obtain an accurate model, we have to adjust randomly initialized parameters $q \in \mathbb{R}^N$ such that it minimizes the loss function $f(X; q) \in \mathbb{R}$ a measure of discrepancy between the model's predictions from input data X and the provided truth. Learning rules specify how we iteratively update these parameters based on their recent trajectory and local geometry of the non-convex loss landscape. We can represent the repeated updates of the millions of parameters during learning as discrete movements of a point in high-dimensional parameter space. The data and model architecture together shape the static loss landscape, and the learning rules govern the trajectory of the parameters $q(t)$ during training. To make the learning dynamics more efficient and stable, researchers and practitioners are actively looking for improved model architectures and learning rules. Thus, principles are needed to effectively navigate the vast design space of model architectures and learning rules.

Symmetries in the model parameters. Symmetry provides such principles for understanding the myriad of modern deep learning architectures [1]. In this work, we focus on symmetries of the loss function $f(X; q)$ in the parameter space q. The first example is the ReLU function defined as $\text{ReLU}(qx) = \max(0, qx)$, which is the most successful and popular activation function in deep learning. Given two consecutive layers q_1 and q_2, the ReLU function introduces rescale symmetry to a network, where the loss function becomes invariant under transformation $(q_1, q_2) \to (aq_1, a^{-1}q_2)$ for a scalar a [2]. This is because scalar multiplication commutes with the ReLU function $a^{-1}q_2\text{ReLU}(aq_1x) = q_2\text{ReLU}(q_1x)$, which makes the activation of the second layer is invariant under the rescale transformation. Notably, once the ReLU function has gained attention with its empirical success in 2011 [3], the ReLU activation has rapidly replaced previously popular

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

hyperbolic tangent and sigmoid functions without such symmetry. Another important example is the batch normalization [4] defined as $\mathrm{BN}(qx) = \frac{qx - \mathrm{E}[qx]}{\sqrt{\mathrm{Var}[qx]}}$. Batch normalization was originally introduced [4] to stabilize forward propagation of activations through deep neural networks and has become an essential component of almost every state-of-the-art deep neural network. Batch normalization layer has scale symmetry $\mathrm{BN}(aqx) = \mathrm{BN}(qx)$, since both numerator and denominator of the batch normalization function is proportional to the scale of weights $|q|$.

Understanding deep learning requires rethinking symmetry of the learning dynamics. What does the symmetry of the loss function in parameter space imply about the symmetry of the learning dynamics? In physical systems, Fig. 1 (a), symmetries of the potential function in space usually imply the equivariance of the dynamics of the coordinates, where a solution of the Newton's equation of motion (grey) stays to be a solution under the symmetry transformations (red). This further implies that the time evolution of the potential energy is invariant under symmetry transformations at any point in the dynamics. However, this intuition does not translate to modern deep learning systems. Consider a concrete example of convolutional neural networks, where the activation of the previous layer x and convolutional filter q, together input to the BatchNorm layer $\mathrm{BN}(qx)$. The scale symmetry implies that the output of the BatchNorm $\mathrm{BN}((1+s)qx) = \mathrm{BN}((1+s)qx)$ for a scalar $s \in \mathbb{R}$, and thus the loss function $f((1+s)qx) = f(qx)$ is invariant to the scaling transformation on the convolutional filter w.

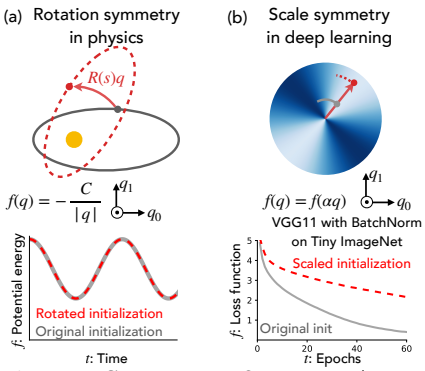

Figure 1: **Symmetry of the loss $\neq$ symmetry of the learning dynamics of the loss.** (a) Physical dynamics in a space with rotational symmetry. (b) Learning dynamics in a parameter space with scale symmetry.

Next, to see if the *learning dynamics* of the loss is invariant under the scale transformation, we train two VGG11 models on Tiny ImageNet with standard initialization (grey) and scaled initialization (red). With the scaled initialization, we multiplied the norm of convolutional filters by 10 at initialization, where this operation does not change the loss. However, this invariance of the loss does not imply the invariance of the *learning dynamics* of the loss as in the bottom of Fig. 1(b). Overall, understanding symmetry of the learning dynamics requires a new theoretical perspective.

Variational perspective on symmetries in deep learning dynamics. In nature, variational formulations of physics based on the Lagrangian and Hamiltonian provide a unifying principle for the role of symmetry in dynamical systems. However, there is still a gap between the tools and concepts in physics and the symmetries in deep learning architectures. We begin by identifying the root cause of this gap is the learning rule, namely the fact that the continuous time limit of stochastic gradient descent (SGD) even with momentum is the first order differential equation in time, gradient flow, $\frac{dq}{dt} = -\nabla f(q)$, unlike physical dynamics governed by the Netwon's second law $\frac{d^2q}{dt^2} = -\nabla f(q)$ which is second-order in time. We bridge this gap by realizing that the effect of discretization η, is to introduce effective "mass" into the continuous-time description of the discrete dynamics. Leveraging this new connection, we make following contributions.

Sec. 1. Modeling deep learning dynamics as Lagrangian dynamics in physics. Building on the Lagrangian formulation of accelerated optimizers [5], we show that non-accelerated optimizers also become Lagrangian dynamics in practical settings with a finite learning rate.

Sec. 2. Unlike in physics, kinetic asymmetry is inherent in learning dynamics. We define *kinetic asymmetry*, where the kinetic energy corresponding to the learning rule does not have the same symmetry as the potential energy corresponding to the loss function.

Sec. 3. We derive Noether's learning dynamics induced by the kinetic asymmetry. We generalize Noether's theorem in physics to derive Noether's learning dynamics, which accounts for the unique symmetries of the loss and the non-Euclidean metric used in learning rules.

Sec. 4. The mechanisms of symmetry breaking enables efficient and successful deep learning. We leverage Noether's learning dynamics to exactly solve for the time-evolution of the effective learning rate for networks with normalization layers. We then find that the dynamics of the Noether charge adaptively stabilizes the effective learning rate in the same manner with an explicit adaptive optimizer (RMSProp [6]).

1 Unified Lagrangian description of learning dynamics

We begin by reviewing the Lagrangian formulation of mechanics in physics and its recent application to study accelerated methods in convex optimization [7, 8]. We then identify the path to connect the Lagrangian formulation and symmetries in deep learning dynamics.

Variational formulation of classical mechanics. Consider a simple setup in classical mechanics, where a particle of mass m at position q moves in potential energy $f(q)$. The dynamics is governed by Newton's equation of motion $m\ddot{q} = -\nabla_q f$, where $\ddot{q} = \frac{d^2q}{dt^2}$ is the acceleration. Now consider the function $\mathcal{L}(q, \dot{q}, t) = \frac{1}{2}m\dot{q}^2 - f(q)$, called Lagrangian, which represents the difference between the kinetic energy $T = \frac{1}{2}m\dot{q}^2$ and the potential energy $V = f(q)$ of the particle. Given the Lagrangian of a system, the principle of least action states that the time evolution of $q(t)$ is such that it minimizes the functional $S[q] = \int_{t_0}^{t_1} \mathcal{L}(q, \dot{q}, t)dt$ called action. The stationary condition for the true trajectory $\frac{\delta S}{\delta q(t)} = 0$ is equivalent to the Euler-Lagrange equation,

$$\frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{q}}\right) = \frac{\partial \mathcal{L}}{\partial q}. \tag{1}$$

Substituting the Lagrangian into the Euler-Lagrange equation, we obtain the Newton's equation of motion $m\ddot{q} = -\nabla_q f$ confirming the consistency of the variational formulation.

Damping in optimization. One of the universal features of learning dynamics is damping, a term $\dot{q} = \frac{dq}{dt}$ that is proportional to the velocity of the parameter vector q. Intuitively, damping is need for any optimization to converge, since without damping the learning dynamics conserves total energy continuing to move. We can incorporate the damping by introducing explicit time-dependence to the Lagrangian [9] $\mathcal{L}(q, \dot{q}, t) = e^{\frac{\mu}{m}t}(\frac{1}{2}m\dot{q}^2 - f(q))$, where μ is the damping (or friction) coefficient. Substituting this Lagrangian into the Euler-Lagrange equation and dividing both sides by $e^{\frac{\mu}{m}t}$, we obtain the equation of motion of a damped particle $m\ddot{q} + \mu\dot{q} = -\nabla_q f$ as intended.

Non-Euclidean geometry in optimization. Any dynamics has its own underlying geometry. Consider the gradient descent $q(t+\eta) = q(t) - \eta\nabla f(q)$ as an example. A step of the gradient descent update is equivalent to solving a minimization problem $q(t+\eta) = \arg\min_{q'}\left[\langle \nabla f(q(t)), q'\rangle + \eta^{-1}\frac{1}{2}|q' - q(t)|^2\right]$ trading off the costs of the loss and the squared Euclidean distance. However, the local geometry of an optimization problem can be non-Euclidean. Mirror descent $q(t+\eta) = \arg\min_{q'}\left[\langle \nabla f(q(t)), q'\rangle + \eta^{-1}D_h(q', q(t))\right]$ generalizes the geometry by replacing the Euclidean distance by the Bregman divergence with a distance-generating function $h(x)$, which is defined as $D_h(y, x) = h(y) - h(x) - \langle \nabla h(x), y - x\rangle$. Indeed, a distance generating function $h_E(x) = \frac{1}{2}|x|^2$, simplifies the Bregman divergence to be the squared distance $D_E(y, x) = \frac{1}{2}|x - y|^2$ encapsulating gradient descent. Beyond gradient descent, the Bregman divergence encapsulates an array of non-Euclidean optimization methods represented by the natural gradient descent [10] generated from the Kullback-Leibler divergence.

The Bregman Lagrangian. By combining these features, Wibisono, Wilson, and Jordan have introduced the Bregman Lagrangian [5, 8] defined as follows,

$$\mathcal{L}(q, \dot{q}, t) = e^{\alpha_t + \gamma_t}\left(D_h(q + e^{-\alpha_t}\dot{q}, q) - e^{\beta_t}f(q)\right), \tag{2}$$

where q is a vector of model parameters and $\dot{q} \equiv \frac{d}{dt}q$, and t is time. In discrete settings, the continuous time is equivalent to $t = n\eta$ after n iterations of parameter updates with the learning rate (time step size) η. The time-dependent factors α_t, β_t, and γ_t are the algorithmic degrees of freedom for

	α_t	β_t	γ_t	$h(x)$	Euler-Lagrange equation
Nesterov momentum	$\log 2 - \log t$	$2\log t + \log 1/4$	$2\log t$	$\frac{1}{2}\lvert x\rvert^2$	$\ddot{q} + 3/t + \nabla f = 0$
Damped physical motion	$-\log m$	$\log m$	$\frac{\mu}{m}t$	$\frac{1}{2}\lvert x\rvert^2$	$m\ddot{q} + \mu\dot{q} + \nabla f(q) = 0$
Natural gradient flow	$-\log m$	$\log m$	$\frac{\mu}{m}t$	$h(x)$	$\dot{q} + F^{-1}\nabla f(q) = 0\ (m \to 0)$
GD + momentum	$-\log\frac{\eta(1+\beta)}{2}$	$\log\frac{\eta(1+\beta)}{2}$	$\frac{2(1-\beta)}{\eta(1+\beta)}t$	$\frac{1}{2}\lvert x\rvert^2$	$\frac{\eta}{2}(1+\beta)\ddot{q} + (1-\beta)\dot{q} + \nabla f = 0$

Table 1: The Bregman Lagrangian [5, 8] provides a unified description of a family of learning rules with time dependent parameters α_t, β_t, γ_t and metric $h(x)$ which provide the algorithmic degrees of freedom.

the Bregman-Lagrangian to encompass a family of learning rules including Nesterov's accelerated gradient methods [11, 7], natural gradient flow [10], and gradient descent with learning rate η and momentum β [12, 13, 14] as summarized in table 1. This unified property of the Bregman-Lagrangian makes the rest of our analysis generalizable to all these learning rules.

Gradient descent becomes Lagrangian dynamics with a finite learning rate. In the continuous-time limit of diminishing learning rate $\eta \to 0$, gradient descent $q(t+\eta) = q(t) - \eta g(t)$ becomes a first order differential equation called gradient flow $\dot{q} = -g$. Unlike Nesterov's momentum [7], the trajectory of gradient descent even with heavy ball momentum β as applied in deep learning [12] follows gradient flow just rescaling time $(1-\beta)\dot{q} = -g$. Here, we apply modified equation analysis [13, 14] to incorporate the effect of a finite learning rate and leverage the fact that the discrete steps of gradient descent closely follow the continuous-time trajectory of the second-order differential equation in time. We can informally see this by Taylor expanding an update with respect to a small learning rate $q(t+\eta) = q + \eta\dot{q} + \frac{\eta^2}{2}\ddot{q} + O(\eta^3)$ and then plugging this expression to the gradient descent updates $\frac{1}{\eta}(q(t+\eta) - q(t)) = -g(t)$. We keep the first order in η and obtain a second-order differential equation $\frac{\eta}{2}\ddot{q} + \dot{q} = -g$ whose continuous-time limit recovers gradient flow $\dot{q} = -g$ as expected. Performing a similar analysis [13, 14] accounting for the effects of the heavy ball momentum with weight decay k yields a second-order differential equation,

$$\frac{\eta}{2}(1+\beta)\ddot{q} + (1-\beta)\dot{q} + \nabla f(q) + kq = 0. \tag{3}$$

Indeed, we can connect this second-order dynamics to the Bregman Lagrangian by considering parameters $[\alpha_t = -\log\frac{\eta(1+\beta)}{2}, \beta_t = \log\frac{\eta(1+\beta)}{2}, \gamma_t = \frac{2(1-\beta)}{\eta(1+\beta)}t, h(x) = \frac{1}{2}|x|^2]$. The Lagrangian describing the learning dynamics under gradient descent with a finite learning rate η, heavy ball momentum β, the loss function $f(q)$, and weight decay constant k is

$$\mathcal{L}(q,\dot{q},t) = e^{\frac{2(1-\beta)}{\eta(1+\beta)}t}\left[\frac{\eta(1+\beta)}{4}|\dot{q}|^2 - \left(f(q) + \frac{k}{2}|q|^2\right)\right]. \tag{4}$$

2 Kinetic asymmetry in learning

In the previous section, we introduced the Lagrangian formulation of learning dynamics, in which the learning rule corresponds to the kinetic energy and the loss function corresponds to the potential energy. The kinetic energy of the Bregman Lagrangian,

$$T_h \equiv e^{\alpha_t} D_h(q + e^{-\alpha_t}\dot{q}, q) = e^{\alpha_t}\left(h(q + e^{-\alpha_t}\dot{q}) - h(q) - \langle \nabla h(q), e^{-\alpha_t}\dot{q}\rangle\right), \tag{5}$$

is as important as the loss function, but its symmetry properties are much less discussed. Motivated by the symmetries inherent in modern deep learning architectures, here we investigate if the Bregman kinetic energy T_h describing learning rules respect the same symmetry as the potential (loss) function.

Symmetry of Euclidean kinetic energy. First, we study the symmetry properties of the Euclidean kinetic energy and identify a unique mechanism of *kinetic asymmetry* in learning systems, where the kinetic energy of the Lagrangian does not respect the symmetry of the potential function. Consider a one-parameter family of differentiable maps $q(t) \to Q(q(t), s)$ parameterized by a scalar $s \in \mathbb{R}$ such that $s = 0$ gives an identity $Q(q(t), 0) = q(t)$. For the remainder, we always assume that any derivative of Q is evaluated by the identity $s = 0$. We say a function $f(q)$ possesses a differentiable symmetry if its invariant to the transformation $f(Q(q(t), s)) = f(q)$. The Euclidean kinetic energy,

$$\partial_s T_{\text{Euclidean}} = \frac{e^{-\alpha_t}}{2}\partial_s|\dot{Q}|^2, \tag{6}$$

is symmetric under the differentiable transformation s if $\partial_s T_E = 0$. For spatial translation $Q(q,s) = q + s\hat{n}$ in the direction of a time-independent vector $\hat{n}$, the definition implies that the velocity is invariant under the transformation $\dot{Q} = \dot{q}$. Thus, $\partial_s|\dot{Q}|^2 = \partial_s|\dot{q}|^2 = 0$ indicating that the Euclidean kinetic energy is invariant under spatial translation $\partial_s T_E = 0$. For spatial rotation $Q(q,s) = R(s)q$, the temporal differentiation commutes with the rotation $\dot{Q} = R\dot{q}$. Since the rotation conserves its norm $|\dot{Q}|^2 = \langle R\dot{q}, R\dot{q}\rangle = |\dot{q}|^2$ the Euclidean kinetic energy is invariant under spatial rotation $\partial_s T_E = 0$. For spatial scaling $Q(q,s) = (1+s)q$, the operation also commutes with the temporal

differentiation $\dot{Q} = (1+s)\dot{q}$. However, the norm is no longer conserved and the kinetic energy does not respect scale symmetry $\partial_s |\dot{Q}|^2 = (\partial_s (1+s)^2)|_{s=0} |\dot{q}|^2 = 2|\dot{q}|^2$ even under the Euclidean metric $\partial_s T_E = e^{-2\alpha_t}|\dot{q}|^2 \neq 0$. Thus, the Euclidean kinetic energy is asymmetric to the scale symmetry of the potential function, which is unique to learning systems.

Symmetry of non-Euclidean kinetic energy. Next, we consider settings with non-Euclidean metrics and find that the kinetic energy does not respect any of the symmetries in general. The Taylor expansion of the Bregman kinetic energy with respect to the first term $h(q + e^{-\alpha_t}\dot{q})$ in Eq.5 gives

$$T_h = (e^{-\alpha_t}/2) \left\langle \dot{Q}, \nabla^2 h(Q) \dot{Q} \right\rangle + O\left((e^{-\alpha_t})^2 \right). \tag{7}$$

The reason why the Euclidean kinetic energy depends on the transformation s only through the squared norm of velocity $|\dot{Q}|^2$ was that the Hessian of the Euclidean metric is identity $\nabla^2 \frac{1}{2}|x|^2 = I$ and the higher order derivatives are zero. However, with non-Euclidean metrics, such simplifications special to the Euclidean case do not apply in general, and the Bregman kinetic energy does not respect these symmetries without special care. The natural gradient [10] is an elegant example of a non-Euclidean optimization method, which has invariance properties by construction. While the invariance properties of the natural gradient descent can be broken due to discretization [15], the fast and accurate approximation of natural gradients in modern large-scale neural networks is an active area of research [16, 17].

Overall, we have identified and defined *kinetic asymmetry* unique to learning systems that motivate us to generalize established concepts in physics next.

3 Noether's learning dynamics

Here we generalize the Noether's theorem to account for this kinetic asymmetry due to symmetry unique to learning systems and non-Euclidean geometry of optimization. As a result, we derive Noether's learning dynamics, a unified equality that holds for any combination of symmetry and learning rules.

Noether's theorem. A general non-restricted form of Noether's theorem relates the dynamics of Noether charge $\langle \partial_{\dot{q}} \mathcal{L}, \partial_s Q \rangle$ to the change of Lagrangian at infinitesimal transformation $\partial_s \mathcal{L}$ as

$$\tfrac{d}{dt} \langle \partial_{\dot{q}} \mathcal{L}, \partial_s Q \rangle = \partial_s \mathcal{L} = \partial_s T - \partial_s V, \tag{8}$$

where T is the kinetic energy and V is the potential energy. We can derive this relationship as $\partial_s \mathcal{L} = \langle \partial_q \mathcal{L}, \partial_s Q \rangle + \langle \partial_{\dot{q}} \mathcal{L}, \partial_s \dot{Q} \rangle = \frac{d}{dt} \langle \partial_{\dot{q}} \mathcal{L}, \partial_s Q \rangle$, where we plugged in the Euler-Lagrange equation $\partial_q \mathcal{L} = \frac{d}{dt}(\partial_{\dot{q}} \mathcal{L})$ and performed integration by parts. A common application of Noether's theorem in physics is in its restricted form, where a certain symmetry is assumed for the whole Lagrangian including both kinetic and potential energies. For example, if we assume symmetry of the Lagrangian due to homogeneity or isotropy of space, Noether's theorem $\frac{d}{dt} \langle \partial_{\dot{q}} \mathcal{L}, \partial_s Q \rangle = \partial_s \mathcal{L} = 0$ directly implies the conservation of momentum or angular momentum. However, as we have seen, the Bregman kinetic energy for learning systems is often asymmetric to the symmetry of the potential function. Thus, this restricted form of Noether's theorem does not generally apply in learning systems.

Noether's learning dynamics. Here, we derive general equality describing the kinetic-asymmetry-induced dynamics of the Noether charge. If the Bregman kinetic energy does not respect the symmetry of the loss function, the Eq.8 becomes

$$\frac{d}{dt} \langle \partial_{\dot{q}} \mathcal{L}, \partial_s Q \rangle = \partial_s T_h = e^{\alpha_t + \gamma_t} \partial_s D_h(Q + e^{-\alpha_t}\dot{Q}, Q), \tag{9}$$

and the kinetic asymmetry $\partial_s T_h$ induces motion of the Noether charge. By evaluating this form of Noether's theorem with the Bregman Lagrangian, we obtain the *Noether's learning dynamics*,

$$\tfrac{d}{dt} \overbrace{\langle \Delta_h, \partial_s Q \rangle}^{\text{Noether charge}} + \overbrace{\dot{\gamma}_t \langle \Delta_h, \partial_s Q \rangle}^{\text{dissipation}} = \overbrace{\langle \Delta_h, \partial_s \dot{Q} \rangle}^{\text{dynamic asymmetry}} + \overbrace{e^{\alpha_t} \langle \Delta_h - e^{-\alpha_t} \nabla^2 h(q) \dot{q}, \partial_s Q \rangle}^{\text{non-Euclidean metric}}, \tag{10}$$

$$\Delta_h(q, \dot{q}, \alpha_t) \equiv \nabla h(q + e^{-\alpha_t}\dot{q}) - \nabla h(q). \tag{11}$$

Each term representing Noether's learning dynamics has an intuitive meaning. As a base case, recall the conventional setting in classical mechanics, where the whole Lagrangian is symmetric, there is no dissipation, and the metric is Euclidean. In this case, the Noether charge is the inner product of momentum and the generator of the symmetry transformation $\langle e^{-\alpha_t}\dot{q}, \partial_s Q\rangle$. The first term of the Noether's learning dynamics (Eq.10) $\langle \Delta_h, \partial_s Q\rangle$ is a geometrical generalization of this conventional Noether charge $\langle e^{-\alpha_t}\dot{q}, \partial_s Q\rangle$ to encompass non-Euclidean metrics. In the Euclidean case, we can see that $\Delta_E = e^{-\alpha_t}\dot{q}$ and the term reduces to the conventional Noether charge $\langle \Delta_E, \partial_s Q\rangle = \langle e^{-\alpha_t}\dot{q}, \partial_s Q\rangle$ as expected. The second term $\dot{\gamma}_t\langle \Delta_h, \partial_s Q\rangle$ represents the contribution of the dissipation and the term becomes zero when γ_t is constant and there is no dissipation. The third term of Eq.10 $\langle \Delta_h, \partial_s \dot{Q}\rangle$ represents the dynamic asymmetry due to symmetries unique to learning systems. Under the Euclidean metric, this term is proportional to $\langle \dot{q}, \partial_s \dot{Q}\rangle$. For spatial translation, the transformation $Q = q + s\hat{n}$ implies $\partial_s \dot{Q} = 0$ and the dynamic asymmetry term is zero $\langle \Delta_h, 0\rangle = 0$. Similarly, in the case of spatial rotation, the transformation implies $Q = R(s)q$, and the dynamic asymmetry term is zero $\langle R|_{s=0}\dot{q}, (\partial_s R)\dot{q}\rangle = \langle \dot{q}, (\partial_s R)\dot{q}\rangle = 0$. Here, we used the fact that the generator of rotation is a skew-symmetric matrix. However, for scale transformation $Q = (1+s)q$ with $\partial_s \dot{Q} = \dot{q}$ unique to learning systems, this kinetic asymmetry term is proportional to $|\dot{q}|^2$ and thus present. The fourth term of Eq.10 $e^{\alpha_t}\langle \Delta_h - e^{-\alpha_t}\nabla^2 h(q)\dot{q}, \partial_s Q\rangle$ represents the contribution of the non-Euclidean geometry and is zero under the Euclidean metric $h(x) = \frac{1}{2}|x|^2$.

Noether's theorem learning dynamics gradient flow. Next, we show that in the over-damped limit of gradient flow, the conventional Noether charge $\langle \Delta_h, \partial_s Q\rangle$ itself, rather than its time-derivative becomes zero. Consider Eq.10 in the setting of accelerated natural gradient flow, where $[\alpha_t = -\log m, \beta_t = \log m, \gamma_t = \frac{\mu}{m}t]$,

$$m\frac{d}{dt}\langle \Delta_h, \partial_s Q\rangle + \mu\langle \Delta_h, \partial_s Q\rangle = m\partial_s T. \tag{12}$$

Without the kinetic asymmetry $\partial_s T = 0$,, the Noether's dynamics describes how the previously conserved quantity decays exponentially with damping $\langle \Delta_h, \partial_s Q\rangle = e^{-\frac{\mu}{m}t}$ as in Fig. 2.

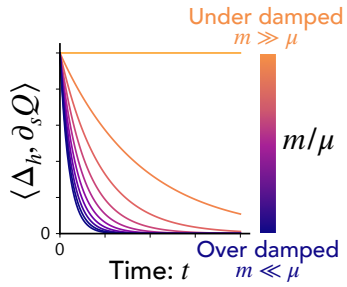

Figure 2: **Noether's theorem with damping.** In the under-damped limit $m/\mu \gg 1$, the Noether charge $\langle \Delta_h, \partial_s Q\rangle$ becomes a conserved quantity in time. However, in the over-damped limit $m \to 0$, the Noether charge exponentially diminishes to zero.

In the mass-less limit $m \to 0$ with a fixed friction μ, the terms proportional to m all vanish and we get

$$0 = \langle \Delta_h, \partial_s Q\rangle = \langle \nabla h(q + e^{-\alpha_t}\dot{q}) - \nabla h(q), \partial_s Q\rangle, \tag{13}$$

which states that the conventional Noether charge itself is zero, rather than its time derivative. Furthermore, in the Euclidean settings, the result simplifies to $\langle \dot{q}, \partial_s Q\rangle$. As a physical analogy, this means that quantities such as momentum or angular momentum becomes zero in the over-damped limit. In machine learning, this formula unifies an array of conservation properties noticed under gradient flow [4, 18, 19, 14] with a formal theoretical connection to Noether's theorem.

Overall, just like how Noether's theorem [20] unified an array of conservation laws and provided a theoretical foundation to discover new ones in physics, we have unified conservation laws previously observed in learning systems [18, 19, 14] and generalized these results for any combination of differentiable symmetries in neural network architectures and learning rules (e.g., natural gradient descent, Nesterov's accelerated gradient descent, accelerated mirror descent, and cubic-regularized Newton's method) with or without the kinetic asymmetry.

4 Symmetry breaking mechanisms underlying success in deep learning

So far, we have harnessed the Lagrangian formulation to characterize a new kind of symmetry breaking mechanism, where the kinetic energy of learning breaks the symmetry of the loss function. Here, we apply our Noether's learning dynamics to a concrete setting of deep learning models with BatchNorm layers and find that this identified mechanism plays a beneficial role in learning.

Euler-Lagrange equations in scale symmetric potential. We begin by developing the mechanics of a particle with mass m, friction μ, and spring constant k, moving in a potential function $f(q)$ with

scale symmetry $f((1+s)q) = q$. We investigate this physical system because the learning dynamics of the most successful deep learning systems are exactly analogous to this system. For example, BatchNorm layers [4], ubiquitous in modern successful deep learning architectures [21], introduce scale symmetry in the parameter space q of the potential (loss) function and are often combined with (stochastic) gradient descent with a finite learning rate η, momentum β, and weight decay k. As derived in Eq.4, the Lagrangian for this learning system is $\mathcal{L} = e^{\frac{\mu}{m}t}\left(\frac{m}{2}|\dot{q}|^2 - \left(f(q) + \frac{k}{2}|q|^2\right)\right)$ where $m = \frac{\eta(1+\beta)}{2}$ and $\mu = 1-\beta$. To take advantage of the scale symmetry of the potential function, we perform a coordinate transformation $q = r\hat{q}$ separating the radial norm r from the direction vector $\hat{q}$, and add a Lagrange multiplier to enforce a unit norm constraint $C(\hat{q}) = |\hat{q}|^2 - 1 = 0$ to the direction vector. Lagrangian upon this coordinate transformation is

$$\mathcal{L}_c = e^{\frac{\mu}{m}t}\left[\frac{m}{2}\dot{r}^2 + \left(\frac{m}{2}\left|\dot{\hat{q}}\right|^2 - \frac{k}{2}\right)r^2 - f(\hat{q})\right] + \lambda(t)\left(|\hat{q}|^2 - 1\right). \tag{14}$$

The Lagrangian formulation has the powerful covariant property that the form of the Euler-Lagrange equations is invariant under coordinate transformations. Taking advantage of this fact, we derive the Euler-Lagrange equation for the radial norm $\partial_r \mathcal{L} = \frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{r}}\right)$ as$m\ddot{r} + \mu\dot{r} = \left(m|\dot{\hat{q}}|^2 - k\right)r$. Similarly, we derive the Euler-Lagrange equation for the directional vector $\partial_{\hat{q}}\mathcal{L} = \frac{d}{dt}\left(\frac{\partial \mathcal{L}}{\partial \dot{\hat{q}}}\right)$ as

$$m\ddot{\hat{q}} + \mu\dot{\hat{q}} = -\frac{1}{r^2}\hat{g}. \tag{15}$$

We assumed that both the mass m and the spring constant k are small and kept only the first order in m and k. This overdamped approximation is valid in practice since we use a small learning rate $\eta \sim 10^{-2}$ and even smaller weight decay $k \sim 10^{-4}$.

Exact solution for the implicit adaptive optimization with BatchNorm. Next, building on our accurate Euler-Lagrange equations modeling the realistic deep learning systems, we establish an exact theoretical analogy between two seemingly unrelated key components of modern deep learning: BatchNorm and adaptive optimizer (RMSProp). Due to the scale symmetry of the network introduced by the BatchNorm layer, the time evolution on the loss landscape depends only on the angular component of the parameter vector $f(q) = f(\hat{q})$. The angular Euler-Lagrange equation, Eq.15, $m\ddot{\hat{q}} + \mu\dot{\hat{q}} = -\hat{g}/r^2(t)$, implies that (i) the time evolution of $f(q)$ for a network with BatchNorm layers trained by stochastic gradient descent is equivalent to (ii) the time evolution of $f(\hat{q})$ for a network with BatchNorm layers with fixed filter norm $\hat{q} = 1$ trained by an adaptive optimizer which adaptively scales the gradient norm by a factor of $r^2(t)$.

Strikingly, Noether's learning dynamics describes the motion of this adaptive scaling factor $r^2(t)$. When $\alpha_t = -\log m$, and $h(x) = \frac{1}{2}|x|^2$, the conventional Noether charge is $\langle e^{-\alpha_t}\dot{q}, \partial_s Q\rangle = m\langle \dot{q}, q\rangle = \frac{1}{2}m\partial_t r^2$, and from Eq.10 the Noether-Bregman dynamcis including the effect of weight decay k is

$$m\partial_t^2 r^2 + \mu\partial_t r^2 = -2kr^2 + \frac{2m}{\mu^2 r^2}|\hat{g}|^2, \tag{16}$$

where we kept the first order terms in m and k. By solving this Bernoulli's differential equation, we can describe the dynamics of the parameter norm squared $r^2(t)$ as a function of the normalized gradients $|\hat{g}| = |\partial f/\partial q|_{q=\hat{q}}|$ as, $r^2(t) = \sqrt{\frac{4m}{\mu^3}\int_0^t e^{-\frac{4k}{\mu}(t-\tau)}|\hat{g}(\tau)|^2 d\tau + e^{-\frac{4k}{\mu}t}r^4(0)}$. By substituting hyper-parameters used in realistic deep-learning settings, we get

$$\text{BatchNorm: } r^2(t) = \sqrt{\frac{2\eta(1+\beta)}{(1-\beta)^3}\int_0^t e^{-\frac{4k}{1-\beta}(t-\tau)}|\hat{g}(\tau)|^2 d\tau + e^{-\frac{4k}{1-\beta}t}r^4(0)}. \tag{17}$$

Each term of this time-evolution of implicit adaptive learning rate has an intuitive meaning. The first integral term keeps track of the recent history of the magnitude of gradient norms $|\hat{g}(\tau)|^2$ that the filters with fixed unit norm receive. The weight decay k controls the time-scale of the short-term memory of the accumulated gradient norms through the exponential kernel $e^{-\frac{4k}{1-\beta}(t-\tau)}$, and when $k = 0$ all the gradients will be accumulated resembling AdaGrad optimizer [22]. Notably, this integral term is proportional to the learning rate η implying that this integral term appears with the kinetic symmetry breaking with finite learning rate. The second term represents exponentially decaying

memory of initialization and finite filter norm at initialization $r^4(0)$ prevents instability of learning dynamics avoiding division by zero.

Continuous-time description of RMSProp. So how does this adaptive scaling of the gradient step sizes compare with hand-designed adaptive optimizers successful in deep learning [22, 23, 24]? Here, we develop a continuous-time model of the explicit adaptive optimizer (RMSProp) and discover that the functional form of BatchNorm's effective learning rate schedule Eq. 17 exactly agrees with that of RMSProp. The RMSProp algorithm is a recursive update rule expressed as $q_{n+1} = q_n - \frac{\eta}{\sqrt{G_n}} g_n$, where the gradient is scaled by a factor of $\sqrt{G_n}$. The factor G_n keeps track of the history of the gradient norms $G_{n+1} = \rho G_n + (1-\rho)|g_n|^2$, where ρ is a hyperparameter. By solving an analogous continuous-time dynamics $\eta \frac{dG}{dt} = -(1-\rho)G(t) + (1-\rho)|g|^2$ of the adaptive scaling factor of RMSProp, we obtain

$$\text{RMSProp: } \sqrt{G(t)} = \sqrt{\frac{1-\rho}{\eta} \int_0^t e^{-\frac{1-\rho}{\eta}(t-\tau)} |g(\tau)|^2 d\tau + e^{-\frac{1-\rho}{\eta}t} G(0)}. \tag{18}$$

Strikingly, we find that this functional form induced by RMSProp exactly matches with the implicit adaptive learning rate schedule Eq.17 due to normalization layers suggesting potential benefits of the implicit adaptive optimization induced by BatchNorm.

In summary, our theory has yielded the following new insights.

1. The optimization geometry of SGD with a finite step size η, momentum β, and weight decay k together breaks the scale symmetry of the loss (e.g., with BatchNorm layer) and generates an implicit mechanism of adaptive optimization akin to RMSProp.

2. The implicit adaptive optimization mechanism exists when the learning dynamics is in the underdamped regime $m/\mu \propto \eta/(1-\beta) \gg 0$ and thus requires the learning rate to be finite.

3. Momentum significantly amplifies the effect of adaptive optimization by $\frac{1+\beta}{(1-\beta)^3} = 1900$ folds as in Eq. 17 even in the standard setting of $\beta = 0.9$.

4. For scale-invariant parameters, symmetry breaking due to weight decay $1-k$ plays a role similar to the discount factor ρ for the cumulative gradient norm in RMSProp.

Finally, we empirically validate these predictions by training convolutional neural networks with BatchNorm (VGG11) on a large data set (Tiny-ImageNet) with varying hyper-parameters (see Fig. 3). In addition to the standard training with "unconstrained filter norm" (pink), we perform ablation experiments with "constrained filter norm" (blue). As a concrete example, consider a convolutional filter $q \in \mathbb{R}^n$ that comes into the BatchNorm layer. Since the output of BatchNorm is scale-invariant, the loss is invariant under the scaling transformation of the filter norm as well $f((1+s)q) = f(q)$. To test the presence and benefits of implicit adaptive optimization to the learning dynamics of the loss, we freeze the dynamics of the filter norm by repeatedly performing an operation $q(t) \leftarrow \frac{q(t)}{|q(t)|}|q(0)|$ after each step of optimization. This enforces the filter norm to be fixed $|q(t)| = |q(0)|$ throughout training for any t. When trained with a constant low learning rate (~ 0.001), the final test accuracy (A-1) and loss dynamics (A-2) of models trained with unconstrained filter norm (pink) and constrained filter norm (blue) are almost identical. However, in high learning rate regime ($\eta \sim 0.03$), the models trained with unconstrained filter norm (pink) largely outperforms the models trained with constrained filter norm (blue) validating the existence and benefits of implicit adaptive optimization. Similarly, we confirm our prediction that the presence of momentum $\beta \sim 0.9$ is essential to amplify the effects of implicit adaptive optimization, as seen in test accuracy (B-1) and loss dynamics (B-2). Finally, we validate the benefits of symmetry breaking due to weight decay k which acts in analogy with the discounting factor ρ for the cumulative gradient norms of RMSProp. Indeed, the final test accuracy (C-1) and the learning dynamics of the loss (C-2) are both enhanced by the presence of weight decay k. Overall, the elements of the modern optimizer - finite learning rate, momentum, and weight decay - each play an important role in breaking the scale symmetry of the loss function, providing an mechanism of implicit adaptive optimization, a key to successful deep learning.

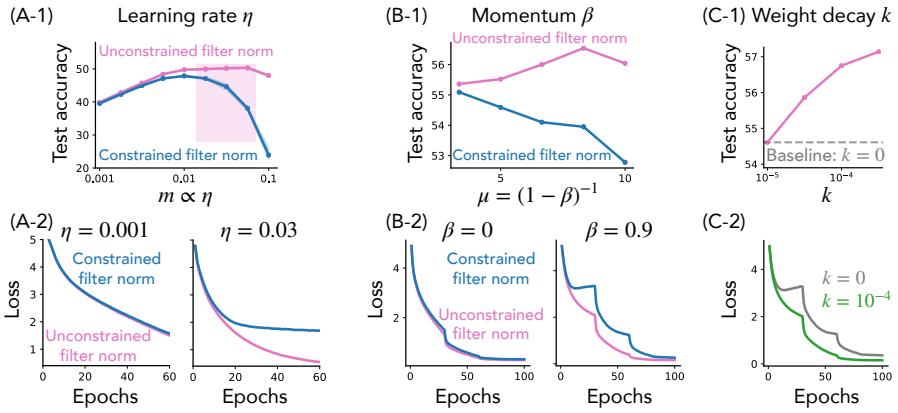

Figure 3: **Broken-symmetry induced dynamics of the filter norm is essential to successful learning. (VGG11 with BatchNorm on Tiny ImageNet)** (A-1) The final test accuracy and (A-2) the loss dynamics of models trained at various constant learning rates η. (B-1) The final test accuracy and (B-2) the loss dynamics of models when trained with various momentum β with standard learning rate drops. In agreement with our theory, the advantage of implicit adaptive optimization is present when trained with high learning rates and high momentum. (C-1) The final test accuracy and (C-2) the loss dynamics when the model is trained with various weight decay k, validating its benefits.

5 Related works

Geometry of optimization. The geometric property of optimization has a long history of study [10, 25]. For example, natural gradient [10] has attractive invariance properties, and despite it being expensive to compute, various methods have been developed to efficiently approximate natural gradient [10] while retaining the original benefits [26, 27, 16, 28]. In deep learning, we commonly use gradient-based optimization methods [12, 22, 23, 24]. A pioneering work [2] has identified that the gradient descent does not respect the invariance of the deep network functions, followed by works proposing to solve such discrepancies [29, 30]. While above works have always tried to remove such broken symmetries, we hypothesized and proved that the broken-symmetry can play an important role in deep learning, thus providing a unique counterexample.

Auto-rate tuning by normalization layers. BatchNorm [4] and its variants [31, 32] are essential to train *deep* neural network models [21]. An important benefit of BatchNorm is to enable stable training with large learning rates [33]. A pioneering work [34] has rigorously studied the benefits of monotonically decreasing effective learning rate in simple setting without momentum or weight decay. Another line of works have noticed the qualitative role of weight decay to enforce smaller norms [35, 36, 37], and thus higher learning rate whose theoretical treatment has lead counter-intuitive exponential learning rate schedule [38]. In the present analysis, we leveraged the Lagrangian formulation to handle the complexity of modern practical optimizers and derived an exact solution describing the time-evolution of the effective learning rate in the presence of momentum and weight decay. By developing continuous-time description of adaptive optimizers in parallel, we discovered that the functional form of the solution exactly matches with that of RMSProp, where momentum significantly amplifies the effect and weight decay acts in exact analogy with the discounting factor.

6 Conclusion.

Despite many attempts to bring symmetry to machine learning systems [1, 2], symmetry breaking has rarely been discussed as a design principle [39], or even treated as an obstacle to successful learning [2, 29, 30]. In this work, we have discovered a novel role of symmetry breaking in learning systems by applying the Lagrangian formulation to modern deep learning. In future works, we can harness the generality of our theory to investigate the dynamics with more advanced learning rules such as natural gradient descent with or without Nesterov's momentum in combination with any differentiable symmetries inherent in deep learning architectures. Exploring the analogous mechanism of implicit adaptive optimization through the interaction of gradient descent and rescale symmetry of the ReLU function may provide new insights.

References

[1] Michael M Bronstein, Joan Bruna, Taco Cohen, and Petar Veličković. Geometric deep learning: Grids, groups, graphs, geodesics, and gauges. *arXiv preprint arXiv:2104.13478*, 2021.

[2] Behnam Neyshabur, Russ R Salakhutdinov, and Nati Srebro. Path-sgd: Path-normalized optimization in deep neural networks. In *Advances in Neural Information Processing Systems*, pages 2422–2430, 2015.

[3] Xavier Glorot, Antoine Bordes, and Yoshua Bengio. Deep sparse rectifier neural networks. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 315–323. JMLR Workshop and Conference Proceedings, 2011.

[4] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[5] Andre Wibisono and Ashia C. Wilson. On accelerated methods in optimization, 2015.

[6] Geoffrey Hinton, Nitish Srivastava, and Kevin Swersky. Neural networks for machine learning lecture 6a overview of mini-batch gradient descent. *Cited on*, 14(8), 2012.

[7] Weijie Su, Stephen Boyd, and Emmanuel Candes. A differential equation for modeling nesterov's accelerated gradient method: Theory and insights. In *Advances in neural information processing systems*, pages 2510–2518, 2014.

[8] Andre Wibisono, Ashia C Wilson, and Michael I Jordan. A variational perspective on accelerated methods in optimization. *proceedings of the National Academy of Sciences*, 113(47):E7351–E7358, 2016.

[9] E Kanai. On the quantization of the dissipative systems. *Progress of Theoretical Physics*, 3(4): 440–442, 1948.

[10] Shun-Ichi Amari. Natural gradient works efficiently in learning. *Neural computation*, 10(2): 251–276, 1998.

[11] Yurii E Nesterov. A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Dokl. akad. nauk Sssr*, volume 269, pages 543–547, 1983.

[12] 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, 2013.

[13] Nikola B. Kovachki and Andrew M. Stuart. Continuous time analysis of momentum methods. *Journal of Machine Learning Research*, 22(17):1–40, 2021. URL `http://jmlr.org/papers/v22/19-466.html`.

[14] Daniel Kunin, Javier Sagastuy-Brena, Surya Ganguli, Daniel LK Yamins, and Hidenori Tanaka. Neural mechanics: Symmetry and broken conservation laws in deep learning dynamics. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=q8qLAbQBupm`.

[15] Yang Song, Jiaming Song, and Stefano Ermon. Accelerating natural gradient with higher-order invariance. In *International Conference on Machine Learning*, pages 4713–4722. PMLR, 2018.

[16] James Martens. New insights and perspectives on the natural gradient method. *arXiv preprint arXiv:1412.1193*, 2014.

[17] Kevin Luk and Roger Grosse. A coordinate-free construction of scalable natural gradient. *arXiv preprint arXiv:1808.10340*, 2018.

[18] Simon S Du, Wei Hu, and Jason D Lee. Algorithmic regularization in learning deep homogeneous models: Layers are automatically balanced. In *Advances in Neural Information Processing Systems*, pages 384–395, 2018.

[19] Hidenori Tanaka, Daniel Kunin, Daniel LK Yamins, and Surya Ganguli. Pruning neural networks without any data by iteratively conserving synaptic flow. *arXiv preprint arXiv:2006.05467*, 2020.

[20] Emmy Noether. Invariante variationsprobleme, math-phys. *Klasse, pp235-257*, 1918.

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

[22] John Duchi, Elad Hazan, and Yoram Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[23] Tijmen Tieleman and Geoffrey Hinton. Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude. *COURSERA: Neural networks for machine learning*, 4(2): 26–31, 2012.

[24] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[25] Shun-ichi Amari. Information geometry. *Japanese Journal of Mathematics*, 16(1):1–48, 2021.

[26] Nicolas Le Roux, Pierre-Antoine Manzagol, and Yoshua Bengio. Topmoumoute online natural gradient algorithm. In *NIPS*, pages 849–856. Citeseer, 2007.

[27] James Martens. Deep learning via hessian-free optimization. In *ICML*, volume 27, pages 735–742, 2010.

[28] Razvan Pascanu and Yoshua Bengio. Revisiting natural gradient for deep networks. *arXiv preprint arXiv:1301.3584*, 2013.

[29] Vijay Badrinarayanan, Bamdev Mishra, and Roberto Cipolla. Understanding symmetries in deep networks. *arXiv preprint arXiv:1511.01029*, 2015.

[30] Minhyung Cho and Jaehyung Lee. Riemannian approach to batch normalization. In *Advances in Neural Information Processing Systems*, pages 5225–5235, 2017.

[31] Tim Salimans and Durk P Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In *Advances in neural information processing systems*, pages 901–909, 2016.

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

[33] Johan Bjorck, Carla Gomes, Bart Selman, and Kilian Q Weinberger. Understanding batch normalization. *arXiv preprint arXiv:1806.02375*, 2018.

[34] Sanjeev Arora, Zhiyuan Li, and Kaifeng Lyu. Theoretical analysis of auto rate-tuning by batch normalization. *arXiv preprint arXiv:1812.03981*, 2018.

[35] Elad Hoffer, Ron Banner, Itay Golan, and Daniel Soudry. Norm matters: efficient and accurate normalization schemes in deep networks. In *Advances in Neural Information Processing Systems*, pages 2160–2170, 2018.

[36] Guodong Zhang, Chaoqi Wang, Bowen Xu, and Roger Grosse. Three mechanisms of weight decay regularization. In *International Conference on Learning Representations*, 2019. URL `https://openreview.net/forum?id=B1lz-3Rct7`.

[37] Twan Van Laarhoven. L2 regularization versus batch and weight normalization. *arXiv preprint arXiv:1706.05350*, 2017.

[38] Zhiyuan Li and Sanjeev Arora. An exponential learning rate schedule for deep learning. *arXiv preprint arXiv:1910.07454*, 2019.

[39] Robert Bamler and Stephan Mandt. Improving optimization for models with continuous symmetry breaking. In *International Conference on Machine Learning*, pages 423–432. PMLR, 2018.

Checklist

The checklist follows the references. Please read the checklist guidelines carefully for information on how to answer these questions. For each question, change the default **[TODO]** to [Yes] , [No] , or [N/A] . You are strongly encouraged to include a **justification to your answer**, either by referencing the appropriate section of your paper or providing a brief inline description. For example:

- Did you include the license to the code and datasets? [Yes] See Section **??**.
- Did you include the license to the code and datasets? [No] The code and the data are proprietary.
- Did you include the license to the code and datasets? [N/A]

Please do not modify the questions and only use the provided macros for your answers. Note that the Checklist section does not count towards the page limit. In your paper, please delete this instructions block and only keep the Checklist section heading above along with the questions/answers below.

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [Yes]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes]
3. If you ran experiments...
   (a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]
   (b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [Yes]
   (c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [Yes]
   (d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]
4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...
   (a) If your work uses existing assets, did you cite the creators? [N/A]
   (b) Did you mention the license of the assets? [N/A]
   (c) Did you include any new assets either in the supplemental material or as a URL? [N/A]
   (d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]
   (e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]
5. If you used crowdsourcing or conducted research with human subjects...
   (a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]
   (b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]
   (c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]