CS 6.5: Theories of Deep Learning
Problem Sheet 3

Prof. Jared Tanner

February 11, 2019

Submission Due: 10:00 AM, 19-02-2019

Gradient-Descent based optimization for neural networks
Gradient descent is one of the most popular algorithms to perform optimization and by far the most common way to optimize neural networks. Gradient descent is a way to minimize an objective function $J(\theta)$ parameterized by a model's parameters θ by updating the parameters in the opposite direction of the gradient of the objective function $\nabla J(\theta)$ w.r.t. to the parameters. The learning rate η determines the size of the steps we take to reach a (local) minimum. In other words, we follow the direction of the slope of the surface created by the objective function downhill until we reach a valley.

The aims of this exercise is to provide you with intuitions towards the behaviour of such optimizing algorithms and associated challenges during actual training in practice. While, in class we will discuss a few of such algorithms, this exercise focuses on 'ADADELTA'[1]. ADADELTA, try to adjust the learning rate during training by annealing, i.e. reducing the learning rate according to a pre-defined criteria. However, the same learning rate doesn't applies to all the dimentions of θ, and depends on the frequency of their use in overall learning process. Additionally, with ADADELTA, we do not even need to set a default learning rate η, as it has been eliminated from the update rule.

1. Task1: Write a short report summarizing the ADADELTA algorithm. Discuss, the advantages and disadvantageous you might think it has for training networks and also compared to vanilla stochastic gradient descent (SGD). Your report should be written in the format and style of a NIPS Proceedings, abridged to not exceed 2 pages. Latex style files and an exemplar template are provided on the course page, and are similar to last exercise.

2. Task2: Weight Decay: One of the ways to update the parameters θ is to consider the updates of form:
$$\theta_{t+1} = (1-\lambda)\theta_t - \eta \nabla J_t(\theta_t), \tag{1}$$
where λ defines the rate of the weight decay per step and $\nabla J_t(\theta_t)$ is the t-th batch gradient to be multiplied by a learning rate η. Prove that the standard SGD with base learning rate η executes the same steps on $J_t(\theta)$ with weight decay λ (defined in Equation 1) as it executes without weight decay on $J_t^{'}(\theta) = J_t(\theta) + \frac{\lambda^{'}}{2}\|\theta\|_2^2$, with $\lambda^{'} = \frac{\lambda}{\eta}$.

3. Task3: Weight Decay for adaptive gradients based approaches[2] (such as ADADELTA, ADAGRAD, ADAM): Let O be an optimizer that iterates as $\theta_{t+1} = \theta_t - \eta \mathbf{M}_t J_t(\theta_t)$ without weight decay, and $\theta_{t+1} = (1-\lambda)\theta_t - \eta \mathbf{M}_t J_t(\theta_t)$ with weight decay, with $\mathbf{M}_t \neq k\mathbf{I}, k \in \mathbb{R}$. Prove that for O there exists no coefficient $\lambda^{'}$ such that running O on $J_t^{'}(\theta) = J_t(\theta) + \frac{\lambda^{'}}{2}\|\theta\|_2^2$ without weight decay is equivalent to running O on $J_t(\theta)$ with decay $\lambda \in \mathbf{R}^+$

[1] ADADELTA:https://arxiv.org/pdf/1212.5701.pdf
[2] Survey on GD:https://arxiv.org/pdf/1609.04747.pdf