Improved Adam Optimizer for Deep Neural Networks

Zijun Zhang
Department of Computer Science
University of Calgary
zijun.zhang@ucalgary.ca

***Abstract*—Adaptive optimization algorithms, such as Adam and RMSprop, have witnessed better optimization performance than stochastic gradient descent (SGD) in some scenarios. However, recent studies show that they often lead to worse generalization performance than SGD, especially for training deep neural networks (DNNs). In this work, we identify the reasons that Adam generalizes worse than SGD, and develop a variant of Adam to eliminate the generalization gap. The proposed method, normalized direction-preserving Adam (ND-Adam), enables more precise control of the direction and step size for updating weight vectors, leading to significantly improved generalization performance. Following a similar rationale, we further improve the generalization performance in classification tasks by regularizing the softmax logits. By bridging the gap between SGD and Adam, we also hope to shed light on why certain optimization algorithms generalize better than others.**

I. Introduction

In contrast with the growing complexity of neural network architectures [1, 2, 3], the training methods remain relatively simple. Most practical optimization methods for deep neural networks (DNNs) are based on the stochastic gradient descent (SGD) algorithm. However, the learning rate of SGD, as a hyperparameter, is often difficult to tune, since the magnitudes of different parameters vary widely, and adjustment is required throughout the training process.

To tackle this problem, several adaptive variants of SGD were developed, including Adagrad [4], Adadelta [5], RMSprop [6], Adam [7]. These algorithms aim to adapt the learning rate to different parameters automatically, based on the statistics of gradient. Although they usually simplify learning rate settings, and lead to faster convergence, it is observed that their generalization performance tend to be significantly worse than that of SGD in some scenarios [8]. This intriguing phenomenon may explain why SGD (possibly with momentum) is still prevalent in training state-of-the-art deep models, especially feedforward DNNs [1, 2, 3]. Furthermore, recent work has shown that DNNs are capable of fitting noise data [9], suggesting that their generalization capabilities are not the mere result of DNNs themselves, but are entwined with optimization [10].

This work aims to bridge the gap between SGD and Adam in terms of the generalization performance. To this end, we identify two problems that may degrade the generalization performance of Adam, and show how these problems are (partially) avoided by using SGD with L2 weight decay. First, the updates of SGD lie in the span of historical gradients, whereas it is not the case for Adam. This difference has been discussed in rather recent literature [8], where the authors show that adaptive methods can find drastically different but worse solutions than SGD. Second, while the magnitudes of Adam parameter updates are invariant to rescaling of the gradient, the effect of the updates on the *same* overall network function still varies with the magnitudes of parameters. As a result, the effective learning rates of weight vectors tend to decrease during training, which leads to sharp local minima that do not generalize well [11].

To address these two problems of Adam, we propose the normalized direction-preserving Adam (ND-Adam) algorithm, which controls the update direction and step size in a more precise way. We show that ND-Adam is able to achieve significantly better generalization performance than vanilla Adam, and matches that of SGD in image classification tasks.

We summarize our contributions as follows:

- We observe that the directions of Adam parameter updates are different from that of SGD, *i.e.*, Adam does not preserve the directions of gradients as SGD does. We fix the problem by adapting the learning rate to each weight vector, instead of each individual weight, such that the direction of the gradient is preserved.
- For both Adam and SGD without L2 weight decay, we observe that the magnitude of each vector's direction change depends on its L2-norm. We show that, using SGD with L2 weight decay implicitly normalizes the weight vectors, and thus remove the dependence in an approximate manner. We fix the problem for Adam by explicitly normalizing each weight vector, and by optimizing only its direction, such that the effective learning rate can be precisely controlled.
- We further demonstrate that, without proper regularization, the learning signal backpropagated from the softmax layer may vary with the overall magnitude of the logits in an undesirable way. Based on the observation, we apply batch normalization or L2-regularization to the logits, which further improves the generalization performance in classification tasks.

II. Normalized Direction-preserving Adam

We present the normalized direction-preserving Adam (ND-Adam) algorithm, which essentially improves the optimization

978-1-5386-2542-2/18/$31.00 ©2018 IEEE

of the input weights of hidden units, while employing the vanilla Adam algorithm to update other parameters. Specifically, we divide the trainable parameters, θ, into two sets, θ^v and θ^s, such that $\theta^v = \{w_i | i \in \mathcal{N}\}$, and $\theta^s = \{\theta \setminus \theta^v\}$. Then we update θ^v and θ^s by different rules, as described by Alg. 1. The learning rates for the two sets of parameters are denoted by α_t^v and α_t^s, respectively.

Algorithm 1: Normalized direction-preserving Adam

```
/* Initialization                                      */
$t \leftarrow 0$;
for $i \in \mathcal{N}$ do
    $w_{i,0} \leftarrow w_{i,0} / \|w_{i,0}\|_2$;
    $m_0(w_i) \leftarrow 0$;
    $v_0(w_i) \leftarrow 0$;
/* Perform $T$ iterations of training                  */
while $t < T$ do
    $t \leftarrow t + 1$;
    /* Update $\theta^v$                                   */
    for $i \in \mathcal{N}$ do
        $\bar{g}_t(w_i) \leftarrow \partial L / \partial w_i$;
        $g_t(w_i) \leftarrow \bar{g}_t(w_i) - (\bar{g}_t(w_i) \cdot w_{i,t-1}) w_{i,t-1}$;
        $m_t(w_i) \leftarrow \beta_1 m_{t-1}(w_i) + (1 - \beta_1) g_t(w_i)$;
        $v_t(w_i) \leftarrow \beta_2 v_{t-1}(w_i) + (1 - \beta_2) \|g_t(w_i)\|_2^2$;
        $\hat{m}_t(w_i) \leftarrow m_t(w_i) / (1 - \beta_1^t)$;
        $\hat{v}_t(w_i) \leftarrow v_t(w_i) / (1 - \beta_2^t)$;
        $\bar{w}_{i,t} \leftarrow w_{i,t-1} - \alpha_t^v \hat{m}_t(w_i) / \left(\sqrt{\hat{v}_t(w_i)} + \epsilon\right)$;
        $w_{i,t} \leftarrow \bar{w}_{i,t} / \|\bar{w}_{i,t}\|_2$;
    /* Update $\theta^s$ using Adam                        */
    $\theta_t^s \leftarrow \mathrm{AdamUpdate}(\theta_{t-1}^s; \alpha_t^s, \beta_1, \beta_2)$;
return $\theta_T$;
```

As shown in Alg. 1, the proposed algorithm adapts the learning rate to each input weight vector, instead of each individual weight, such that the direction of the gradient is preserved. In addition, it strictly keeps the magnitude of each weight vector constant, and optimizes only its direction, such that the effective learning rate can be precisely controlled.

To evaluate the performance of ND-Adam, we use SGD and ND-Adam to train wide residual networks (WRN) [12] on the CIFAR-10 and CIFAR-100 datasets, and summarize the results in Table I.

TABLE I
TEST ERROR RATES OF WRN-22-7.5 AND WRN-28-10 NETWORKS ON CIFAR-10 AND CIFAR-100.

Method	CIFAR-10 Error (%)	CIFAR-100 Error (%)
WRN-22-7.5		
SGD	3.84	**19.24**
ND-Adam	**3.70**	19.30
WRN-28-10		
SGD	3.80	18.48
ND-Adam	**3.70**	**18.42**

III. Conclusion

We introduced ND-Adam, a tailored version of Adam for training DNNs, to bridge the generalization gap between Adam and SGD. ND-Adam is designed to preserve the direction of gradient for each weight vector, and produce the regularization effect of L2 weight decay in a more precise and principled way. We further introduced regularized softmax, which limits the magnitude of softmax logits to provide better learning signals. Combining ND-Adam and regularized softmax, we show through experiments significantly improved generalization performance, eliminating the gap between Adam and SGD.

References

[1] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1–9.

[2] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[3] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," *arXiv preprint arXiv:1709.01507*, 2017.

[4] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[5] M. D. Zeiler, "Adadelta: an adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[6] T. Tieleman and G. Hinton, "Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude," COURSERA: Neural Networks for Machine Learning, 2012.

[7] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.

[8] A. C. Wilson, R. Roelofs, M. Stern, N. Srebro, and B. Recht, "The marginal value of adaptive gradient methods in machine learning," in *Advances in Neural Information Processing Systems*, 2017.

[9] C. Zhang, S. Bengio, M. Hardt, B. Recht, and O. Vinyals, "Understanding deep learning requires rethinking generalization," in *ICLR 2017*, 2017.

[10] D. Arpit, S. Jastrzębski, N. Ballas, D. Krueger, E. Bengio, M. S. Kanwal, T. Maharaj, A. Fischer, A. Courville, Y. Bengio *et al.*, "A closer look at memorization in deep networks," *arXiv preprint arXiv:1706.05394*, 2017.

[11] S. Hochreiter and J. Schmidhuber, "Flat minima," *Neural Computation*, vol. 9, no. 1, pp. 1–42, 1997.

[12] S. Zagoruyko and N. Komodakis, "Wide residual networks," *arXiv preprint arXiv:1605.07146*, 2016.