[Qiuz1] Perceptron and Hinge Loss

Bonggun Shin BONGGUN.SHIN@EMORY.EDU
CS, Emory University, Atlanta, GA

Abstract

This paper explains the difference between perceptron and adaptive gradient decent, and discusses the effect of averaging. In addition, it presents some notes on L2-regularization.

1. Perceptron vs. AdaGrad. and their effects of averaging

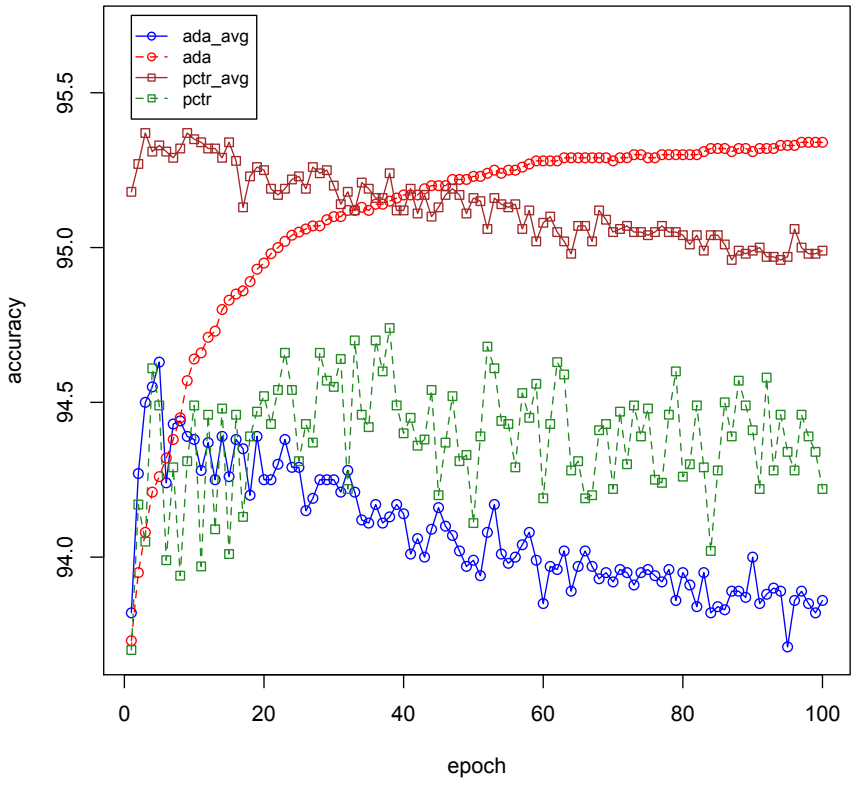

Figure 1: Perceptron vs. AdaGrad.

1.1. Findings

- **As epoch repeats, the averaged AdaGrad keeps improving, but initially the performance is not good.**

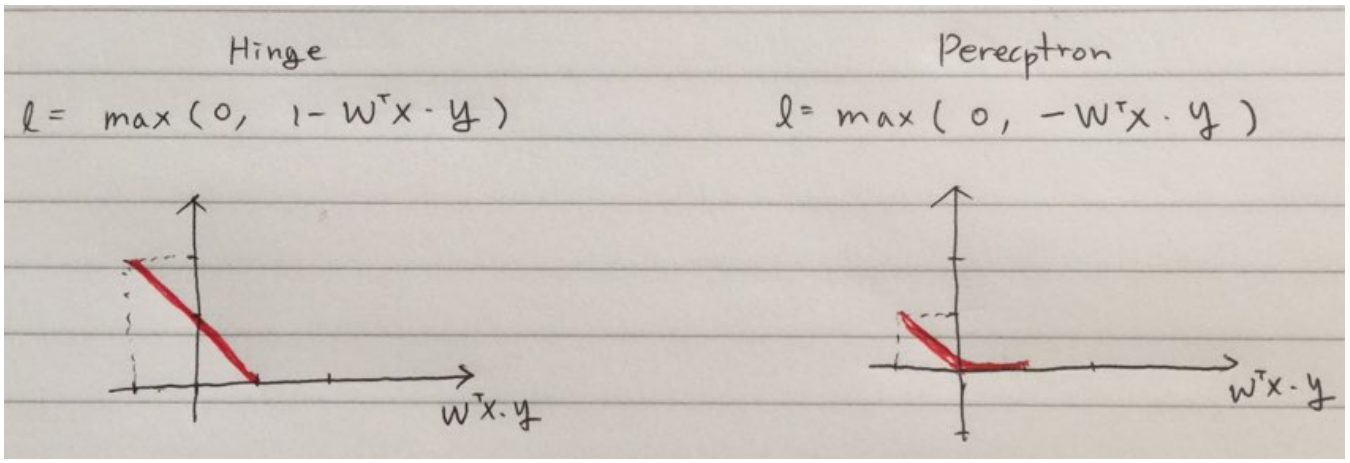

Figure 2: Loss functions

  - **steady improveness**: The paramount reason for it is adaptive gradient. As epoch goes it adjust update amount so that fine tuning is possible. Inadition, As shown in the Fig. 2, AdaGrad reflects error linearly, because $-\mathbf{w^T x} \cdot y$ varies from -1 to 1. Hence the loss is linear. However, Perceptron squeezes loss to 0 when $-\mathbf{w^T x} \cdot y$ is greater than 0, and it linearly reflects only when $-\mathbf{w^T x} \cdot y$ is [-1,0].
  - **Initial performance**: Adaptive η initially has a big value, which sometimes make big jump update and it could make performance bad. Also, since the weights are initialized to zero, the weight are small in the early stage. This makes weights to be frequently updated (because the updates occur when $-\mathbf{w^T x} \cdot y < 1$, satisfied most of time), but initial guesses are likely to be incorrect most of times. This explains low performance of the initial epoch.

- **Generally, averaging has adversary effect for later stage of learning** The features are sparse, which implies there are many items on the boundary that are equally important. We can't treat them as noise, rather should put impartial weights on them. However, when averaged, this (presumably) heavy tail gets small weights, and the performance is lowered.
- Averaging make oscillation small. (intuitive)
- AdaGrad performs bad when averaged, while Perceptron shows improvement of averaging.
  - **Averaged AdaGrad** In later stage, it has finer updates and yet more accurate, but gets small weights compared to what is done badly. However, bad memories have big impact in the whole epoch because it had a big update.
  - **Averaged Perceptron** It has identical η, and not accurate gradient compared to AdaGrad, hence big oscillation during all epochs. Therefore averaging helps increasing the performance.

2. L-2 Regularization

2.1. Gradients

Perceptron
$l = \max\{0, -\mathbf{w}^\mathbf{T}\mathbf{x} \cdot y\} + \lambda||\mathbf{w}||^2$
$\Delta l = -x \cdot y + \lambda\mathbf{w}$, when $-\mathbf{w}^\mathbf{T}\mathbf{x} \cdot y < 0$,
$\Delta l = \lambda\mathbf{w}$, otherwise

AdaGrad
$l = \max\{0, 1 - \mathbf{w}^\mathbf{T}\mathbf{x} \cdot y\} + \lambda||\mathbf{w}||^2$
$\Delta l = -x \cdot y + \lambda\mathbf{w}$, when $-\mathbf{w}^\mathbf{T}\mathbf{x} \cdot y < 1$,
$\Delta l = \lambda\mathbf{w}$, otherwise

2.2. Results

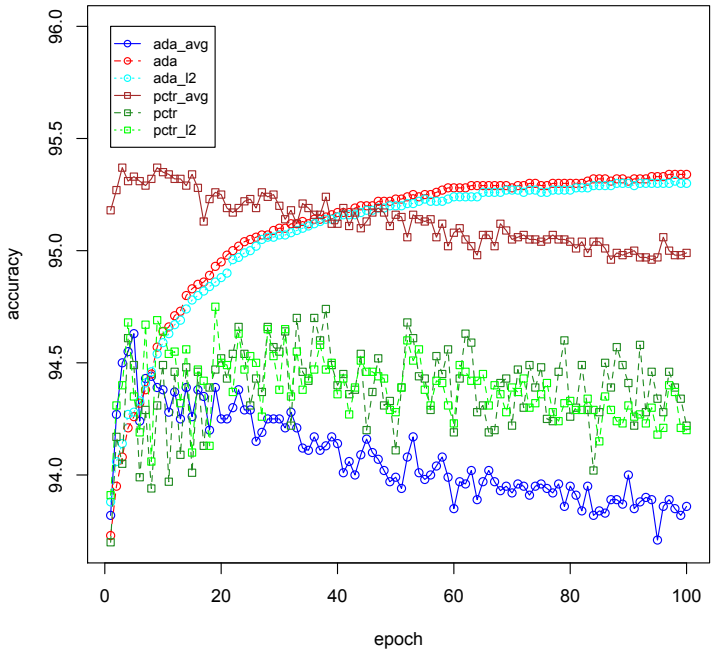

Figure 3: L-2 Regularization

I put $\lambda = 1$. Cyan colored circles in the Fig. 3 show the L-2 regularized version of the AdaGrad. Its performance slightly lower than non regularized version. This is because the regularizer keeps the model from overfitting to the data. The same phenomenon also shows in the perceptron case (lighter green squares in the Fig. 3).