Affine Invariant Stochastic Optimization

Optimization and Big Data 2015

Jose Vidal Alcala Burgos

CIMAT and CONACYT

May 6, 2015

Stochastic Optimization

The problem at hand is to find θ_* minimizing $f(\theta)$ when we have samples W_θ; such that,

$$\nabla f(\theta) = E[W_\theta]\,. \tag{1}$$

A general Robbins-Monro iteration takes the form

$$\widehat{\theta}_{n+1} = \widehat{\theta}_n - \frac{1}{n^\gamma} K_n \, W_n \,, \tag{2}$$

where W_n is a sample of $W_{\widehat{\theta}_n}$. The optimal K_n is the inverse of the Hessian of f at the optimal θ_* and the optimal γ is 1. We follow Bottou and aim to minimize

$$\mathcal{L} = E[f(\widehat{\theta}_n) - f(\theta_*)]. \tag{3}$$

Affine Invariant Optimization

- The optimization procedure O is Affine Invariant if :

$$B\,\theta_O(f \circ B) = \theta_O(f)\,. \tag{4}$$

  for all lineal transformations $B : \mathcal{S} \longrightarrow \mathcal{S}$
- A consequence is that the optimization can not be improved by a linear transformation of the feature space.
- Second order methods, like the optimal Robbins-Monro, are affine invariant.

Linear regression approximation of H^{-1}.

Let X_n and Y_n be the corresponding $n \times (p+1)$ and $n \times p$ matrices with entries

$$x_k = (\widehat{\theta}_k, 1), \qquad y_k = W_k \ ,$$

consider the linear regression $Y = XB$ and denote the first p rows of a matrix M by $\overline{M}$. We calculate the natural estimators

$$\begin{aligned} B_n &= (X_n^T X_n)^{-1} X_n^T Y, \qquad H_n = \overline{B_n} \\ G_n &= \overline{B_n}^{-1}, \qquad K_n = \frac{G_n + G_n^T}{2} \ , \end{aligned} \tag{5}$$

and use K_n as our H^{-1} estimator and $\gamma = 0{,}6$.

- The estimator is $\overline{\theta}_n = \frac{1}{n}(\widehat{\theta}_1 + \ldots + \widehat{\theta}_n)$, Polyak averaging

Similar algorithms (with $\gamma = 1$ and no Polyak averaging) where analized by Lai and Robbins in 1981, with no numerical simulations. This optimization is Affine invariant.

Online update

We use the *online update*

$$\begin{aligned} s_{n+1} &= \frac{1}{1 + x_{n+1} P_n x_{n+1}^T} \\ u_{n+1} &= s_{n+1} \overline{P_n x_{n+1}^T} \\ v_{n+1} &= y_{n+1} - x_{n+1} B_n \\ t_{n+1} &= \frac{1}{1 + v_{n+1} G_n u_{n+1}} \\ G_{n+1} &= G_n - t_{n+1} G_n u_{n+1} v_{n+1} G_n \\ B_{n+1} &= B_n + s_{n+1} P_n x_{n+1}^T v_{n+1} \\ P_{n+1} &= P_n - s_{n+1} P_n x_{n+1}^T x_n P_n^T \end{aligned} \tag{6}$$

- $P_n = (X_n^T X_n)^{-1}$ is the precision matrix.
- $O(p^2)$ completely parallelizable operations .

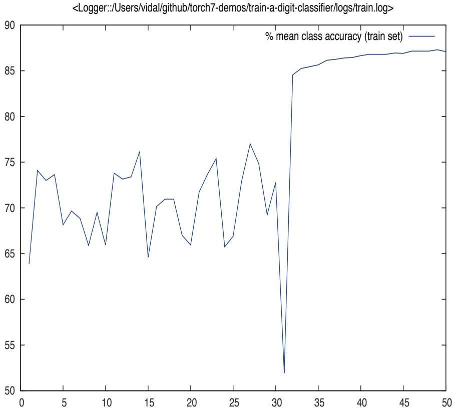
Torch7

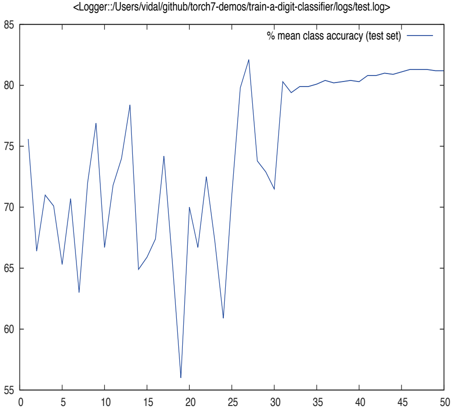
Opensource machine learning library in Lua (scripting) maintained by Idiap Research Institute, NYU and NEC Laboratories America.

- Supports CUDA and OpenMP
- Recent neural networks, like Dropout, are implemented.
- Many optimization methods are implemented.

MNIST

- Convolutional neural network with 4,000 parameters.
- Minimize negative log likelihood
- We run experiments in Tesla K40, provided by NVIDIA.
- We can not increase the number of parameters because we run out of memory.





Low rank approximation (Matthew Brand)

Given the *thin SVD* decomposition $G = USV^T$, with $S \in M_{r\times r}$, find the decomposition of the rank one update

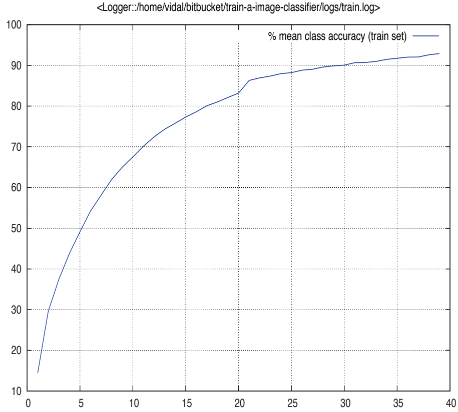
$$G + ab^T = \overline{U}\,\overline{S}\,\overline{V}^T, \qquad \overline{S} \in M_{(r+1)\times(r+1)}\,,$$

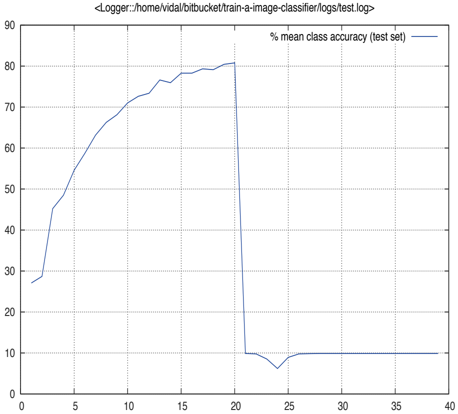
with the steps

$$\begin{aligned}
m &= U^T a; \quad p = a - Um, \quad R_a = \|p\|\,; \quad P = R_a^{-1} p\,,\\
n &= V^T b; \quad q = b - Vn, \quad R_b = \|q\|\,; \quad Q = R_b^{-1} q\,,\\
K &= \begin{bmatrix} S + mn^T & \|q\|\,m \\ \|p\|\,n^T & \|p\|\,\|q\| \end{bmatrix}, \quad K = U'S'V'^T\\
\overline{U} &= [U\;P]\,U'; \qquad \overline{S} = S'; \qquad \overline{V} = [V\;Q]\,V'
\end{aligned}$$

CIFAR-10

- Convolutional neural network with Dropout and 9,000,000 parameters
- Minimize negative log likelihood
- We run experiments in Tesla K40. Thank you again NVIDIA !.
- Accuracy increases only on the test set. Conclusion: even with Dropout, we have overfitting in the model.





References

- Jose Vidal Alcala Burgos, Optimizing the exercise boundary for the holder of an American option over a parametric family, Ph.D. Thesis, ProQuest (2012).

  `http://gradworks.umi.com/35/24/3524127.html`

- Code for the *affine invariant* algorithm is available at https://github.com/vidalalcala/sopt-ols
- Code for MNIST is available at https://bitbucket.org/vidalalcala/affine-invariant-sopt
- Code for CIFAR *affine invariant* is available at https://bitbucket.org/vidalalcala/train-a-image-classifier with password.