Optimization Methods Other than Stochastic Gradient

- We have explained why stochastic gradient is popular for deep learning
- The same reasons may explain why other methods are not suitable for deep learning
- But we also notice that from the simplest SG to what people are using many modifications were made
- Can we extend other optimization methods to be suitable for deep learning?

Newton Method

- Consider an optimization problem

$$\min_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$$

- Newton method solves the 2nd-order approximation to get a direction $\boldsymbol{d}$

$$\min_{\boldsymbol{d}} \quad \nabla f(\boldsymbol{\theta})^T \boldsymbol{d} + \frac{1}{2} \boldsymbol{d}^T \nabla^2 f(\boldsymbol{\theta}) \boldsymbol{d} \tag{1}$$

- If $f(\boldsymbol{\theta})$ isn't strictly convex, (1) may not have a unique solution

Newton Method (Cont'd)

- We may use a positive-definite G to approximate $\nabla^2 f(\boldsymbol{\theta})$.
- Then (1) can be solved by

$$G\boldsymbol{d} = -\nabla f(\boldsymbol{\theta})$$

- The resulting direction is a descent one

$$\nabla f(\boldsymbol{\theta})^T \boldsymbol{d} = -\nabla f(\boldsymbol{\theta})^T G^{-1} \nabla f(\boldsymbol{\theta}) < 0$$

Newton Method (Cont'd)

The procedure:

while stopping condition not satisfied **do**

Let G be $\nabla^2 f(\boldsymbol{\theta})$ or its approximation

Exactly or approximately solve

$$G\boldsymbol{d} = -\nabla f(\boldsymbol{\theta})$$

Find a suitable step size α

Update

$$\boldsymbol{\theta} \leftarrow \boldsymbol{\theta} + \alpha \boldsymbol{d}.$$

end while

Step Size I

- Selection of the step size α: usually two types of approaches
  - Line search
  - Trust region (or its predecessor: Levenberg-Marquardt algorithm)
- If using line search, details are similar to what we had for gradient descent
- We gradually reduce α such that

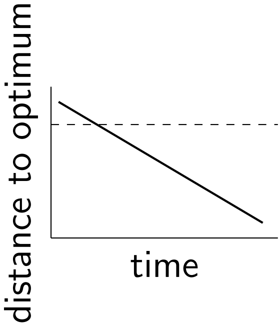
$$f(\boldsymbol{\theta} + \alpha \boldsymbol{d}) < f(\boldsymbol{\theta}) + \nu \nabla f(\boldsymbol{\theta})^T (\alpha \boldsymbol{d})$$

Newton versus Gradient Descent I

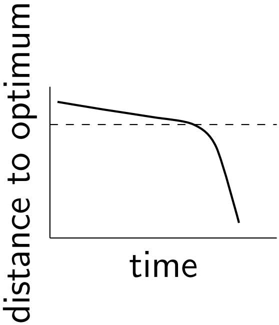
- We know they use second-order and first-order information respectively
- What are their special properties?
- It is known that using higher order information leads to faster final local convergence

Newton versus Gradient Descent II

- An illustration (modified from Tsai et al. (2014)) presented earlier



Slow final convergence



Fast final convergence

Newton versus Gradient Descent III

- But for machine learning (especially if the problem is non-convex) we mentioned that fast local convergence may not be needed
- However, higher-order methods tend to be more robust
- Their behavior may be more consistent across easy and difficult problems
- It is known that stochastic gradient is sometimes sensitive to parameters
- Thus what we hope to try here is if we can have a more robust optimization method

Difficulties of Newton for NN I

- The Newton linear system

$$Gd = -\nabla f(\theta) \qquad (2)$$

can be large.

$$G \in R^{n\times n},$$

where n is the total number of variables

- Thus G is often too large to be stored

Difficulties of Newton for NN II

- Evan if we can store G, calculating

$$\boldsymbol{d} = -G^{-1}\nabla f(\boldsymbol{\theta})$$

  is usually very expensive
- Thus a direct use of Newton for deep learning is hopeless

Existing Works Trying to Make Newton Practical I

- Many works have been available
- Their approaches significantly vary
- I roughly categorize them to two groups
  - Hessian-free (Martens, 2010; Martens and Sutskever, 2012; Wang et al., 2020; Henriques et al., 2018)
  - Hessian approximation (Martens and Grosse, 2015; Botev et al., 2017; Zhang et al., 2017)
    In particular, diagonal approximation

Existing Works Trying to Make Newton Practical II

- There are many others where I didn't put into the above two groups for various reasons (Osawa et al., 2019; Wang et al., 2018; Chen et al., 2019; Wilamowski et al., 2007)
- There are also comparisons (Chen and Hsieh, 2018)
- With the many possibilities it is difficult to reach conclusions
- We decide to first check the robustness of standard Newton methods on small-scale data

Existing Works Trying to Make Newton Practical III

- Thus in our discussion we try not to do approximations

References I

A. Botev, H. Ritter, and D. Barber. Practical Gauss-Newton optimisation for deep learning. In *Proceedings of the 34th International Conference on Machine Learning*, pages 557–565, 2017.

P. H. Chen and C.-J. Hsieh. A comparison of second-order methods for deep convolutional neural networks, 2018. URL https://openreview.net/forum?id=HJYoqzbC-.

S.-W. Chen, C.-N. Chou, and E. Y. Chang. An approximate second-order method for training fully-connected neural networks. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence*, 2019.

J. F. Henriques, S. Ehrhardt, S. Albanie, and A. Vedaldi. Small steps and giant leaps: Minimal Newton solvers for deep learning, 2018. arXiv preprint 1805.08095.

J. Martens. Deep learning via Hessian-free optimization. In *Proceedings of the 27th International Conference on Machine Learning (ICML)*, 2010.

J. Martens and R. Grosse. Optimizing neural networks with Kronecker-factored approximate curvature. In *Proceedings of the 32nd International Conference on Machine Learning*, pages 2408–2417, 2015.

J. Martens and I. Sutskever. Training deep and recurrent networks with Hessian-free optimization. In *Neural Networks: Tricks of the Trade*, pages 479–535. Springer, 2012.

References II

K. Osawa, Y. Tsuji, Y. Ueno, A. Naruse, R. Yokota, and S. Matsuoka. Large-scale distributed second-order optimization using kronecker-factored approximate curvature for deep convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

C.-H. Tsai, C.-Y. Lin, and C.-J. Lin. Incremental and decremental training for linear classification. In *Proceedings of the 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2014. URL http://www.csie.ntu.edu.tw/~cjlin/papers/ws/inc-dec.pdf.

C.-C. Wang, K.-L. Tan, C.-T. Chen, Y.-H. Lin, S. S. Keerthi, D. Mahajan, S. Sundararajan, and C.-J. Lin. Distributed Newton methods for deep learning. *Neural Computation*, 30(6): 1673–1724, 2018. URL http://www.csie.ntu.edu.tw/~cjlin/papers/dnn/dsh.pdf.

C.-C. Wang, K. L. Tan, and C.-J. Lin. Newton methods for convolutional neural networks. *ACM Transactions on Intelligent Systems and Technology*, 11(2):19:1–19:30, 2020. URL https://www.csie.ntu.edu.tw/~cjlin/papers/cnn/newton-CNN.pdf.

B. M. Wilamowski, N. Cotton, J. Hewlett, and O. Kaynak. Neural network trainer with second order learning algorithms. In *In Proceedings of the 11th International Conference on Intelligent Engineering Systems*, 2007.

H. Zhang, C. Xiong, J. Bradbury, and R. Socher. Block-diagonal Hessian-free optimization for training neural networks, 2017. arXiv preprint arXiv:1712.07296.