On the importance of initialization and momentum in deep learning

Author:
Chi Zhang

November 10, 2022

Contents

1 Legacy Issues

1.1 Tuning of the model

1. Data augmentation on the training set

```python
# Import data and data enhancement
transform_train = torchvision.transforms.Compose([
torchvision.transforms.ToTensor(),
# Randomly flip the picture
# Randomly adjust brightness
torchvision.transforms.RandomHorizontalFlip(),
torchvision.transforms.RandomGrayscale(),
torchvision.transforms.RandomCrop(32, padding=4),
torchvision.transforms.Normalize(mean=[0.485, 0.456, 0.406], std=[0.229, 0.224, 0.225])
])
```

   Tuned final test set accuracy of about 80.28%. (30 epochs)

2. Dynamic tuning of learning rate

   Tuned final test set accuracy of about 82.50%. (30 epochs) 83.03%. (50 epochs)

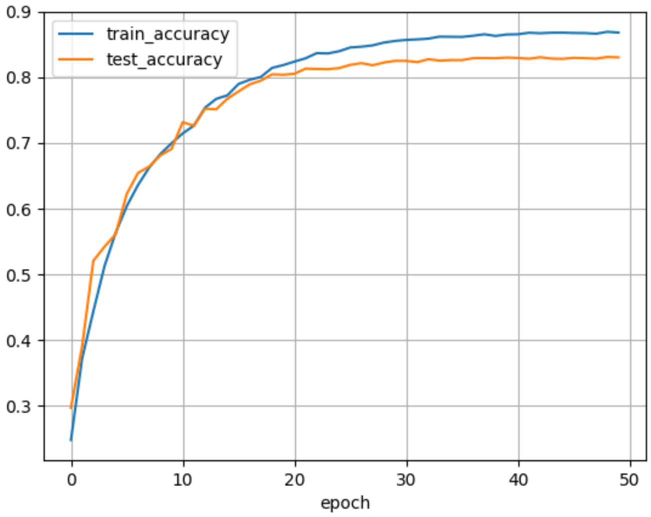


Figure 1: Accuracy in the train dataset and testset

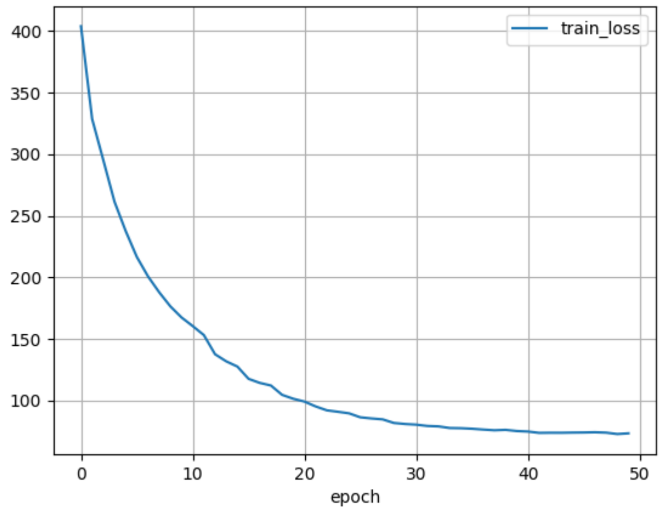


Figure 2: Loss in the train dataset

2 Introduction

1. The current dilemma of deep learning

   Deep and recurrent neural networks (DNNs and RNNs respectively) are powerful models that were considered to be almost impossible to train using stochastic gradient descent with momentum. [5]

2. What caused this dilemma

   - the use of poorly designed standard random initializations
   - failure to use a specific type of momentum wisely

3. Contribution of this paper

   - a much more thorough investigation of the difficulty of training deep and temporal networks
   - improvements in initialization and momentum meta-parameters to give better results for training
   - improvements to the HF method(Hessian-free Optimization)

3 Momentum and Nesterov's Accelerated Gradient

3.1 The SGD method

Given an objective $f(\theta)$ to be minimized, classical momentum is given by:

$$\begin{aligned} v_{t+1} &= -\varepsilon \nabla f(\theta) \\ \theta_{t+1} &= \theta_t + v_{t+1} \end{aligned} \tag{1}$$

where $\varepsilon > 0$ is the learning rate, and $\nabla f(\theta)$ is the gradient at θ_t.

3.2 The momentum method

Given an objective $f(\theta)$ to be minimized, classical momentum is given by:

$$\begin{aligned} v_{t+1} &= \mu v_t - \varepsilon \nabla f(\theta) \\ \theta_{t+1} &= \theta_t + v_{t+1} \end{aligned} \tag{2}$$

where $\varepsilon > 0$ is the learning rate, $\mu \in [0, 1]$ is the momentum coefficient, and $\nabla f(\theta)$ is the gradient at θ_t.

3.3 The Nesterov's accelerated gradient method

Given an objective $f(\theta)$ to be minimized, classical momentum is given by:

$$\begin{aligned} v_{t+1} &= \mu v_t - \varepsilon \nabla f(\theta + \mu v_t) \\ \theta_{t+1} &= \theta_t + v_{t+1} \end{aligned} \tag{3}$$

where $\varepsilon > 0$ is the learning rate, $\mu \in [0, 1]$ is the momentum coefficient, and $\nabla f(\theta)$ is the gradient at θ_t.

3.4 Why the momentum method was not taken seriously before

While the classical convergence theories for both methods rely on noiseless gradient estimates (i.e., not stochastic), with some care in practice they are both applicable to the stochastic setting.

3.5 "Recent" Findings

However, while local convergence is all that matters in terms of asymptotic convergence rates (and on certain very simple/shallow neural network optimization problems it may even dominate the total learning time), in practice, the "transient phase" of convergence (Darken Moody, 1993), which occurs before fine local convergence sets in, seems to matter a lot more for optimizing deep neural networks. In this transient phase of learning, directions of reduction in the objective tend to persist across many successive gradient estimates and are not completely swamped by noise.

3.6 The Relationship between CM and NAG

Figure 3 shows the relationship between CM and NAG.

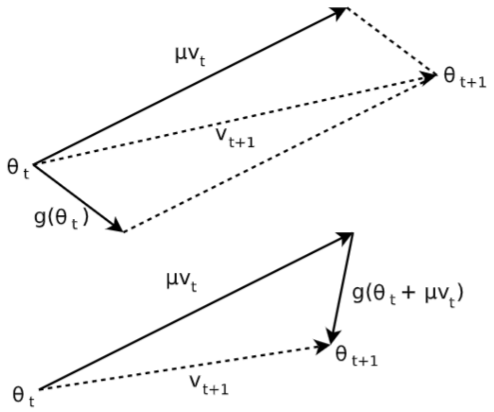


Figure 3: (Top) Classical Momentum (Bottom), Nesterov Accelerated Gradient

Figure 4 shows the trajectories of CM, NAG, and SGD.

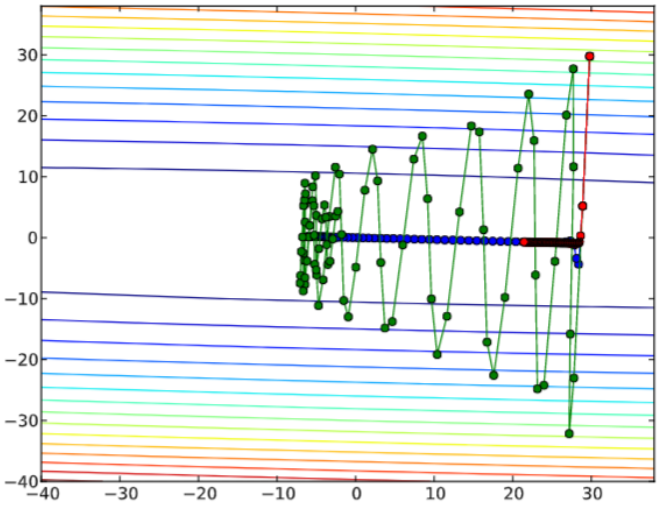
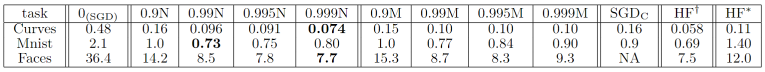


Figure 4: (Green curve) Classical Momentum, (Bule curve) Nesterov Accelerated Gradient and (Red curve) SGD

3.7 A mathematical calculation quantifies the relationship between two numerical algorithms

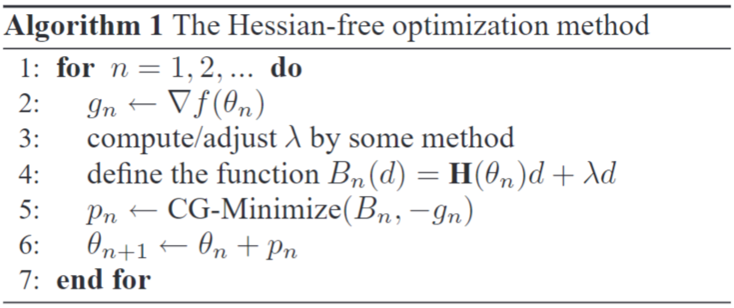
Proof in the pdf file.



task	$0_{(\mathrm{SGD})}$	0.9N	0.99N	0.995N	0.999N	0.9M	0.99M	0.995M	0.999M	$\mathrm{SGD}_{\mathrm{C}}$	$\mathrm{HF}^{\dagger}$	HF^{*}
Curves	0.48	0.16	0.096	0.091	**0.074**	0.15	0.10	0.10	0.10	0.16	0.058	0.11
Mnist	2.1	1.0	**0.73**	0.75	0.80	1.0	0.77	0.84	0.90	0.9	0.69	1.40
Faces	36.4	14.2	8.5	7.8	**7.7**	15.3	8.7	8.3	9.3	NA	7.5	12.0

Figure 5: Results of various experiments

4 Momentum and HF



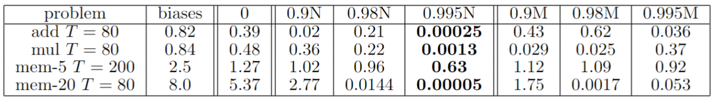
Algorithm 1 The Hessian-free optimization method

```
for $n = 1, 2, \ldots$ do
    $g_n \leftarrow \nabla f(\theta_n)$
    compute/adjust $\lambda$ by some method
    define the function $B_n(d) = \mathbf{H}(\theta_n)d + \lambda d$
    $p_n \leftarrow$ CG-Minimize$(B_n, -g_n)$
    $\theta_{n+1} \leftarrow \theta_n + p_n$
end for
```

Figure 6: The Hessian-free optimization method flowchart [2]

Here is the approximate difference formula for Hd in the algorithm.

$$Hd = \lim_{\varepsilon \to 0} \frac{\nabla f(\theta + \varepsilon d) - \nabla f(\theta)}{\varepsilon} \tag{4}$$



problem	biases	0	0.9N	0.98N	0.995N	0.9M	0.98M	0.995M
add $T = 80$	0.82	0.39	0.02	0.21	**0.00025**	0.43	0.62	0.036
mul $T = 80$	0.84	0.48	0.36	0.22	**0.0013**	0.029	0.025	0.37
mem-5 $T = 200$	2.5	1.27	1.02	0.96	**0.63**	1.12	1.09	0.92
mem-20 $T = 80$	8.0	5.37	2.77	0.0144	**0.00005**	1.75	0.0017	0.053

Figure 7: Results of various experiments

5 Discussion

In this paper we have completed this picture and demonstrated conclusively that a large part of the remaining performance gap that is not addressed by using a well-designed random initialization is in fact addressed by careful use of momentumbased acceleration (possibly of the Nesterov type).

6 Literature Research

1. Algorithm stability analysis:

   Algorithmic Instabilities of Accelerated Gradient Descent [1]

   Damped Anderson Mixing for Deep Reinforcement Learning: Acceleration, Convergence, and Stabilization [4]

2. Problem-specific optimization algorithms :

   Near Optimal Policy Optimization via REPS [3]

   An Even More Optimal Stochastic Optimization Algorithm: Minibatching and Interpolation Learning [6]

References

[1] Amit Attia and Tomer Koren. Algorithmic instabilities of accelerated gradient descent. *Advances in Neural Information Processing Systems*, 34:1204–1214, 2021.

[2] James Martens. Deep learning via hessian-free optimization. *international conference on machine learning*, 2010.

[3] Aldo Pacchiano, Jonathan N Lee, Peter Bartlett, and Ofir Nachum. Near optimal policy optimization via reps. *Advances in Neural Information Processing Systems*, 34:1100–1110, 2021.

[4] Ke Sun, Yafei Wang, Yi Liu, Bo Pan, Shangling Jui, Bei Jiang, Linglong Kong, et al. Damped anderson mixing for deep reinforcement learning: Acceleration, convergence, and stabilization. *Advances in Neural Information Processing Systems*, 34:3732–3743, 2021.

[5] Ilya Sutskever, James Martens, George E. Dahl, and Geoffrey E. Hinton. On the importance of initialization and momentum in deep learning. *international conference on machine learning*, 2013.

[6] Blake Woodworth and Nathan Srebro. An even more optimal stochastic optimization algorithm: Minibatching and interpolation learning. *arXiv: Learning*, 2021.