Authors:

- Mhd Jawad Al Rahwanji - 7038980 - mhal00002@stud.uni-saarland.de
- Christian Singer - 7039059 - chsi00002@stud.uni-saarland.de

Exercise 8.1 - Optimization Algorithms

a) Convexity of NN

   a) No, Deep Neural Networks are not convex in nature. There exists no guarantee for the Hessian matrix to either be positive or negative semidefinite. The values in the diagonal have arbitrary signs and the larger the Hessian the more unlikely it is for the values to be unanimously positive or negative. Hence, the cost function contains saddle points, local minima and possibly a global minimum or many. As a result we use methods like GD for optimization to efficiently navigate the hyper space and preferably find the global minima.

   b) It is highly beneficial for a Deep Neural Network to be convex or concave. A well behaved Hessian allows for better convergence, that is, finding a global minimum/maximum rather than one of the local minima which is the case with ill behaved (non-convex) Hessians. An easier to traverse search space with higher reproducibility. A perfectly convex cost function alleviates the need for optimization algorithms for we can simply find the global minimum using multivariate derivation.

b) AdaGrad Optimizer

   a) AdaGrad adapts the weight updates to each individual parameter and its importance.
   When dealing with sparse data, AdaGrad performs larger updates to less supported parameters.
   So it assigns a learning rate to each parameter as follows:

$$\theta_{t+1,i} = \theta_{t,i} - \frac{\eta}{\sqrt{G_{t,ii} + \epsilon}} \cdot \nabla_{\theta_t} J(\theta_{t,i})$$

   It updates the learning rate w.r.t θ_i at every t using G_t which is a diagonal matrix that represents the sum of the squares of all the past gradients w.r.t to θ up to t. Lastly, ϵ is a term to prevent division by zero.

b) An important property of AdaGrad is that it eliminates the need to tune the learning rate as a hyperparameter. Since it adapts the learning rate to the weights (parameters) regardless of the dataset, hence, Ada from adaptive, its use improves the robustness of SGD.

c) One main shortcoming of AdaGrad is that over however many t 's $G_{t,ii}$ accumulates and becomes large enough to cause η to diminish which in turn limits the model's learning capabilities.

c) Adam Optimizer

a) Yes, both RMSProp and Momentum are gradient descent optimization algorithms used to train deep learning models. They both try to improve the optimization of the model by making the optimization process more efficient and faster.

RMSProp and Momentum can be combined to "RMSProp with Momentum" whose update rule combines both RMSProp and Momentum. The update rule for the weights at each iteration is given by:

*w(t+1) = w(t) - learning_rate * gradient(t) / sqrt(v(t) + epsilon) - learning_rate * momentum * m(t)*

Where v(t) is the exponentially weighted moving average of the squares of the gradients and m(t) is the exponentially weighted moving average of the gradients.

b) Besides storing an exponentially decaying average of past squared gradients v_t like RMSprop, Adam also keeps an exponentially decaying average of past gradients mt, similar to momentum.

One advantage of the Adam update rule is that it can make more efficient use of the data available for training. Because it uses moving averages of both the gradient and the squared gradient, it can adapt the learning rate for each parameter based on how fast the parameter changes. This can help the model converge more quickly and with better accuracy.

Another advantage of Adam is that it requires fewer hyperparameters than other optimization algorithms, such as RMSProp with momentum. This can make it easier to tune the model and can result in faster training times.

d) AdamW

   AdamW is an extension of the Adam optimization algorithm that has been proposed to address the problem of weight decay regularization in Adam.

   In Adam, weight decay regularization can be implemented by adding a penalty term to the objective function that is being optimized. This penalty term is usually of the form L2 * w^2, where w is the weight vector of the model and L2 is a hyperparameter that controls the strength of the regularization.

   However, this approach has the potential to interact poorly with the adaptive learning rate of Adam, which can result in suboptimal convergence.

   To address this issue, AdamW introduces a correction factor to the weight updates that are designed to decouple the weight decay from the adaptive learning rate.

As additional sources, we used the book “Deep Learning” by I. Goodfellow and the paper “Decoupled Weight Decay Regularization” introducing AdamW.