A EXPERIMENTS

We evaluated the momentum decay rule with Adam and SGDM on Residual CNNs, Non Residual CNNS, RNNs, generative models, and Capsule Networks. For CNNs, we used the image classification datasets CIFAR10, CIFAR100 and STL10 datasets. For RNNs, we used the language modeling dataset PTB. For generative modeling, we used the MNIST and CIFAR10 datasets. For Capsule Networks, we used FMNIST. We tried to select a variety of well-known architectures on well-accepted datasets. For each network dataset pair other than NSCN and Capsule Networks, we evaluated Adam, QHAdam, AMSGrad, AdamW, YellowFin, DEMON Adam, AggMo, QHM, DEMON SGDM, SGDM. For adaptive learning rate methods and adaptive momentum methods, we generally perform a grid search over the learning rate. For SGDM, we generally perform a grid search over learning rate and initial momentum. For SGDM, Aggmo, and QHM, we decay the learning rate by 0.1 at 50% and 75% of the total epochs, following the standard in the literature.

A.1 SETUP

We describe the eight test problems in this paper.

- **CIFAR10 - ResNet20.** CIFAR10 contains 60,000 32x32x3 images with a 50,000 training set, 10,000 test set split. There are 10 classes. ResNet20 (He et al., 2016) is an 20 layers deep CNN with skip connections for image classification. Trained with a batch size of 128.
- **TINY IMAGENET - ResNet56.** Tiny ImageNet contains 110,000 64x64x3 images with a 100,000 training set, 10,000 test set split. There are 200 classes. ResNet56 (He et al., 2016) is a 56 layer deep CNN with skip connections for image classification. Trained with a batch size of 128.
- **CIFAR100 - VGG16.** CIFAR100 is a fine-grained version of CIFAR-10 and contains 60,000 32x32x3 images with a 50,000 training set, 10,000 test set split. There are 100 classes. VGG16 (Simonyan & Zisserman, 2014) is a 16 layers deep CNN with extensive use of 3x3 convolutional filters. Trained with a batch size of 128.
- **STL10 - Wide ResNet 16-8.** STL10 contains 1300 96x96x3 images with a 500 training set, 800 test set split. There are 10 classes. Wide ResNet 16-8 (Zagoruyko & Komodakis, 2016) is a 16 layers deep ResNet which is 8 times wider. Trained with a batch size of 64.
- **PTB - LSTM.** PTB is an English text corpus containing 929,000 training words, 73,000 validation words, and 82,000 test words. There are 10,000 words in the vocabulary. The model is stacked LSTMs (Hochreiter & Schmidhuber, 1997) with 2 layers, 650 units per layer, and dropout of 0.5. Trained with a batch size of 20.
- **FMNIST - CAPS.** FMNIST contains 60,000 32x32x1 grayscale images with a 50,000 training set, 10,000 test set split. There are 10 classes of 10 clothing items. Capsule Networks (Sabour et al., 2017) represent Neural Networks as a set of capsules, where each capsule encodes a specific entity or meaning. The activations of capsules depend on comparing incoming pose predictions, as opposed to standard neural networks. The Capsule Network uses 3 iterations in the routing algorithm. Trained with a batch size of 128.
- **MNIST - VAE.** MNIST contains 60,000 32x32x1 grayscale images with a 50,000 training set, 10,000 test set split. There are 10 classes of 10 digits. VAE (Kingma & Welling, 2015) with three dense encoding layers and three dense decoding layers with a latent space of size 2. Trained with a batch size of 100.
- **CIFAR10 - NCSN.** CIFAR10 contains 60,000 32x32x3 images with a 50,000 training set, 10,000 test set split. There are 10 classes. NCSN (Song & Ermon, 2019) is a recent state-of-the-art generative model which achieves the best reported inception score. We compute inception scores based on a total of 50000 samples. Since DEMON depends on a predefined number of epochs, we evaluate inception score at the end of training; otherwise, we follow the exact implementation in and defer details to the original paper.

A.2 METHODS

A.2.1 ADAPTIVE LEARNING RATE

Adam (Kingma & Ba, 2014), as previously introduced in section 2, keeps an exponentially decaying average of squares of past gradients to adapt the learning rate. It also introduces an exponentially decaying average of gradients.

The Adam algorithm is parameterized by learning rate $\eta > 0$, discount factors $\beta_1 < 1$ and $\beta_2 < 1$, a small constant ϵ, and uses the update rule:

$$\begin{aligned}
\mathcal{E}^{g}_{t+1} &= \beta_1 \cdot \mathcal{E}^{g}_{t} + (1-\beta_1)\cdot g_t, \\
\mathcal{E}^{g\circ g}_{t+1} &= \beta_2 \cdot \mathcal{E}^{g\circ g}_{t} + (1-\beta_2)\cdot (g_t \circ g_t), \\
\theta_{t+1,i} &= \theta_{t,i} - \frac{\eta}{\sqrt{\mathcal{E}^{g\circ g}_{t+1,i}+\varepsilon}} \cdot \mathcal{E}^{g}_{t+1,i}, \quad \forall t.
\end{aligned}$$

AMSGrad (Reddi et al., 2019) resolves an issue in the proof of Adam related to the exponential moving average $\mathcal{E}^{g\circ g}_t$, where Adam does not converge for a simple optimization problem. Instead of an exponential moving average, AMSGrad keeps a running maximum of $\mathcal{E}^{g\circ g}$.

The AMSGrad algorithm is parameterized by learning rate $\eta > 0$, discount factors $\beta_1 < 1$ and $\beta_2 < 1$, a small constant ϵ, and uses the update rule:

$$\begin{aligned}
\mathcal{E}^{g}_{t+1} &= \beta_1 \cdot \mathcal{E}^{g}_{t} + (1-\beta_1)\cdot g_t, \\
\mathcal{E}^{g\circ g}_{t+1} &= \beta_2 \cdot \mathcal{E}^{g\circ g}_{t} + (1-\beta_2)\cdot (g_t \circ g_t), \\
\hat{\mathcal{E}}^{g\circ g}_{t+1,i} &= max(\hat{\mathcal{E}}^{g\circ g}_{t,i}, \mathcal{E}^{g\circ g}_{t,i}), \\
\theta_{t+1,i} &= \theta_{t,i} - \frac{\eta}{\sqrt{\hat{\mathcal{E}}^{g\circ g}_{t+1,i}+\varepsilon}} \cdot \mathcal{E}^{g}_{t+1,i}, \quad \forall t,
\end{aligned}$$

where $\mathcal{E}^{g}_{t+1}$ and $\mathcal{E}^{g\circ g}_{t+1}$ are defined identically to Adam.

AdamW (Loshchilov & Hutter, 2017) modifies the typical implementation of weight decay regularization in Adam, by decoupling the weight decay from the gradient update. In particular, L2 regularization in Adam is usually implemented with the below modification where w_t is the rate of the weight decay at time t:

$$g_t = \nabla f(\theta_t) + w_t \theta_t,$$

while AdamW, instead, adjusts the weight decay term to appear in the gradient update:

$$\theta_{t+1,i} = \theta_{t,i} - \eta\Big(\frac{1}{\sqrt{\hat{\mathcal{E}}^{g\circ g}_{t+1,i}+\varepsilon}} \cdot \mathcal{E}^{g}_{t+1,i} + w_{t,i}\theta_{t,i}\Big), \quad \forall t.$$

QHAdam (Quasi-Hyperbolic Adam) (Ma & Yarats, 2018) extends QHM (Quasi-Hyperbolic Momentum), introduced further below, to replace both momentum estimators in Adam with quasi-hyperbolic terms. This quasi-hyperbolic formulation is capable of recovering Adam and NAdam (Dozat, 2016), amongst others.

The QHAdam algorithm is parameterized by learning rate $\eta > 0$, discount factors $\beta_1 < 1$ and $\beta_2 < 1$, $\nu_1, \nu_2 \in \mathbb{R}$, a small constant ϵ, and uses the update rule:

$$\begin{aligned}
\mathcal{E}^{g}_{t+1} &= \beta_1 \cdot \mathcal{E}^{g}_{t} + (1-\beta_1)\cdot g_t, \\
\mathcal{E}^{g\circ g}_{t+1} &= \beta_2 \cdot \mathcal{E}^{g\circ g}_{t} + (1-\beta_2)\cdot (g_t \circ g_t), \\
\hat{\mathcal{E}}^{g}_{t+1} &= (1+\beta_1^{t+1})^{-1} \cdot \mathcal{E}^{g}_{t+1}, \\
\hat{\mathcal{E}}^{g\circ g}_{t+1} &= (1+\beta_2^{t+1})^{-1} \cdot \mathcal{E}^{g\circ g}_{t+1}, \\
\theta_{t+1,i} &= \theta_{t,i} - \eta\left[\frac{(1-\nu_1)\cdot g_t + \nu_1 \cdot \hat{\mathcal{E}}^{g}_{t+1}}{\sqrt{(1-\nu_2) g_t^2 + \nu_2 \cdot \hat{\mathcal{E}}^{g\circ g}_{t+1}} + \epsilon}\right], \quad \forall t,
\end{aligned}$$

Table 6: Additional information hyperparameter tuning details for main optimizers. For SGDM, we extensively tuned the learning rate schedule, including schemes from Huang et al. (2017); Zagoruyko & Komodakis (2016); Hu et al. (2017); Lin et al. (2017); Wang et al. (2017). We attempt decay on plateau with patience in intervals of 5 and decay schedules such as 0.1 at 50% and 75% of total epochs; 0.1 at 25%, 50%, 75%; 0.1 at 33% and 66%; and 0.1 at 10%, 25%, 50%, 75%. A smooth decay schedule (both per epoch and at every 10% of total epochs) to 0.01 (and 0.001) across total epochs and no learning rate decay was also attempted. A total of 10 settings. We tried delaying DEMON till 50% and 75% of epochs, for a total of 3 settings.

	Tuning ranges			
Optimizer	η	β/β_{init}	η Schedule	Decay Schedule
SGDM	Same	Same	10x	-
DEMONSGDM	Same	Same	-	3x

where $\mathcal{E}^g_{t+1}$ and $\mathcal{E}^{g\circ g}_{t+1}$ are defined identically to Adam.

YellowFin. (Zhang & Mitliagkas, 2017) is motivated by robustness properties and analysis of quadratic objectives. For quadratic objectives, the optimizer tunes both the learning rate and the momentum to keep the hyperparameters within a region in which the convergence rate is a constant rate equal to the root momentum. This notion is extended empirically to non-convex objectives. On every iteration, YellowFin optimizes the hyperparameters to minimize a local quadratic optimization. Due to the many details, we defer an indepth explanation to the paper (Zhang & Mitliagkas, 2017).

A.2.2 ADAPTIVE MOMENTUM

AggMo (Aggregated Momentum) (Lucas et al., 2018) takes a linear combination of multiple momentum buffers. It maintains K momentum buffers, each with a different discount factor, and averages them for the update.

The AggMo algorithm is parameterized by learning rate $\eta > 0$, discount factors $\beta \in \mathbb{R}^K$, and uses the update rule:

$$(\mathcal{E}^g_{t+1})^{(i)} = \beta^{(i)} \cdot (\mathcal{E}^g_t)^{(i)} + g_t, \quad \forall i \in [1, K],$$
$$\theta_{t+1,i} = \theta_{t,i} - \eta \left[\frac{1}{K} \cdot \sum_{i=1}^{K} (\mathcal{E}^g_{t+1})^{(i)} \right], \quad \forall t.$$

QHM (Quasi-Hyperbolic Momentum) (Ma & Yarats, 2018) is a weighted average of the momentum and plain SGD. QHM is capable of recovering Nesterov Momentum (Nesterov, 1983), Synthesized Nesterov Variants (Lessard et al., 2016), accSGD (Jain et al., 2017) and others.

The QHM algorithm is parameterized by learning rate $\eta > 0$, discount factor $\beta < 1$, immediate discount factor $\nu \in \mathbb{R}$, and uses the update rule:

$$\mathcal{E}^g_{t+1} = \beta \cdot \mathcal{E}^g_t + (1 - \beta) \cdot g_t,$$
$$\theta_{t+1,i} = \theta_{t,i} - \eta \Big[(1 - \nu) \cdot g_t + \nu \cdot \mathcal{E}^g_{t+1} \Big], \quad \forall t.$$

A.3 TUNING OF SGDM AND DEMON SGDM

A.4 OPTIMIZER HYPERPARAMETERS

Table 7: Best parameters for CIFAR-10 with ResNet-20.

Optimization method	epochs	η	other parameters
Adam	30	0.001	$\beta_1 = 0.9, \beta_2 = 0.999$
Adam	75	0.001	
Adam	150	0.001	
Adam	300	0.0003	
AMSGrad	30	0.001	$\beta_1 = 0.9, \beta_2 = 0.999$
AMSGrad	75	0.001	
AMSGrad	150	0.001	
AMSGrad	300	0.001	
QHAdam	30	0.001	$\nu_1 = 0.7, \nu_2 = 1.0, \beta_1 = 0.9, \beta_2 = 0.99$
QHAdam	75	0.0003	
QHAdam	150	0.0003	
QHAdam	300	0.0003	
AdamW	30	0.001	$\beta_1 = 0.9, \beta_2 = 0.999, wd = 0.0001$
AdamW	75	0.001	
AdamW	150	0.001	
AdamW	300	0.001	
YellowFin	30	0.001	$\beta_1 = 0$
YellowFin	75	0.001	
YellowFin	150	0.001	
YellowFin	300	0.001	
DEMON Adam	30	0.0001	$\beta_{\text{init}} = 0.9, \beta_2 = 0.999$
DEMON Adam	75	0.0001	
DEMON Adam	150	0.0001	
DEMON Adam	300	0.0001	
AggMo	30	0.03	$\beta = [0, 0.9, 0.99]$
AggMo	75	0.03	
AggMo	150	0.03	
AggMo	300	0.03	
QHM	30	3.0	$\nu = 0.7, \beta = 0.999$
QHM	75	3.0	
QHM	150	3.0	
QHM	300	1.0	
DEMON SGDM	30	0.03	$\beta_{\text{init}} = 0.95$
DEMON SGDM	75		
DEMON SGDM	150		
DEMON SGDM	300		
SGDM	30	0.3	$\beta_1 = 0.9$
SGDM	75	0.1	$\beta_1 = 0.9$
SGDM	150	0.3	$\beta_1 = 0.9$
SGDM	300	0.1	$\beta_1 = 0.9$

B ABLATION STUDY

See Table 14 for an ablation study with respect to the parameter T, which defines the proportion of epochs at which DEMON begins. There exists a small difference.

	β=.9			β=.95			β=.9			β=.95		
T	0	.5	.75	0	.5	.75	0	.5	.75	0	.5	.75
DemonSGDM	30.02	29.68	29.32	29.80	29.02	28.88	9.53	9.56	8.59	10.34	9.60	9.20
DemonAdam	28.84	28.63	28.33	28.53	28.97	29.34	9.03	8.96	8.49	9.96	9.73	9.42

Table 14: Test error for 150 epochs VGG16-CIFAR100 (leftmost 6 cols), and 150 epochs RN20-CIFAR10. η fixed across T per setting.

Table 8: Best parameters for CIFAR-100 with VGG-16.

Optimization method	epochs	η	other parameters
Adam	75	0.0003	$\beta_1 = 0.9, \beta_2 = 0.999$
Adam	150	0.0003	
Adam	300	0.0003	
AMSGrad	75	0.0003	$\beta_1 = 0.9, \beta_2 = 0.999$
AMSGrad	150	0.0003	
AMSGrad	300	0.0003	
QHAdam	75	0.0003	$\nu_1 = 0.7, \nu_2 = 1.0, \beta_1 = 0.9, \beta_2 = 0.99$
QHAdam	150	0.0003	
QHAdam	300	0.0003	
AdamW	75	0.0003	$\beta_1 = 0.9, \beta_2 = 0.999, wd = 0.01$
AdamW	150	0.0003	$\beta_1 = 0.9, \beta_2 = 0.999, wd = 0.001$
AdamW	300	0.0003	$\beta_1 = 0.9, \beta_2 = 0.999, wd = 0.001$
YellowFin	75	0.1	$\beta_1 = 0$
YellowFin	150	0.1	
YellowFin	300	0.1	
DEMON Adam	75	0.00003	$\beta_{\text{init}} = 0.9, \beta_2 = 0.999$
DEMON Adam	150	0.00003	
DEMON Adam	300	0.00003	
AggMo	75	0.03	$\beta = [0, 0.9, 0.99]$
AggMo	150	0.01	
AggMo	300	0.01	
QHM	75	1.0	$\nu = 0.7, \beta = 0.999$
QHM	150	1.0	
QHM	300	0.3	
DEMON SGDM	75		$\beta_{\text{init}} = 0.95$
DEMON SGDM	150	0.01	
DEMON SGDM	300		
SGDM	75	0.1	$\beta_1 = 0.9$
SGDM	150	0.03	$\beta_1 = 0.9$
SGDM	300	0.03	$\beta_1 = 0.9$

Table 9: Best parameters for STL10 with Wide ResNet 16-8.

Optimization method	epochs	η	other parameters
Adam	50	0.001	$\beta_1 = 0.9$, $\beta_2 = 0.999$
Adam	100	0.0003	
Adam	200	0.0003	
AMSGrad	50	0.0003	$\beta_1 = 0.9$, $\beta_2 = 0.999$
AMSGrad	100	0.0003	
AMSGrad	200	0.0003	
QHAdam	50	0.0003	$\nu_1 = 0.7$, $\nu_2 = 1.0$, $\beta_1 = 0.9$, $\beta_2 = 0.99$
QHAdam	100	0.0003	
QHAdam	200	0.0003	
AdamW	50	0.0003	$\beta_1 = 0.9$, $\beta_2 = 0.999$, $wd = 0.001$
AdamW	100	0.0003	
AdamW	200	0.0003	
YellowFin	50	0.1	$\beta_1 = 0$
YellowFin	100	0.1	
YellowFin	200	0.1	
DEMON Adam	50	0.00003	$\beta_{\text{init}} = 0.9$, $\beta_2 = 0.999$
DEMON Adam	100	0.00003	
DEMON Adam	200	0.00003	
AggMo	50	0.1	$\beta = [0, 0.9, 0.99]$
AggMo	100	0.1	
AggMo	200	0.1	
QHM	50	1.0	$\nu = 0.7$, $\beta = 0.999$
QHM	100	3.0	
QHM	200	3.0	
DEMON SGDM	50		$\beta_{\text{init}} = 0.95$
DEMON SGDM	100	0.03	
DEMON SGDM	200		
SGDM	50	0.1	$\beta_1 = 0.9$
SGDM	100	0.1	$\beta_1 = 0.9$
SGDM	200	0.1	$\beta_1 = 0.9$

Table 10: Best parameters for Tiny ImageNet with ResNet-56.

Optimization method	epochs	η	other parameters
Adam	20	0.001	$\beta_1 = 0.9$, $\beta_2 = 0.999$
Adam	40	0.0003	
DEMON Adam	20	0.0001	$\beta_{\text{init}} = 0.95$, $\beta_2 = 0.999$
DEMON Adam	40	0.0001	
DEMON SGDM	20	0.01	$\beta_{\text{init}} = 0.95$
DEMON SGDM	40	0.01	
SGDM	20	0.1	$\beta_1 = 0.9$
SGDM	40	0.1	

Table 11: Best parameters for PTB with LSTM architecture.

Optimization method	epochs	η	other parameters
Adam	25	0.0003	$\beta_1 = 0.9$, $\beta_2 = 0.999$
Adam	39	0.0003	
AMSGrad	25	0.001	$\beta_1 = 0.9$, $\beta_2 = 0.999$
AMSGrad	39	0.001	
QHAdam	25	0.0003	$\nu_1 = 0.7$, $\nu_2 = 1.0$, $\beta_1 = 0.9$, $\beta_2 = 0.999$
QHAdam	39	0.0003	
AdamW	25	0.001	$\beta_1 = 0.9$, $\beta_2 = 0.999$, $wd = 0.00005$
AdamW	39	0.001	
YellowFin	25	0.1	$\beta_1 = 0$
YellowFin	39	0.1	
DEMON Adam	25	0.0001	$\beta_{\text{init}} = 0.9$, $\beta_2 = 0.999$
DEMON Adam	39	0.0001	
AggMo	25	0.03	$\beta = [0, 0.9, 0.99]$
AggMo	39	0.03	
QHM	25	1.0	$\nu = 0.7$, $\beta = 0.999$
QHM	39	1.0	
DEMON SGDM	25	1.0	$\beta_{\text{init}} = 0.5$, $\beta_{final} = -0.5$
DEMON SGDM	39	1.0	$\beta_{\text{init}} = 0.3$, $\beta_{final} = -0.5$
SGDM	25	0.1	$\beta_1 = 0.9$, smooth learning rate decay
SGDM	39	1.0	$\beta_1 = 0.0$, smooth learning rate decay

C Convergence Analysis

We analyze the global convergence of DEMON SGDM in the convex setting, following (Ghadimi et al., 2014). For an objective function f which is convex, continuously differentiable, its gradient $\nabla f(\cdot)$ is Lipschitz continuous with constant L, our goal is to show that $f(\bar{\theta}_T)$ converges to the optimum f^* with decreasing momentum, where $\bar{\theta}_T$ is the average of θ_t for $t = 1, ..., T$. Our following theorem holds for a constant learning rate and β_t decaying with t.

Theorem 1. *Assume that f is convex, continuously differentiable, its gradient $\nabla f(\cdot)$ is Lipschitz continuous with constant L, with a decreasing momentum, but constant step size, as in:*

$$\beta_t = \frac{1}{t} \cdot \frac{t+1}{t+2}, \quad \alpha \in \left(0, \frac{2}{3L}\right).$$

We consider the SGDM iteration in non-stochastic settings, where:

$$\theta_{t+1} = \theta_t - \alpha \nabla f(\theta_t) + \beta_t \left(\theta_t - \theta_{t-1}\right).$$

Then, the sequence $\{\theta_t\}_{t=1}^T$ generated by the SGDM iteration, with decreasing momentum, satisfies:

$$f(\bar{\theta}_T) - f^* \leq \frac{\|\theta_1 - \theta^\star\|^2}{T} \left(\frac{3}{4}L + \frac{1}{2\alpha}\right),$$

where $\bar{\theta}_T$ is the Cesaro average of the iterates: $\bar{\theta}_T = \frac{1}{T}\sum_{t=1}^T \theta_t$.

Proof. Let $\beta_t = \frac{1}{t} \cdot \frac{t+1}{t+2}$ and

$$p_t = \tfrac{1}{t}(\theta_t - \theta_{t-1}).$$

We consider the SGDM iteration in non-stochastic settings, where:

$$\theta_{t+1} = \theta_t - \alpha \nabla f(\theta_t) + \beta_t \left(\theta_t - \theta_{t-1}\right).$$

Using the definition of p_t above, one can easily prove that:

$$\theta_{t+1} + p_{t+1} = (1 + \frac{1}{t+1})\theta_{t+1} - \frac{1}{t+1}\theta_t = \theta_t + p_t - \tfrac{\alpha(t+2)}{t+1}\nabla f(\theta_t).$$

Table 12: Best parameters for MNIST with VAE.

Optimization method	epochs	η	other parameters
Adam	50	0.001	$\beta_1 = 0.9, \beta_2 = 0.999$
Adam	100	0.001	
Adam	200	0.001	
AMSGrad	50	0.001	$\beta_1 = 0.9, \beta_2 = 0.999$
AMSGrad	100	0.001	
AMSGrad	200	0.001	
QHAdam	50	0.001	$\nu_1 = 0.7, \nu_2 = 1.0, \beta_1 = 0.9, \beta_2 = 0.99$
QHAdam	100	0.001	
QHAdam	200	0.001	
AdamW	50	0.001	$\beta_1 = 0.9, \beta_2 = 0.999, wd = 0.0001$
AdamW	100	0.001	
AdamW	200	0.001	
YellowFin	50	0.0001	$\beta_1 = 0$
YellowFin	100	0.0001	
YellowFin	200	0.0001	
DEMON Adam	50	0.0001	$\beta_{\text{init}} = 0.9, \beta_2 = 0.999$
DEMON Adam	100	0.0001	
DEMON Adam	200	0.0001	
AggMo	50	0.000003	$\beta = [0, 0.9, 0.99]$
AggMo	100	0.000003	
AggMo	200	0.000003	
QHM	50	0.0001	$\nu = 0.8, \beta = 0.999$
QHM	100	0.0001	
QHM	200	0.0001	
DEMON SGDM	50	0.0003	$\beta_{\text{init}} = 0.95$
DEMON SGDM	100		
DEMON SGDM	200		
SGDM	50	0.00001	$\beta_1 = 0.9$
SGDM	100	0.00001	$\beta_1 = 0.9$
SGDM	200	0.00001	$\beta_1 = 0.9$

Table 13: Best parameters for FMNIST with Capsule Network.

Optimization method	epochs	η	other parameters
Adam	50	0.001	$\beta_1 = 0.9, \beta_2 = 0.999$
Adam	100	0.001	
AMSGrad	50	0.001	$\beta_1 = 0.9, \beta_2 = 0.999$
AMSGrad	100	0.001	
QHAdam	50	0.0003	$\nu_1 = 0.7, \nu_2 = 1.0, \beta_1 = 0.9, \beta_2 = 0.999$
QHAdam	100	0.0003	
AdamW	50	0.001	$\beta_1 = 0.9, \beta_2 = 0.999, wd = 0.0001$
AdamW	100	0.001	
YellowFin	50	0.001	$\beta_1 = 0$
YellowFin	100	0.001	
DEMON Adam	50	0.001	$\beta_{\text{init}} = 0.9, \beta_2 = 0.999$
DEMON Adam	100	0.001	

Using this expression, we will analyze the term $\|\theta_{t+1} + p_{t+1} - \theta^\star\|_2$:

$$\begin{aligned}\|\theta_{t+1} + p_{t+1} - \theta^\star\|^2 &= \|\theta_t + p_t - \theta^\star\|^2 - \tfrac{2\alpha(t+2)}{t+1}\langle \theta_t + p_t - \theta^\star, \nabla f(\theta_t)\rangle + \left(\tfrac{\alpha(t+2)}{t+1}\right)^2 \cdot \|\nabla f(\theta_t)\|^2 \\ &= \|\theta_t + p_t - \theta^\star\|^2 - \tfrac{2\alpha(t+2)}{t(t+1)}\langle \theta_t - \theta_{t-1}, \nabla f(\theta_t)\rangle \\ &\quad - \tfrac{2\alpha(t+2)}{t+1}\langle \theta_t - \theta^\star, \nabla f(\theta_t)\rangle + \left(\tfrac{\alpha(t+2)}{t+1}\right)^2 \cdot \|\nabla f(\theta_t)\|^2\end{aligned}$$

Since f is convex, continuously differentiable, its gradient is Lipschitz continuous with constant L, then

$$\tfrac{1}{L}\|\nabla f(\theta_t)\|^2 \le \langle \theta_t - \theta^\star, \nabla f(\theta_t)\rangle, \tag{2}$$

$$f(\theta_t) - f^* + \tfrac{1}{2L}\|\nabla f(\theta_t)\|^2 \le \langle \theta_t - \theta^\star, \nabla f(\theta_t)\rangle, \tag{3}$$

$$f(\theta_t) - f(\theta_{t-1}) \le \langle \theta_t - \theta_{t-1}, \nabla f(\theta_t)\rangle. \tag{4}$$

Substituting the above inequalities leads to

$$\begin{aligned}\|\theta_{t+1} + p_{t+1} - \theta^\star\|^2 &\le \|\theta_t + p_t - \theta^\star\|^2 - \tfrac{2\alpha(t+2)}{t(t+1)}(f(\theta_t) - f(\theta_{t-1})) \\ &\quad - 2\alpha\tfrac{(1-\lambda)(t+2)}{L(t+1)} \cdot \|\nabla f(\theta_t)\|^2 - 2\alpha\lambda\tfrac{t+2}{t+1}(f(\theta_t) - f^*) \\ &\quad - \left(\alpha\tfrac{\lambda(t+2)}{L(t+1)}\right) \cdot \|\nabla f(\theta_t)\|^2 + \left(\frac{\alpha(t+2)}{t+1}\right)^2 \cdot \|\nabla f(\theta_t)\|^2\end{aligned}$$

where $\lambda \in (0, 1]$ is a parameter weighting (2) and (3). Grouping together terms yields

$$\begin{aligned}&\left(\tfrac{2\alpha(t+2)}{t(t+1)} + \tfrac{2\alpha\lambda(t+2)}{t+1}\right)(f(\theta_t) - f^*) + \|\theta_{t+1} + p_{t+1} - \theta^\star\|^2 \\ &\qquad \le \tfrac{2\alpha(t+2)}{t(t+1)}(f(\theta_{t-1}) - f^*) + \|\theta_t + p_t - \theta^\star\|^2 \\ &\qquad\quad + \tfrac{\alpha(t+2)}{t+1}\left(\tfrac{\alpha(t+2)}{t+1} - \tfrac{2(1-\lambda)}{L} - \tfrac{\lambda}{L}\right)\|\nabla f(\theta_t)\|^2.\end{aligned}$$

The last term is non-positive when $\alpha \in [0, \tfrac{t+1}{t+2}(\tfrac{2-\lambda}{L})]$ so it can be dropped. Summing over $t = 1, ..., T$ yields

$$\begin{aligned}&2\alpha\lambda\sum_{t=1}^{T}\tfrac{t+2}{t+1}(f(\theta_t) - f^*) + \sum_{t=1}^{T}\left(\tfrac{2\alpha(t+2)}{t(t+1)}(f(\theta_t) - f^*) + \|\theta_{t+1} + p_{t+1} - \theta^\star\|^2\right) \\ &\qquad \le \sum_{t=1}^{T}\left(\tfrac{2\alpha(t+2)}{t(t+1)}(f(\theta_{t-1}) - f^*) + \|\theta_t + p_t - \theta^\star\|^2\right),\end{aligned}$$

implying that:

$$2\alpha\lambda\sum_{t=1}^{T}\tfrac{t+2}{t+1}(f(\theta_t) - f^*) \le 3\alpha(f(\theta_1) - f^*) + \|\theta_1 - \theta^\star\|^2.$$

Since:

$$2\alpha\lambda\sum_{t=1}^{T}(f(\theta_t) - f^*) \le 2\alpha\lambda\sum_{t=1}^{T}\tfrac{t+2}{t+1}(f(\theta_t) - f^*) \le 3\alpha\lambda\sum_{t=1}^{T}(f(\theta_t) - f^*),$$

we further have:

$$3\alpha\lambda\sum_{t=1}^{T}(f(\theta_t) - f^*) \le \tfrac{3}{2}\left(3\alpha(f(\theta_1) - f^*) + \|\theta_1 - \theta^\star\|^2\right).$$

Due to the convexity of f,

$$f(\bar{\theta}_t) \le \tfrac{1}{T}\sum_{t=1}^{T} f(\theta_t),$$

observe that

$$f(\bar{\theta}_T) - f^* \leq \frac{1}{T}\sum_{t=1}^{T}(f(\theta_t) - f^*) \leq \frac{1}{3\alpha\lambda T}\left(\frac{9}{2}\alpha(f(\theta_1) - f^*) + \frac{3}{2}\|\theta_1 - \theta^\star\|^2\right).$$

Since $f(\theta_1) - f^* \leq \frac{L}{2}\|\theta_1 - \theta^\star\|^2$ by Lipschitz continuous gradients, setting $\lambda = 1$ and observing $(t+1)/(t+2) \geq 2/3$ gives the result.

For DEMON Adam, we observe it lies within the definition of Generic Adam in Zou et al. (2018), and inherits the non-convex results. This can be obtained through a re-parameterization and since there is no change in the proof and the result is direct, we give credit to Zou et al. (2018) and leave this as an exercise to the reader.

D LINEAR REGRESSION

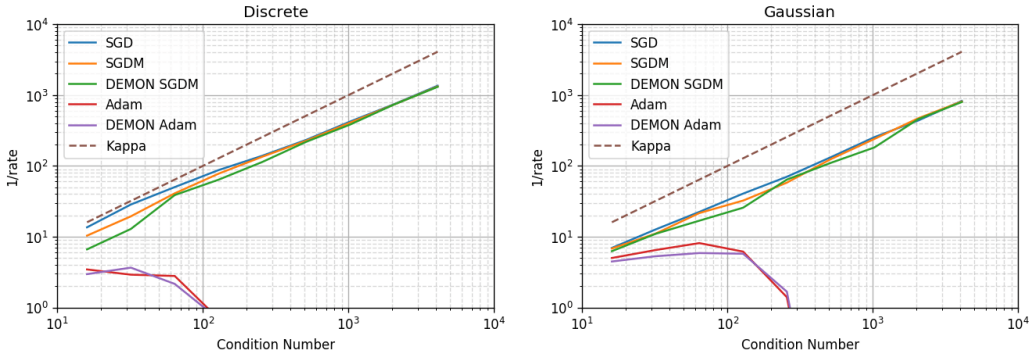

Figure 4: Linear regression with 1/rate vs κ (Condition Number). Left: Discrete. Right: Gaussian.

We replicate the linear regression setting in (Kidambi et al., 2018) and summarize the key details here. We consider two different classes of linear regression problems in two dimensions, where κ is the condition number and samples are (a, b), namely:

Discrete: $a = e_1$ with probability 0.5, and $a = \frac{2}{\kappa}e_2$ w.p. 0.5; e_i is the i-th standard basis vector.

Gaussian: $a \in \mathbb{R}^2$ distributed as a Gaussian random vector with covariance matrix $\begin{pmatrix} 1 & 0 \\ 0 & \frac{1}{\kappa} \end{pmatrix}$.

We evaluate SGD, SGDM, DEMON SGDM, Adam, and DEMON Adam, tuning with a grid search. We fix a randomly generated $\theta^\star$, and let $b = \langle \theta^\star, a \rangle$. κ is varied from 2^4 to 2^{12} in powers of 2 and for each setting we run 100 independent trials for $t = 5\kappa$ iterations, considering only those that converged. Following (Kidambi et al., 2018), the algorithm is considered to converge is no error in the second half of iterations exceeds starting error. Performance is measured using rate $= \frac{\log(f(\theta_1)) - \log(f(\theta_t))}{t}$ and we compute the rate for different κ. Results are given in Figure 4: What is apparent is that the convergence rate is preserved when we decrease the momentum parameter, despite the fact that theory dictates the opposite in convex scenarios.

Table 15: `VGG16-CIFAR100-DEMONSGDM` and `WRN-STL10-DEMONSGDM` generalization error. The number of epochs was predefined before the execution.

	VGG-16			**Wide Residual 16-8**		
	75 epochs	150 epochs	300 epochs	50 epochs	100 epochs	200 epochs
SGD ELR	36.82 ± .68	30.34 ± .30	29.81 ± .31	20.90 ± .47	17.53 ± .32	15.37 ± .51
DEMON SGDM	**33.08** ± .49	**30.22** ± .50	**27.71** ± .05	**19.45** ± .20	**15.98** ± .40	**13.67** ± .13

Table 16: `PTB-LSTM-DEMONSGDM` (perplexity) and `VAE-MNIST-DEMONSGDM` (generalization loss) experiments.

	LSTM		**VAE**		
	25 epochs	39 epochs	50 epochs	100 epochs	200 epochs
SGD ELR	inf	inf	inf	inf	inf
DEMON SGDM	**88.33** ± .16	**88.32** ± .12	**139.32** ± .23	**137.51** ± .29	**135.95** ± .21

E DEMON AND EFFECTIVE LEARNING RATE

In this section, we present results of Demon against the effective learning rate adjusted SGD (SGD ELR). The effective learning rate is proposed to approximate SGDM with SGD, where the learning rate is adjusted with a factor of $1/(1-m)$ and m is the momentum coefficient. However, the results in Tables 15, 16, and 17 demonstrate that DEMON cannot be accurately approximated with an effective learning rate adjusted SGD. For both settings in Table 16 (`PTB-LSTM-DEMONSGDM` and `VAE-MNIST-DEMONSGDM`), SGD ELR causes learning to diverge. In Table 15, there exists a 1-3% generalization error gap for `VGG16-CIFAR100-DEMONSGDM` and `WRN-STL10-DEMONSGDM`. In Table 17, there exists a 1% generalization gap for `RN20-CIFAR10-DEMONSGDM`.

F ADDITIONAL PLOTS

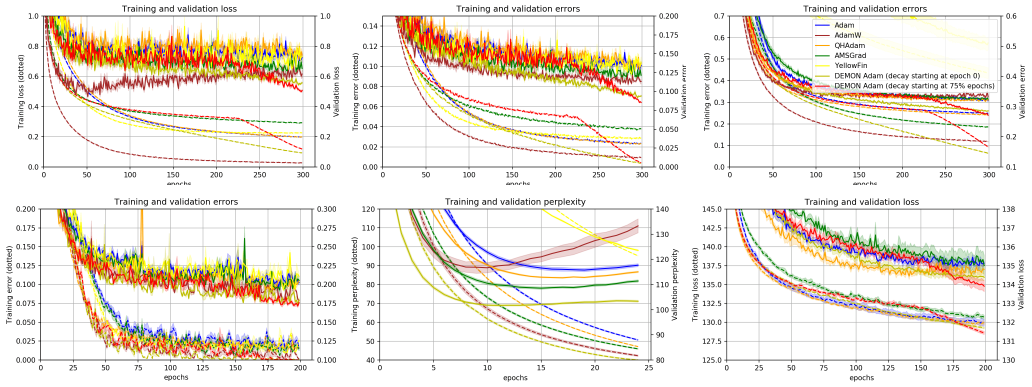

Figure 5: Top row, two left-most plots: `RN20-CIFAR10-DEMONAdam` for 300 epochs. Top row, right-most plot: `VGG16-CIFAR100-DEMONAdam` for 300 epochs. Bottom row, left-most plot: `WRN-STL10-DEMONAdam` for 200 epochs. Bottom row, middle plot: `PTB-LSTM-DEMONAdam` for 25 epochs. Bottom row, right-most plot: `VAE-MNIST-DEMONAdam` for 200 epochs. Dotted and solid lines represent training and generalization metrics respectively. Shaded bands represent one standard deviation.

Table 17: `RN20-CIFAR10-DEMONSGDM` generalization error. The number of epochs was predefined before the execution of the algorithms.

	30 epochs	75 epochs	150 epochs	300 epochs
SGD ELR	11.82 ± .13	9.46 ± .25	8.72 ± .06	8.46 ± .19
DEMON SGDM	**10.39** ± .39	8.74 ± .28	**7.82** ± .27	**7.58** ± .04

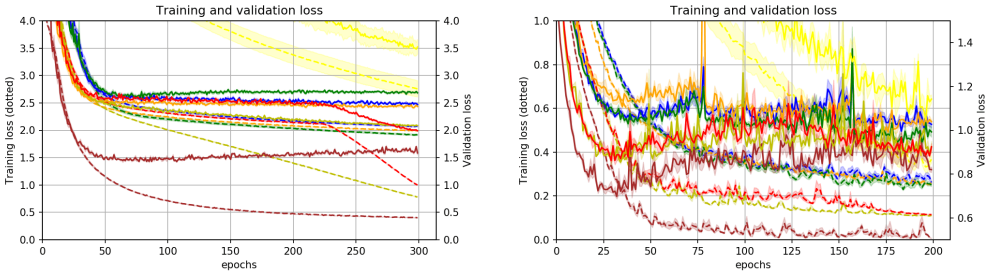

Figure 6: Additional empirical results on adaptive learning rate methods. Left plot: `VGG16-CIFAR100-DEMONAdam` for 300 epochs. Right plot: `WRN-STL10-DEMONAdam` for 200 epochs. Dotted and solid lines represent training and generalization metrics respectively. Shaded bands represent 1 standard deviation.

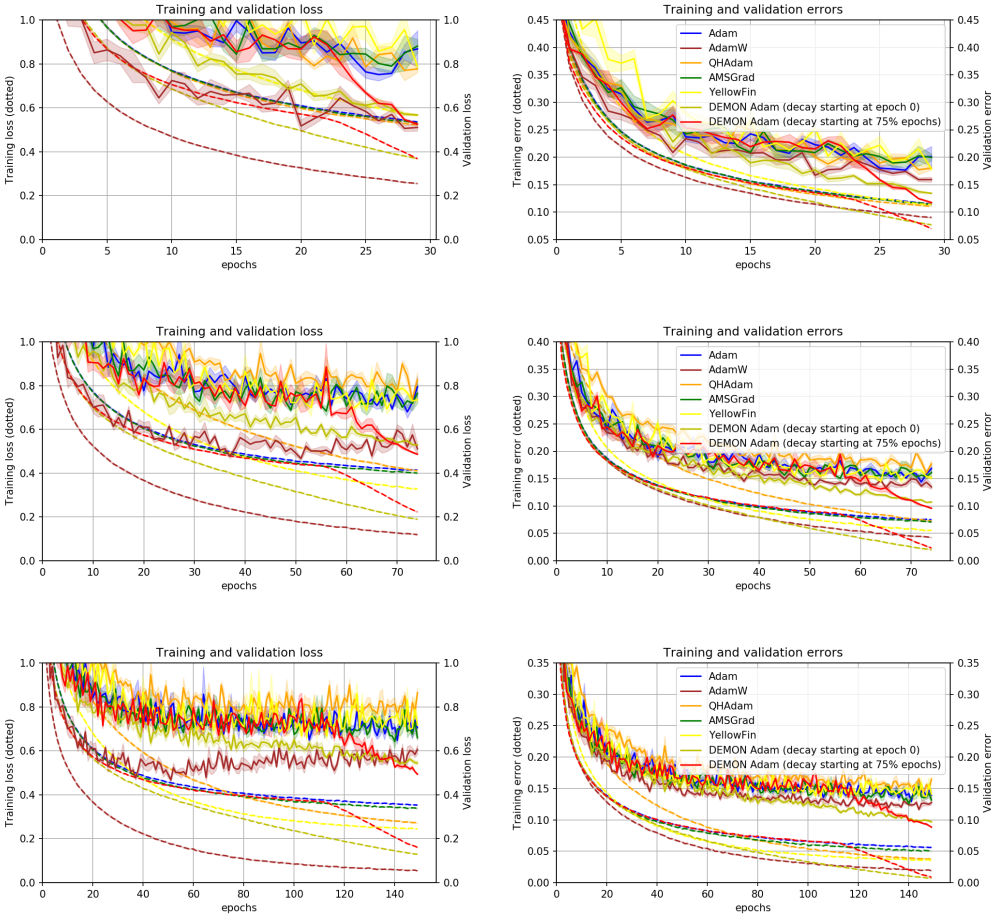

Figure 7: Additional empirical results on `RN20-CIFAR10-DEMONAdam`. Top row: 30 epochs. Middle row: 75 epochs. Bottom row: 150 epochs. Dotted and solid lines represent training and generalization metrics respectively. Shaded bands represent 1 standard deviation.

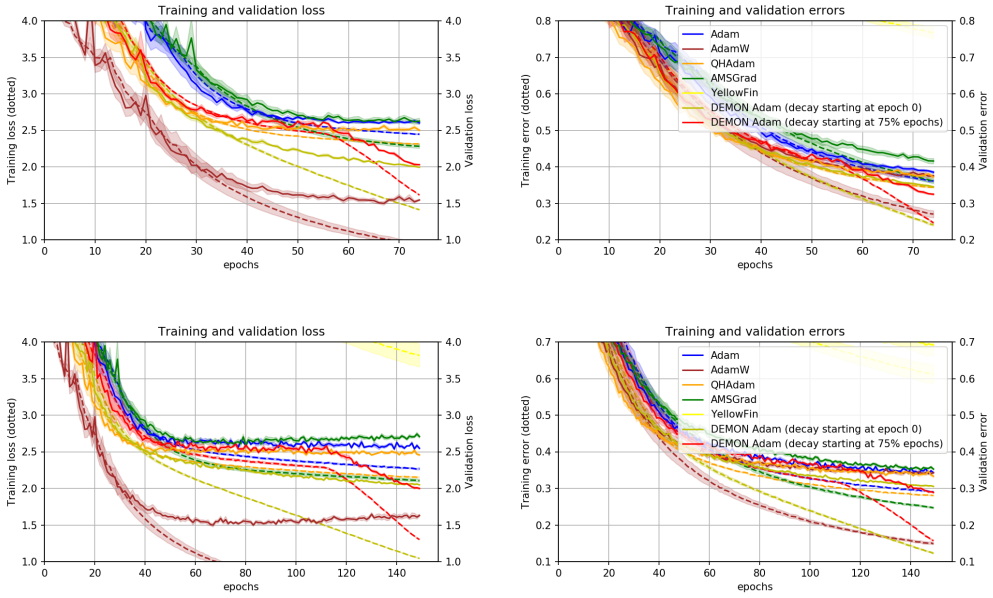

Figure 8: Additional empirical results on `VGG16-CIFAR100-DEMONAdam`. Top row: 75 epochs. Bottom row: 150 epochs. Dotted and solid lines represent training and generalization metrics respectively. Shaded bands represent 1 standard deviation.

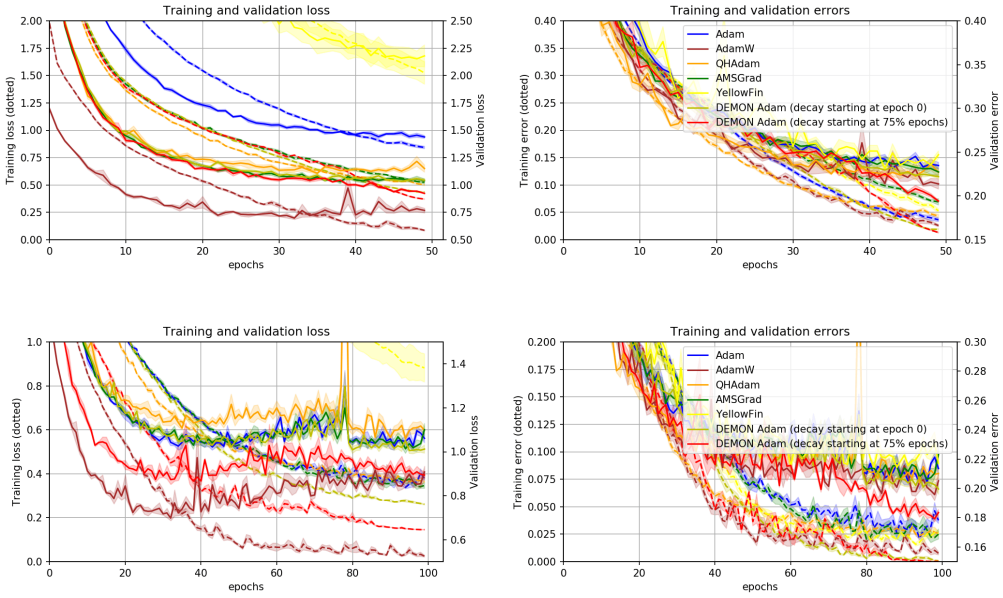

Figure 9: Additional empirical results on `WRN-STL10-DEMONAdam`. Top row: 50 epochs. Bottom row: 100 epochs. Dotted and solid lines represent training and generalization metrics respectively. Shaded bands represent 1 standard deviation.

G Different Momentum Schedules

In this section, we present empirical results for multiple possible momentum schedule variants in comparison to DEMON. In particular, we experiment with a linear decay schedule, a cyclic momen-

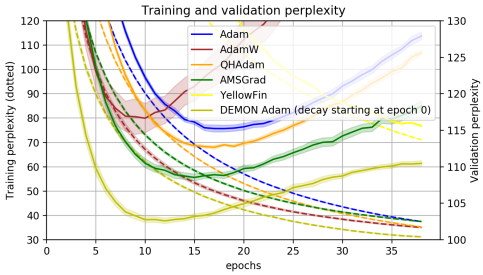

Figure 10: Additional empirical results on `PTB-LSTM-DEMONAdam` for 39 epochs. Dotted and solid lines represent training and generalization metrics respectively. Shaded bands represent 1 standard deviation.

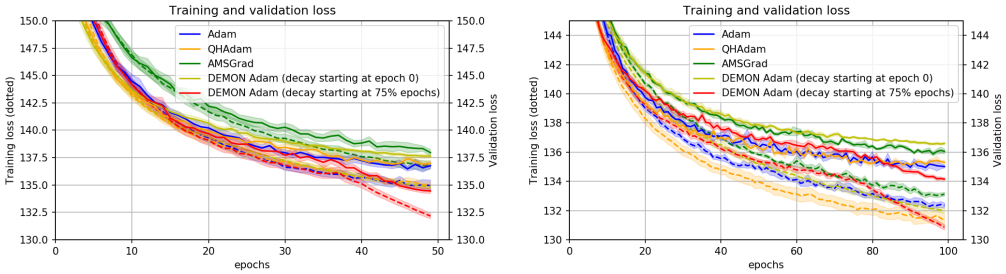

Figure 11: Additional empirical results on `VAE-MNIST-DEMONAdam`. Left: 50 epochs. Right: 100 epochs. Dotted and solid lines represent training and generalization metrics respectively. Shaded bands represent 1 standard deviation.

tum schedule, a DEMON schedule with restarts, and multiple step decay schedules. All experiments were performed on the CIFAR10 dataset using the ResNet-20 architecture. Training was performed for 75 epochs and no learning rate decay was used. All tests were performed with identical hyperparameters, aside from the differing momentum schedules. All of the exact momentum schedules that were tested are as follows:

- LINEAR: a linear decay schedule was performed from a momentum of 0.9 to 0.0. The application of momentum decay was delayed until 75% of epochs, which yielded improved performance.
- CYCLE-1: a linear, cyclic momentum schedule was utilized. For a single cycle, momentum will increase linearly from 0.0 to 0.9, followed by a linear decay back to 0.0. In this test, only a single cycle was performed over the 75 epochs.
- CYCLE-2: the same linear, cyclic momentum schedule was utilized. In this test, two cycles were performed during training instead of one.
- RESTART-2: the DEMON schedule was used with restarts. Namely, the entire Demon schedule was applied until 50% of training was complete, at which point the momentum schedule was reset. Two complete DEMON schedules were completed during training.
- RESTART-4: DEMON with restarts was used once again. This test performed four restarts during training.
- STEP-50-75: a step momentum schedule was utilized. Momentum began at 0.9, was decayed to 0.7 at 50% of total epochs, and decayed to 0.3 at 75% of total epochs.
- STEP-75-90: a step momentum schedule was utilized. Momentum began at 0.9, was decayed to 0.4 at 75% of total epochs, and was decayed to 0.1 at 90% of total epochs.
- STEP-85-95: a step momentum schedule was utilized. Momentum began at 0.9, was decayed to 0.3 at 85% of total epochs, and was decayed to 0.1 at 95% of total epochs.

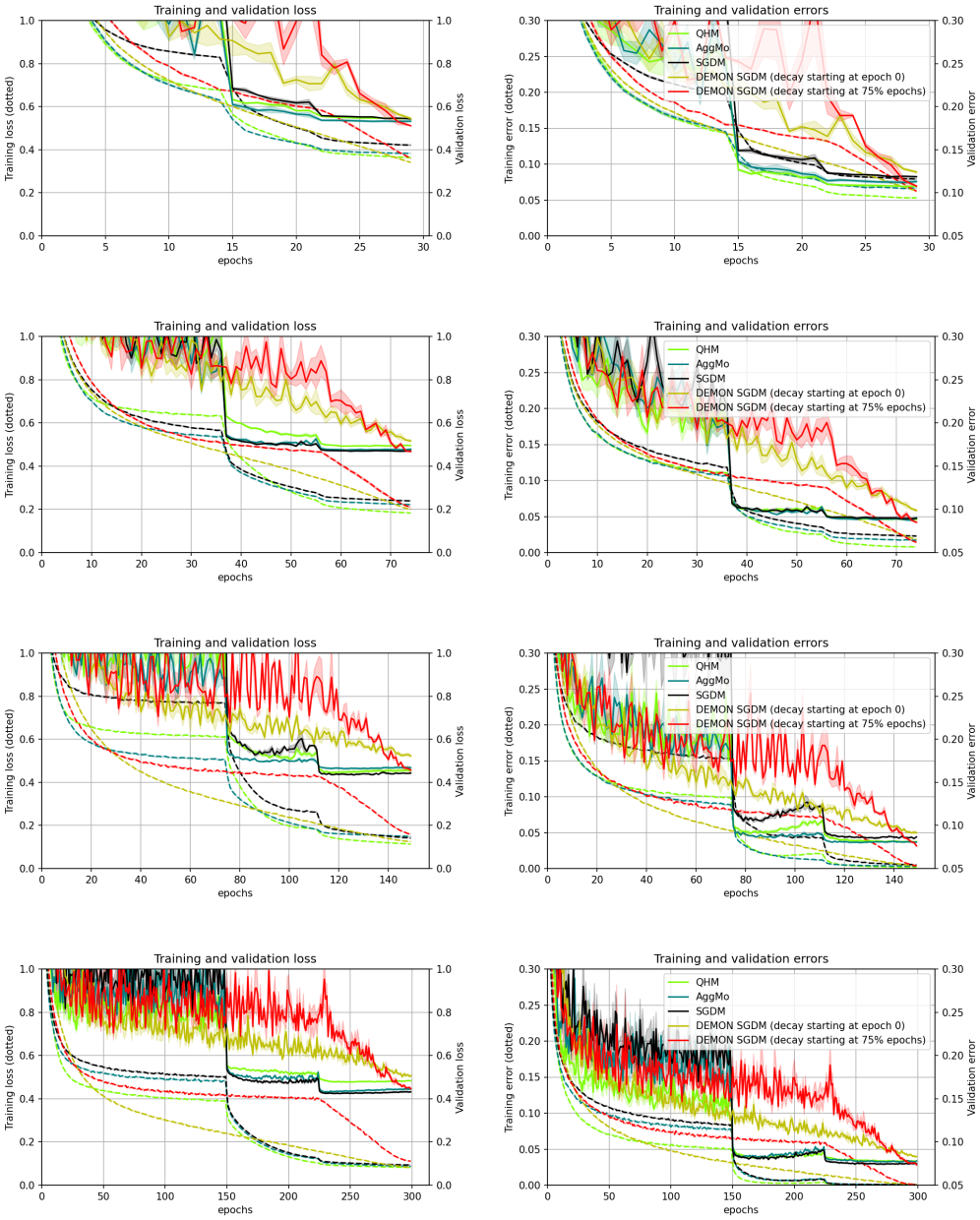

Figure 12: Additional empirical results on `RN20-CIFAR10-DEMONSGDM`. Top row: 30 epochs. Middle upper row: 75 epochs. Middle lower row: 150 epochs. Bottom row: 300 epochs. Dotted and solid lines represent training and generalization metrics respectively. Shaded bands represent 1 standard deviation.

- DEMON: the normal DEMON momentum schedule was used. Momentum decay was not applied until 75% of training was complete, which mirrors the experimental settings presented in the main text.

The validation performance of all models trained with each of these different momentum schedules is presented in Table 18. No other momentum schedule outperforms DEMON.

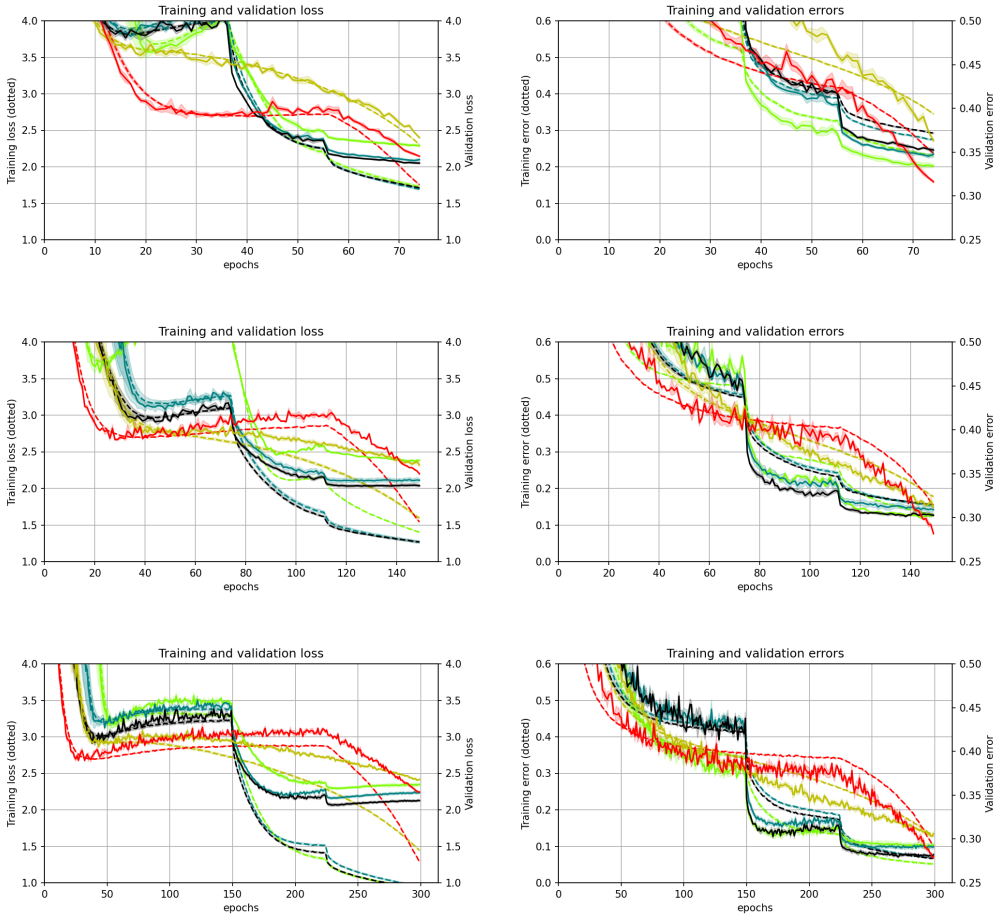

Figure 13: Additional empirical results on `VGG16-CIFAR100-DEMONSGDM`. Top row: 75 epochs. Middle row: 150 epochs. Bottom row: 300 epochs. Dotted and solid lines represent training and generalization metrics respectively. Shaded bands represent 1 standard deviation.

Table 18: Validation error on CIFAR10 for models training with different momentum schedule variants

Experiment	Validation Error
Linear	9.85 ± 0.14
Cycle-1	13.72 ± 0.18
Cycle-2	13.74 ± 0.45
Restart-2	10.90 ± 0.21
Restart-4	10.83 ± 0.11
Step-50-75	10.50 ± 0.05
Step-75-90	9.78 ± 0.18
Step-85-95	9.58 ± 0.10
Demon	9.61 ± 0.09

H PADAM AND ONECYCLE PRELIMINARY RESULTS

In this section, we present preliminary results on Padam Chen & Gu (2018) and OneCycle Smith (2018) on several tasks. We conducted preliminary studies of Padam for the settings of ResNet20 on CIFAR10 with 300 epochs, VGG16 on CIFAR100 with 150 epochs, and Variational AutoEncoder on MNIST with 50 epochs.

For `RN20-CIFAR10`, following the Padam paper, we try learning rate in $[0.1,\ 0.03,\ 0.01,\ 0.003,\ 0.001,\ 0.0003]$, $p \in [1/4,\ 1/8,\ 1/16]$, $\beta_1 = 0.9$, $\beta_2 = 0.999$.

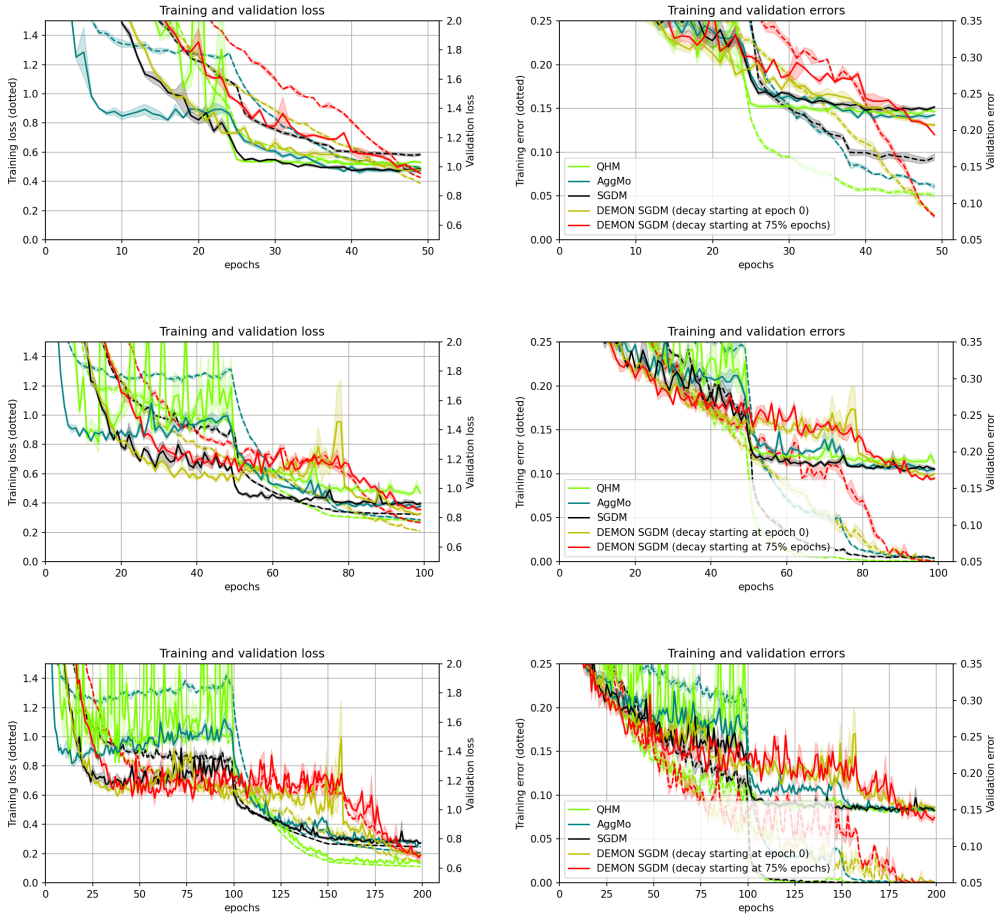

Figure 14: Additional empirical results on `WRN-STL10-DEMONSGDM`. Top row: 50 epochs. Middle row: 100 epochs. Bottom row: 200 epochs. Dotted and solid lines represent training and generalization metrics respectively. Shaded bands represent 1 standard deviation.

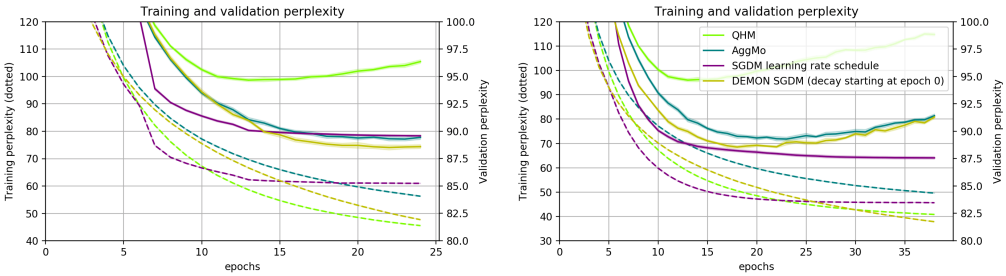

Figure 15: Additional empirical results on `PTB-LSTM-DEMONSGDM`. Left: 25 epochs. Right: 39 epochs. Dotted and solid lines represent training and generalization metrics respectively. Shaded bands represent 1 standard deviation.

The lowest test error is attained with learning rate 0.01 and $p = 1/4$, at $12.13 \pm .70$. Demon Adam achieves significantly lower test error at $8.44 \pm .05$.

For `VGG16-CIFAR100` and Padam, we try learning rate in $[0.1,\ 0.03,\ 0.01,\ 0.003,\ 0.001,\ 0.0003]$, $p \in [1/4, 1/8, 1/16]$, $\beta_1 = 0.9$, $\beta_2 = 0.999$. The

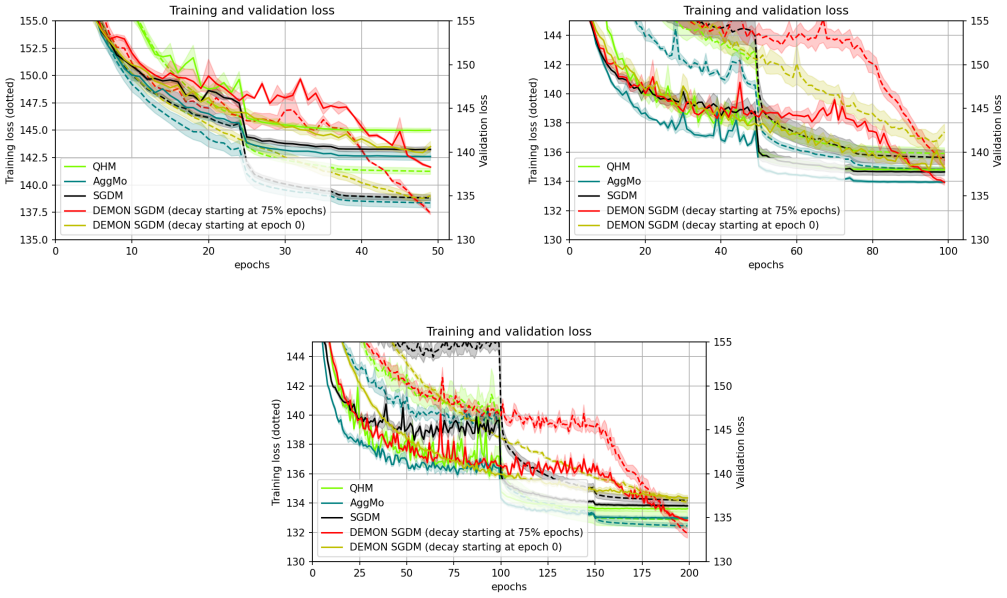

Figure 16: Additional empirical results on `VAE-MNIST-DEMONSGDM`. Left: 50 epochs. Right: 100 epochs. Bottom: 200 epochs. Dotted and solid lines represent training and generalization metrics respectively. Shaded bands represent 1 standard deviation.

lowest test error is attained with learning rate 0.03 and $p = 1/16$, at $34.38 \pm .71$. Demon Adam, again, achieves significantly lower test error at $28.84 \pm .18$.

For `VAE-MNIST` and Padam, we try learning rate in $[0.003,\ 0.001,\ 0.0003,\ 0.0001]$, exponent $p \in [0.4,\ 0.25,\ 0.125,\ 0.0625]$ and $\beta_1 = 0.9$, $\beta_2 = 0.999$. The lowest validation loss is attained with learning rate 0.001 and $p = 0.4$, with a loss value of $137.37 \pm .75$. For this task, Demon Adam achieves $134.46 \pm .17$, substantially better.

We also conducted preliminary studies of 1cycle with momentum SGD for the settings of ResNet20 on CIFAR10 for 300 epochs and VGG16 on CIFAR100 for 150 epochs.

Following the suggestions in the paper, for `RN20-CIFAR10` we try all combinations of maximum learning rate in $[3.0,\ 1.0,\ 0.3,\ 0.1,\ 0.03,\ 0.01]$, maximum momentum in $[0.97,\ 0.95,\ 0.9]$, minimum momentum in $[0.85,\ 0.8]$, batch size in $[128,\ 256,\ 512]$, with minimum learning rate $= 0.1\cdot$ maximum learning rate. The lowest test error is achieved with maximum learning rate 1.0, maximum momentum 0.95, minimum momentum 0.85, batch size 512, achieving $7.65 \pm .13$. Demon SGDM, with no tuning, achieves $7.58 \pm .04$.

For `VGG16-CIFAR100`, we try all combinations of maximum learning rate in $[1.0,\ 0.3,\ 0.1,\ 0.03]$, maximum momentum in $[0.97,\ 0.95,\ 0.9]$, minimum momentum in $[0.85,\ 0.8]$, batch size in $[128,\ 256,\ 512]$, with minimum learning rate $= 0.1\cdot$ maximum learning rate. The lowest test error is achieved with maximum learning rate 0.1, maximum momentum 0.95, minimum momentum 0.85, batch size 512, achieving 32.05 ± 1.05. In comparison, Demon SGDM, with no tuning, achieves significantly lower at $30.22 \pm .50$.

I Additional results for adaptive momentum methods without learning rate decay

We present additional results for Aggregated Momentum (AggMo) (Lucas et al., 2018), and Quasi-Hyperbolic Momentum (QHM) (Ma & Yarats, 2018) without learning rate decay. Since SGDM with learning rate decay is most often used to achieve the state-of-the-art results with the architectures and tasks in question, we include SGDM with learning rate decay as the target to beat. SGDM with

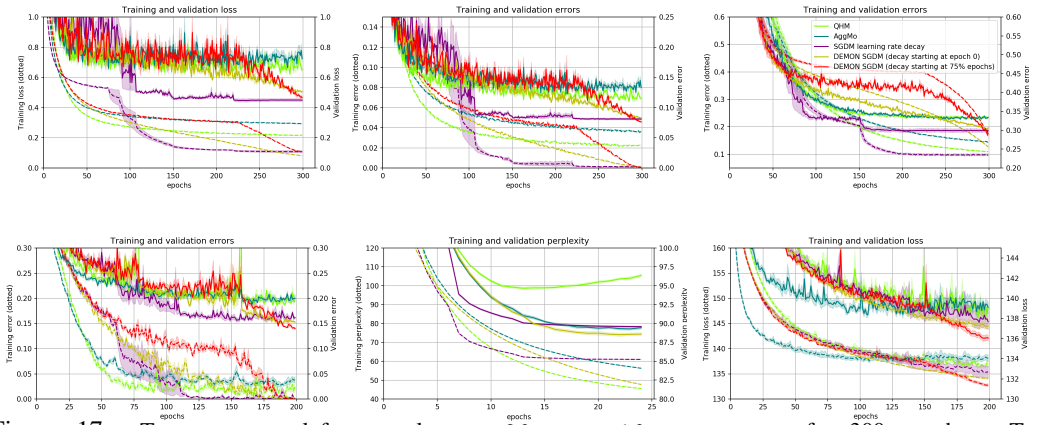

Figure 17: Top row, two left-most plot: `RN20-CIFAR10-DEMONSGDM` for 300 epochs. Top row, right-most plot: `VGG16-CIFAR100-DEMONSGDM` for 300 epochs. Bottom row, left-most plots: `WRN-STL10-DEMONSGDM` for 200 epochs. Bottom row, middle plot: `PTB-LSTM-DEMONSGDM` for 25 epochs. Bottom row, right-most plot: `VAE-MNIST-DEMONSGDM` for 200 epochs. Dotted and solid lines represent training and generalization metrics respectively. Shaded bands represent 1 standard deviation.

learning rate decay is implemented with a decay on validation error plateau, where we hand-tune the number of epochs to define plateau. We tune all learning rates in roughly multiples of 3 and try to keep all other parameters close to those recommended in the original literature. For DEMON SGDM, we leave $\beta_{\text{init}} = 0.9$ for most experiments and generally decay from β_{init} to 0.

Residual Neural Network (`RN20-CIFAR10-DEMONSGDM`). We train a ResNet20 model on the CIFAR-10 dataset. With DEMON SGDM, we achieve better generalization error than SGDM with learning rate decay, the optimizer for producing state-of-the-art results with ResNet architecture. The better performance of decaying momentum relative to learning rate decay is surprising.

Running 5 seeds, DEMON SGDM outperforms all other adaptive momentum methods by a large 3%-8% validation error margin with a small and large number of epochs and is competitive or better than SGDM with learning rate decay. In Figure 17 (*Top row, two left-most plots*), DEMON SGDM is observed to continue learning after other adaptive momentum methods appear to begin to plateau.

Non-Residual Neural Network (`VGG16-CIFAR100-DEMONSGDM`). For the CIFAR-100 dataset, we train an adjusted VGG-16 model. In Figure 17 (*Top row, right-most plot*), we observe DEMON SGDM to learn slowly initially in loss and error, but similar to the previous setting it continues to learn after other methods begin to plateau, resulting in superior final generalization error.

Running 5 seeds, DEMON SGDM achieves an improvement of 1%-8% generalization error margin over all other methods. Refer to Table 20 for more details.

Wide Residual Neural Network (`WRN-STL10-DEMONSGDM`). We train a Wide Residual 16-8 model for the STL-10 dataset. In Figure 17 (*Bottom row, left-most plot*), training in both loss and error slows down quickly for other adaptive momentum methods with a large gap with SGDM learning rate decay. DEMON SGDM continues to improve and eventually catches up to SGDM learning rate decay.

Running 5 seeds, DEMON SGDM outperforms all other methods by a 1.5%-2% generalization error margin with a small and large number of epochs. Refer to Table 20 for more details.

LSTM (`PTB-LSTM-DEMONSGDM`). We train an RNN with LSTM architecture for the PTB language modeling task. Running 5 seeds, DEMON SGDM slightly outperforms other adaptive momentum methods in generalization perplexity, and is competitive with SGDM with learning rate decay. Refer to Figure 17 (*Bottom row, middle plot*) and Table 21 for more details.

Variational AutoEncoder (`VAE-MNIST-DEMONSGDM`). We train the generative model VAE on the MNIST dataset. Running 5 seeds, DEMON SGDM outperforms all other methods by a 2%-6% generalization error for a small and large number of epochs. Refer to Figure 17 (*Bottom row, right-most plot*) and Table 21 for more details.

	30 epochs	75 epochs	150 epochs	300 epochs
SGDM LR decay	11.29 ± .35	9.05 ± .07	8.26 ± .07	7.97 ± .14
AggMo	18.85 ± .27	13.02 ± .23	11.95 ± .15	10.94 ± .12
QHM	14.65 ± .24	12.66 ± .19	11.27 ± .13	10.42 ± .05
DEMON SGDM	**10.39** ± .39	**8.74** ± .28	**7.82** ± .27	**7.58** ± .04

Table 19: `RN20-CIFAR10-DEMONSGDM` generalization error with no learning rate decay. The number of epochs was predefined before the execution of the algorithms.

	VGG-16			**Wide Residual 16-8**		
	75 epochs	150 epochs	300 epochs	50 epochs	100 epochs	200 epochs
SGDM LR decay	35.29 ± .59	30.65 ± .31	29.74 ± .43	21.05 ± .27	17.83 ± 0.39	15.16 ± .36
AggMo	42.85 ± .89	34.25 ± .24	32.32 ± .18	22.70 ± .11	20.06 ± .31	17.90 ± .13
QHM	42.14 ± .79	33.87 ± .26	32.45 ± .13	22.86 ± .15	19.40 ± .23	17.79 ± .08
DEMON SGDM	**33.08** ± .49	**30.22** ± .50	**28.99** (27.71) ± .16 (.05)	**19.45** ± .20	**15.98** ± .40	**13.67** ± .13

Table 20: `VGG16-CIFAR100-DEMONSGDM` and `WRN-STL10-DEMONSGDM` generalization error with no learning rate decay. The number of epochs was predefined before the execution.

	LSTM		**VAE**		
	25 epochs	39 epochs	50 epochs	100 epochs	200 epochs
SGDM LR decay	89.59 ± .07	**87.57** ± .11	140.51 ± .73	139.54 ± .34	137.33 ± .49
AggMo	89.09 ± .16	89.07 ± .15	139.69 ± .17	139.07 ± .26	137.64 ± .20
QHM	94.47 ± .19	94.44 ± .13	145.84 ± .39	140.92 ± .19	137.64 ± .20
DEMON SGDM	**88.33** ± .16	88.32 ± .12	**139.32** ± .23	**137.51** ± .29	**135.95** ± .21

Table 21: `PTB-LSTM-DEMONSGDM` (perplexity) and `VAE-MNIST-DEMONSGDM` with no learning rate decay (generalization loss) experiments.