Deep Residual Partitioning

Neal Lawton (nlawton@usc.edu), Greg Ver Steeg (gregv@isi.edu), Aram Galstyan (galstyan@isi.edu)

University of Southern California / Information Sciences Institute

Abstract

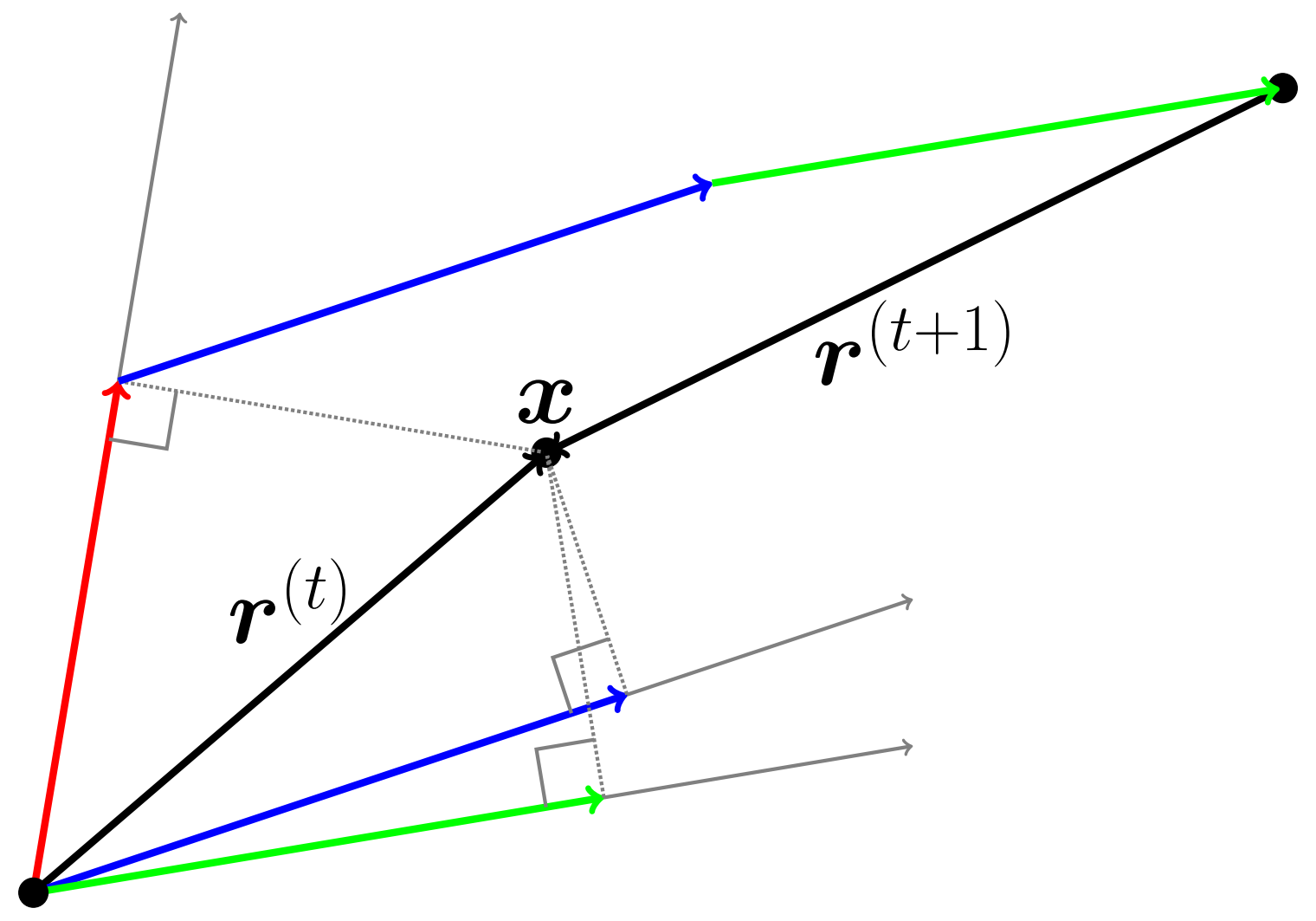
Residual partitioning is a second-order optimization algorithm for training neural nets.

- In each iteration, residual partitioning uses Jensen's inequality to bound the objective function.
- The bound has a diagonal Hessian, and so is easy to optimize.
- We compare with Adam and SGD by training an autoencoder and show residual partitioning quickly converges to a better or comparable solution.

Introduction

Training a neural net is a hard optimization problem. The curvature of the objective function varies greatly in different directions, causing SGD to zig-zag in directions of high curvature and converge slowly in directions of low curvature. Fast optimization algorithms estimate the curvature and adjust the learning rate for each parameter accordingly.

- Adaptive learning rate methods like Adam estimate the curvature from past gradients.
- Newton's method uses the full Hessian matrix, but requires solving a dense linear system in each iteration.
- Approximating the Hessian by its diagonal is fast, but struggles with the non-convexity of the objective function.



(a) Approximating the Hessian by its diagonal is like the Jacobi method for solving a least squares problem: it underestimates the curvature, and only converges under strict assumptions.

Setup

We present a simplified version of residual partitioning here: if the score function is $\mathcal{L}$, the network parameters are w_i, the output of the network is y_s, and the change in network parameters and output are Δw_i and Δy_s, then the objective function is approximately

$$\max_{\{\Delta w_i\}} \frac{1}{S}\sum_{s=1}^{S} \frac{\partial \mathcal{L}}{\partial y_s}(\Delta y_s) + \frac{1}{2}\frac{\partial^2 \mathcal{L}}{\partial y_s^2}(\Delta y_s)^2 \tag{1}$$

$$\Delta y_s \equiv \sum_{i=1}^{n} \frac{\partial y_s}{\partial w_i}\Delta w_i \tag{2}$$

The Bound

This objective function has a dense Hessian matrix: the (i, i') component of the Hessian is $\frac{1}{S}\sum_{s=1}^{S} \frac{\partial y_s}{\partial w_i}\frac{\partial y_s}{\partial w_{i'}}$. However, we can lower bound the objective by bounding $(\Delta y_s)^2$ with Jensen's inequality: first, introduce a set of *partitioning variables* $\{\varepsilon_{si}\}_{i=1}^{n}$ that add to one. Then equation (3) holds due to Jensen's inequality. Note that the Hessian matrix of the bound with respect to $\{\Delta w_i\}$ is diagonal, since the bound separates as a sum of terms, each of which involves only a single Δw_i. The final bound on the objective is achieved by plugging (3) into (1).

Residual Partitioning Bound

$$(\Delta y_s)^2 = \left(\sum_{i=1}^{n} \frac{\partial y_s}{\partial w_i}\Delta w_i\right)^2 = \left(\sum_{i=1}^{n} \frac{\varepsilon_{si}}{\varepsilon_{si}} \cdot \frac{\partial y_s}{\partial w_i}\Delta w_i\right)^2 \leq \sum_{i=1}^{n} \frac{(\partial y_s/\partial w_i)^2}{\varepsilon_{si}}(\Delta w_i)^2 \tag{3}$$

Choosing Partitioning Variables

The learning rates returned by residual partitioning depend on the choice of partitioning variables. Note that the learning rate for w_i will be large if ε_{si} is large, but the ε_{si} add to one, so the learning rates cannot all be large. We can choose the partitioning variables so the learning rates will be large on average by minimizing the sum of the inverse learning rates:

$$\min_{\{\varepsilon_{si}\}} \sum_{s=1}^{S}\sum_{i=1}^{n} \frac{\partial^2 \mathcal{L}}{\partial y_s^2}\frac{(\partial y_s/\partial w_i)^2}{\varepsilon_{si}} \tag{4}$$

The solution is

$$\varepsilon_{si} = \frac{|\partial y_s/\partial w_i|}{\sum_{i'=1}^{n}|\partial y_s/\partial w_{i'}|} \tag{5}$$

This says we should turn down the learning rate for parameters with small gradients, and turn up the learning rate for parameters with big gradients. Residual partitioning optimizes a bound on the objective, which overestimates the curvature and yields smaller than optimal learning rates. This can be fixed by scaling Δw_i by a global learning rate $\lambda > 1$.

Algorithm

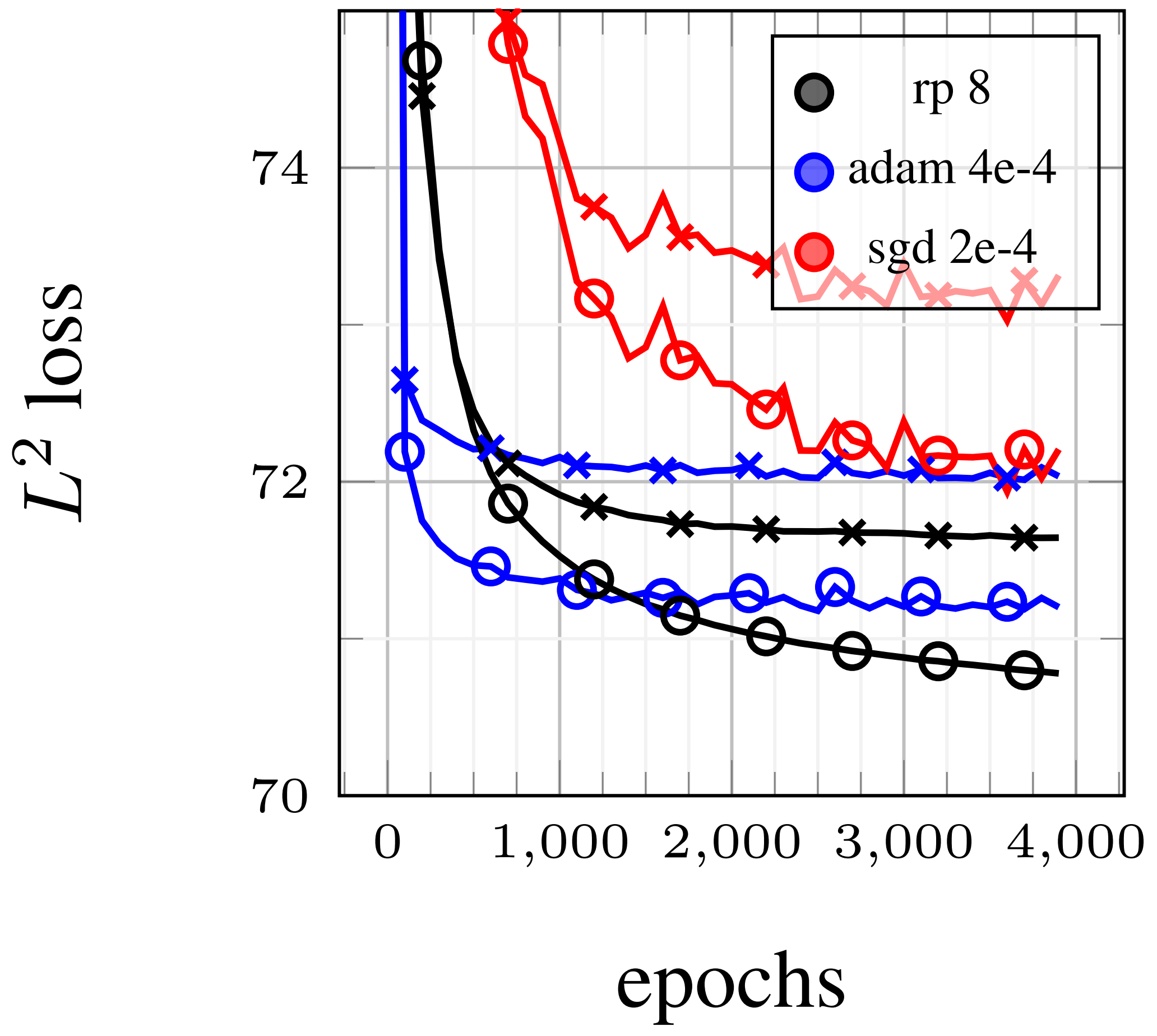
Putting it all together, residual partitioning updates the network parameters with

$$\Delta w_i = -\lambda \cdot \frac{\partial \mathcal{L}/\partial w_i}{\frac{1}{S}\sum_{s=1}^{S} \frac{\partial^2 \mathcal{L}}{\partial y_s^2}|\partial y_s/\partial w_i| \sum_{i'=1}^{n}|\partial y_s/\partial w_{i'}|} \tag{6}$$

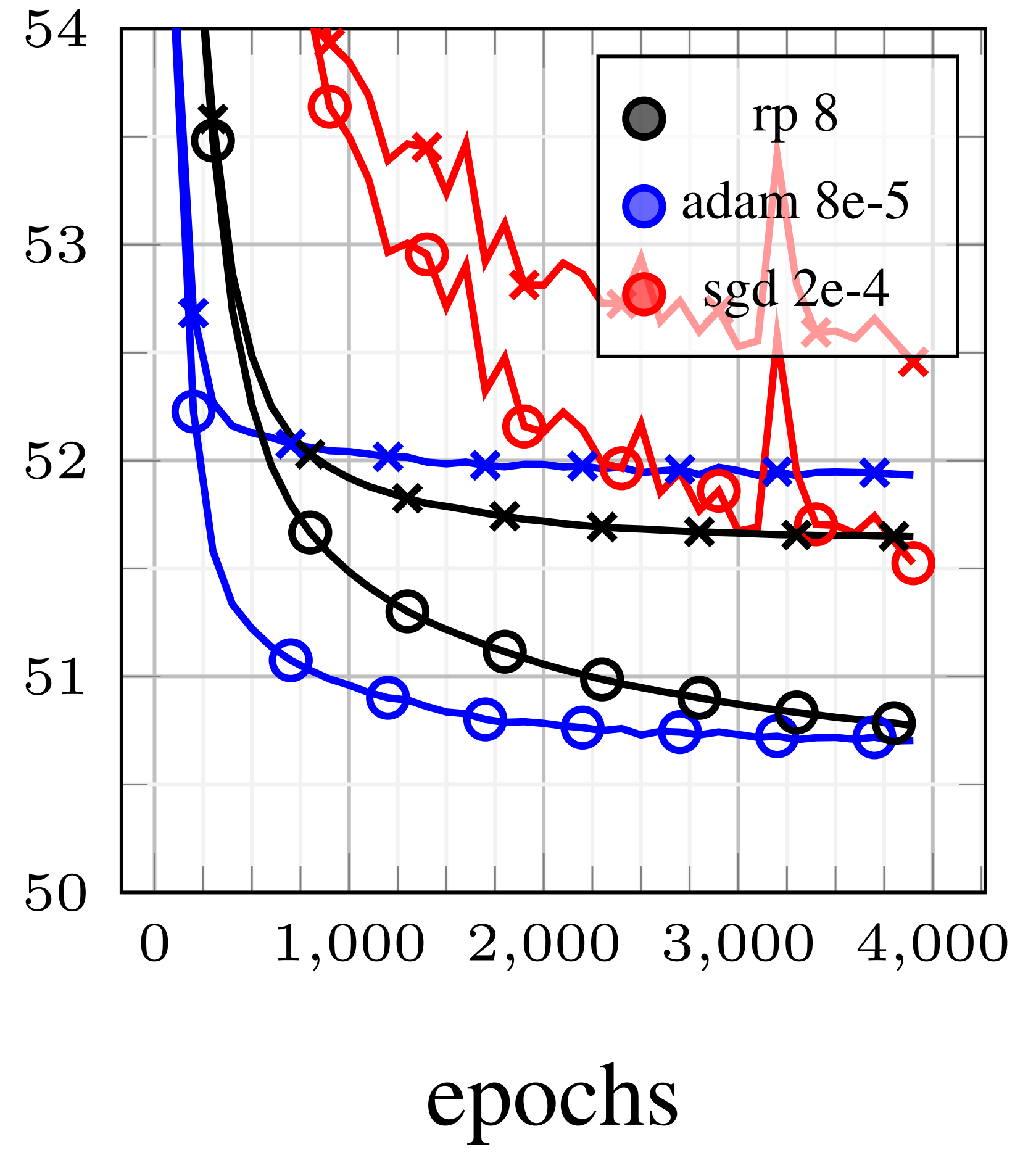
Note that since residual partitioning constructs a diagonal approximation to the Hessian, the update is simply a rescaling of the gradient.

Experiments

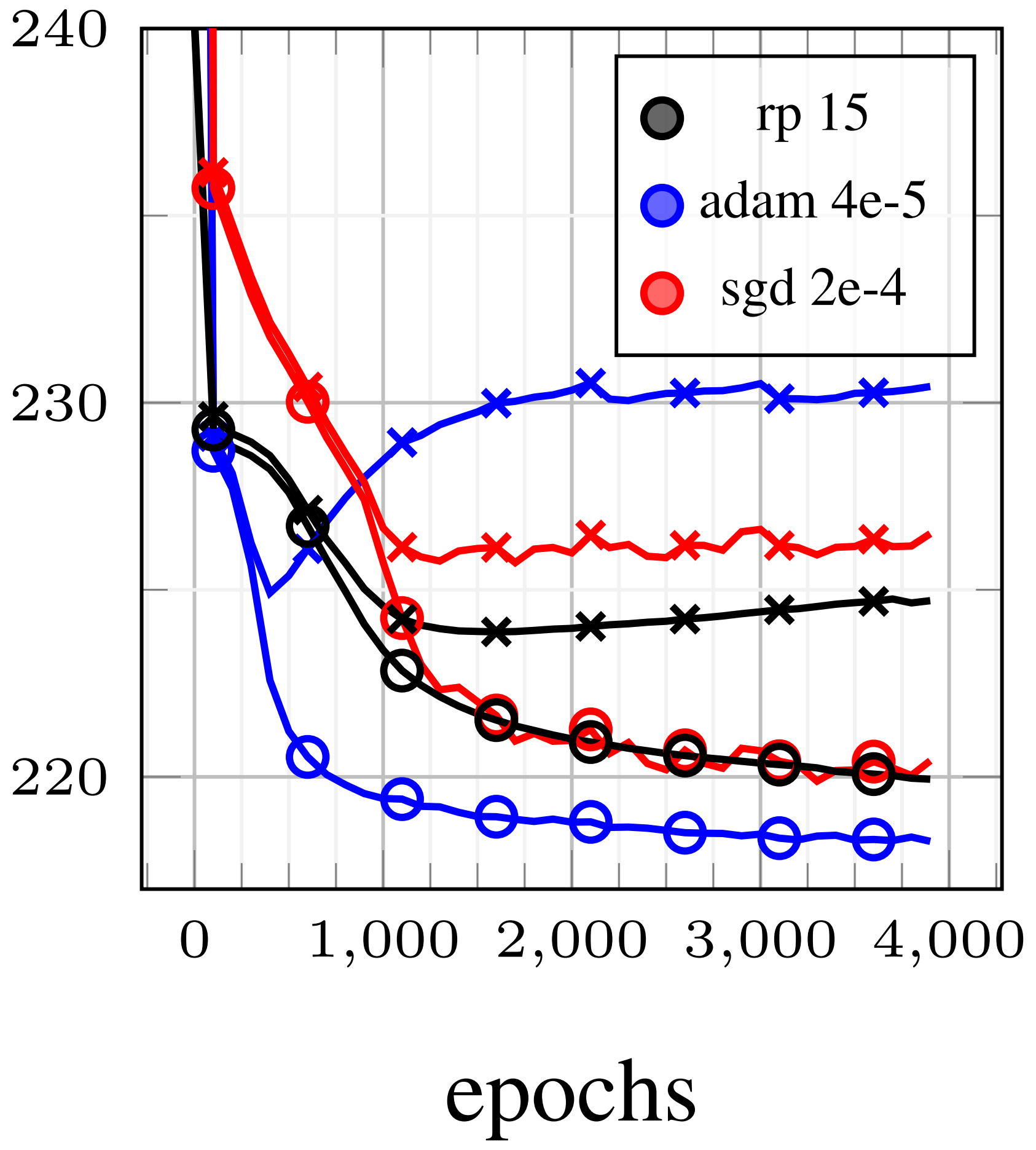
We compare residual partitioning with Adam and SGD by training an autoencoder with 5 layers on MNIST, Fashion-MNIST, and CIFAR-10. We used mini-batch sizes of 100, and tuned the global learning rate for each algorithm and dataset. The results are in Figures (b)-(d). We observe that residual partitioning quickly converges to a better or comparable solution and overfits less than Adam and SGD.



(b) MNIST



(c) Fashion-MNIST



(d) CIFAR-10