Langevin Algorithms for Very Deep Neural Networks with Application to Image Classification

Pierre BRAS

Sorbonne Université, Paris, France

Presented at the International Neural Network Society Deep Learning Innovations and Applications INNS DLIA workshop, part of the International Joint Conference on Neural Networks IJCNN 2023

Abstract

Training a very deep neural network is a challenging task, as the deeper a neural network is, the more non-linear it is. We compare the performances of various preconditioned Langevin algorithms with their non-Langevin counterparts for the training of neural networks of increasing depth. For shallow neural networks, Langevin algorithms do not lead to any improvement, however the deeper the network is and the greater are the gains provided by Langevin algorithms. Adding noise to the gradient descent allows to escape from local traps, which are more frequent for very deep neural networks. Following this heuristic we introduce a new Langevin algorithm called Layer Langevin, which consists in adding Langevin noise only to the weights associated to the deepest layers. We then prove the benefits of Langevin and Layer Langevin algorithms for the training of popular deep residual architectures for image classification.

Langevin Gradient Descent

Consider a training problem with parameter θ and data $\mathcal{D}$ and learning rate γ:

$$\begin{aligned}\text{(Stochastic) Gradient:}\quad & g_{n+1} = \nabla_\theta V(\theta_n; \mathcal{D}_{n+1})\\ \text{(Stochastic) Gradient Descent:}\quad & \theta_{n+1} = \theta_n - \gamma_{n+1} g_{n+1},\\ \text{Langevin Gradient Descent:}\quad & \theta_{n+1} = \theta_n - \gamma_{n+1} g_{n+1} + \sigma\sqrt{\gamma_{n+1}}\mathcal{N}(0, I_d).\end{aligned}$$

Langevin dynamics were first introduced in a Bayesian setting Welling and Teh (2011). Compared with SGD, the small white noise adds **learning regularization** and allows to **escape from traps** for the gradient descent (local minima, saddle points). It has been proven that adding noise is known to improve the learning in some cases Neelakantan et al. (2015); Anirudh Bhardwaj (2019); Gulcehre et al. (2016).

Preconditioned Langevin Gradient Descent Li et al. (2016)

For some preconditioner rule P_{n+1} depending on the previous updates of the gradient:

$$\begin{aligned}\text{Preconditioned Gradient Descent:}\quad & \theta_{n+1} = \theta_n - \gamma_{n+1} P_{n+1}\cdot g_{n+1},\\ \text{Preconditioned Langevin:}\quad & \theta_{n+1} = \theta_n - \gamma_{n+1} P_{n+1}\cdot g_{n+1} + \sigma\sqrt{\gamma_{n+1}}\mathcal{N}(0, \boldsymbol{P_{n+1}}),\end{aligned}$$

with per-dimension adaptive step size. (e.g. Adam, RMSprop, Adadelta). **In the literature**, Li et al. (2016); Ma et al. (2015); Patterson and Teh (2013); Simsekli et al. (2016) compares the benefits of noisy and/or preconditioned optimizers.

Training very deep Neural Networks

Very deep neural networks are crucial, in particular in image classification He et al. (2016). However they are much more difficult to train since they are more "non-linear" and the potential landscape presents **local traps**. In some cases, **vanishing gradients** can appear.

Objectives

- **Side-by-side comparison** of preconditioned Langevin versus their respective non-Langevin counterparts: Adam vs L-Adam, RMSprop vs L-RMSprop, Adadelta vs L-Adadelta.
- We progressively increase the depth of the network.
- Based on this heuristic, **we introduce the Layer Langevin algorithm** which consists in adding noise only to some layers of the network.
- We test (Layer) Langevin algorithms on deep image analysis architectures.

Langevin algorithms for Neural Networks with increasing depth

We compare Preconditioned Langevin optimizers with their non-Langevin counterparts while increasing the depth of the network on:

- Fully connected (Dense) neural networks,
- Convolutional layers followed by dense layers,
- Highway networks Srivastava et al. (2015) which where introduced to help propagate the gradient through the many successive layers using a parametrized residual connection: $y = T_{\theta_T}(x)\cdot D_{\theta_D}(x) + (1 - T_{\theta_T}(x))\cdot x$,

on the MNIST, CIFAR-10 and CIFAR-100 datasets.

We observe that the deeper the network is, the greater as the gains provided by Langevin optimizers in comparison with non-Langevin optimizers; for Highway networks, this is true only from a larger depth.

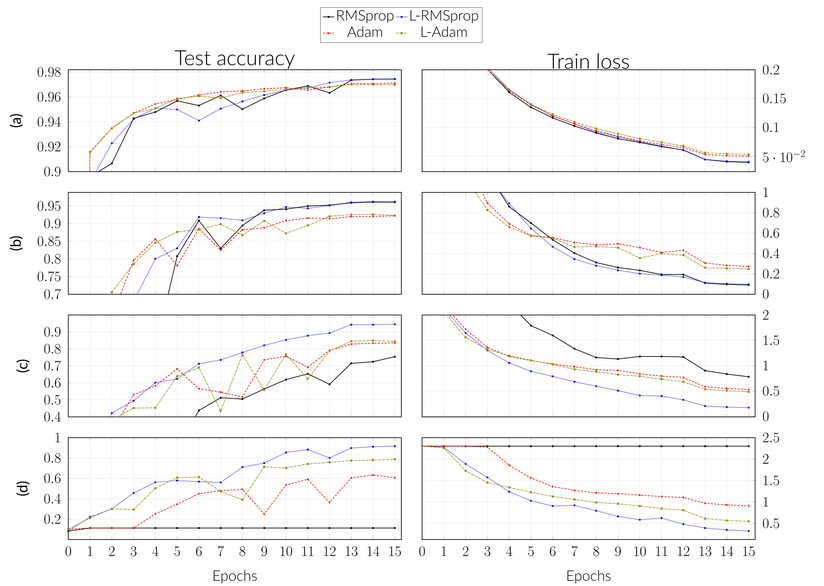

Figure 1. Training of neural networks of various depths on the MNIST dataset using Langevin algorithms compared with their non-langevin counterparts. (a): 3 hidden layers, (b): 20 hidden layers, (c): 30 hidden layers, (d): 40 hidden layers.

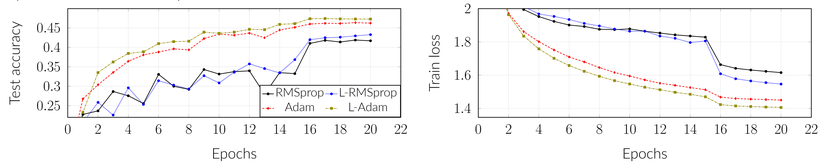

Figure 2. Training of a highway neural network with 80 highway hidden layers on the CIFAR-10 dataset.

Layer Langevin Algorithm

Idea: The deepest layers of the network bear the most non-linearities so are more subject to Langevin optimization. The Layer Langevin update reads:

$$\theta_{n+1}^{(i)} = \theta_n^{(i)} - \gamma_{n+1}[P_{n+1}\cdot g_{n+1}]^{(i)} + \mathbb{1}_{i\in\mathcal{J}}\sigma\sqrt{\gamma_{n+1}}\left[\mathcal{N}(0, P_{n+1})\right]^{(i)}, \tag{1}$$

where $\mathcal{J}$ is a subset of weight indices and P_n is the preconditioner rule. We choose $\mathcal{J}$ to be the first k layers of the network.

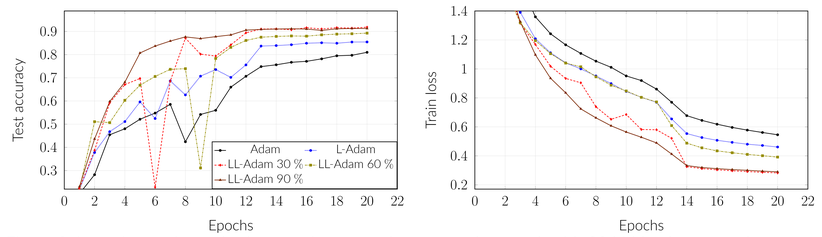

Figure 3. Layer Langevin method comparison on a dense neural network with 30 hidden layers on the MNIST dataset.

Application to deep architectures for image classification

A typical architecture in image recognition consists in a succession of convolutional layers with non-linearities; the dimensions (width and height) of the image are progressively reduced while the number of channels is progressively augmented Simonyan and Zisserman (2015). Depth is crucial and residual connections, where each layer behaves partly as the identity layer, were introduced to pass the information through the successive layers He et al. (2016); Huang et al. (2017).

We compare the performances of non-Langevin optimizers with their respective 30%-Layer Langevin counterparts.

Table 1. Final test accuracy values obtained for ResNet-20

	Adam	LL-Adam	RMSprop	LL-RMSprop	Adadelta	LL-Adadelta
CIFAR-10	76.95 %	77.39 %	84.29 %	85.14 %	75.23 %	75.74 %
CIFAR-100	45.33 %	45.41 %	55.15 %	55.68 %	42.28 %	43.84 %

Table 2. Final test accuracy values on the CIFAR-10 dataset with DenseNet architecture.

	Adam	LL-Adam	RMSprop	LL-RMSprop	Adadelta	LL-Adadelta
CIFAR-10	87.81 %	88.16 %	57.59 %	57.56 %	71.64 %	72.72 %

pierre.bras@sorbonne-universite.fr