DEEP LEARNING (SS2022)
SEMINAR 4

Assignment 1 (Weight initialization for ReLU networks)**.** In this assignment we derive a proper weight initialization for ReLU networks. We will assume that the components of all vectors are statistically independent and identically distributed.

a) Let us consider a single neuron with weight vector w and input vector x. Its pre-activation is $a = w^T x$. Let us denote

$$\mathbb{E}[x_i] = \mu,\ \mathbb{E}[x_i^2] = \chi,\ \mathbb{E}[w_i] = 0, \text{ and } \mathbb{V}[w_i] = v.$$

prove that $\mathbb{E}[a] = 0$ and $\mathbb{V}[a] = nv\chi$, where n is the dimension of the vectors x and w.

b) Show that the distribution of a is symmetric if so is the distribution of w.

c) Consider the neuron output $y = g(a)$, where g denotes the ReLU function. Conclude that $\mathbb{E}[y^2] = \frac{1}{2}\mathbb{V}[a]$.

d) Let us denote $\mathbb{V}[a] = \alpha$ and consider a ReLU network with layers $k = 1, \ldots, m$. Collecting the previous steps we get the following recursive relation for the α_k

$$\alpha_k = \frac{1}{2} n_{k-1} v_k \alpha_{k-1}$$

and obtain the initialisation proposed by He et al. (2015): initialise the weights with zero mean and variance

$$\mathbb{V}[w_{ij}^k] = \frac{2}{n_{k-1}}.$$

Assignment 2 (Batch Normalization)**.** Batch normalization after a linear layer with a weight matrix W and bias b takes the form:

$$\frac{Wx + b - \mu_{\mathcal{B}}}{\sigma_{\mathcal{B}}}\beta + \gamma, \tag{1}$$

where $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}$ denote the mean and standard deviation of the layer output $a = Wx + b$ taken over a batch.

a) Show that the output of batch normalization does not depend on the bias b and also does not change when the weight matrix W is scaled by a positive constant.

b) What is the mean and standard deviation of the BN-normalized layer, if we initialize $\beta = 1$, $\gamma = 0$? Assume, we decided to apply BN after each linear layer. Has the weight initialization from Assignment 1. still an effect for the forward pass? Why could it nevertheless be important for training?

c) Consider a network without BN. Let $\mu_{\mathcal{B}}$ and $\sigma_{\mathcal{B}}$ be the statistics of layer output $a = Wx + b$. We want to introduce a BN layer at this place so that it does not change the network predictions. How shall we initialize β and γ?

Assignment 3 (Dropout, Bernoulli)**.**

a) The dropout noise model can be reformulated for a more convenient implementation. Consider the following Bernoulli noises:

$$Z = \begin{cases} a, & \text{with probability } p \\ 0, & \text{with probability } 1-p \end{cases} \tag{2}$$

What should be the value of a so that $\mathbb{E}[Z] = 1$ holds? This will allow to avoid rescaling of the weights at test time by just applying this noise at training time.

b) Randomized procedures for quantized gradients are sometimes used for a faster communication in distributed systems (if we want to parallelise training).

Let the gradient $g \in \mathbb{R}^n$ be computed at the worker. The worker sends a *quantized* gradient $\tilde{g} \in \{0,1\}^n$ to the server, using only 1 bit per coordinate. The worker additionally sends two real numbers a, b and the server reconstructs the gradient as $a\tilde{g} + b$. How to chose the quantization procedure in a randomized way so that $\mathbb{E}[a\tilde{g} + b] = g$ and hence we preserve the guarantee of an unbiased (but more noisy) gradient estimate? Is the choice of a and b that satisfy this assumption unique? How to choose a and b such that $\mathbb{E}[a\tilde{g} + b] = g$ and the variance of $a\tilde{g} + b$ is minimal?

Assignment 4 (SGD + L2)**.** Consider a regularized loss function $\tilde{L}(\theta) = L(\theta) + \frac{\lambda}{2}\|\theta\|^2$. Let g be a stochastic gradient estimate of L at θ. Notice that the regularization part of the objective, $\frac{\lambda}{2}\|\theta\|^2$, is known in a closed form and so its gradient g_r is non-stochastic.

- Design an SGD algorithm that applies momentum (exponentially weighted averaging) to g only but not to g_r.
- Is it equivalent to an SGD with the momentum applied to both g and g_r but possibly with a different settings of λ, momentum and learning rate?