Intuition Report 3

Ramtin Mojtahedi Saffari - 20307293

Assessed link 1: http://playground.tensorflow.org

Self-reflection

- **Short Summary**

The link provides an interactive tool to build a neural network architecture with six hidden layers and different datasets and features. The tool also allows us to change some of the network's hyperparameters, such as learning rate, activation function type, regularization rate, and even the problem type of classification or regression.

- **Hypothesis and Expectation**

I hypothesized that by selecting ReLU as our activation function and decreasing the regularization rate, we can improve the model's final performance, which the loss function values can prove in the training and testing dataset.

- **What I Achieved**

The best performance was achieved when I used a deep network with the regularization of L2 with a regularization rate of 0.001 and used ReLU as the activation function. For all of the experiments, I trained the model for 100 epochs, and other parameters were considered fixed to make an equal comparison (e.g., lr:0.01/noise and ratio of data split)

- **What I Learned**

From several experiments, I understood that a deeper network provides a better result. However, this decreases the training time and makes the model more complex. Also, the learning rate considered as 0.01 as high or low learning completely drops the results achieved in the loss values (achieved around 0.5, which is equal to the random loss values as this was a binary classification). A tiny learning rate causes a long training process that could stick the model and a large learning rate, whereas a value too large may result in learning a sub-optimal set of weights too fast or an unstable training process [1].

Regarding the activations function, I tested different functions and found that ReLU achieved the best result. One of the advantages of the ReLU is that it doesn't activate all of the neurons simultaneously and reduces the probability of vanishing gradients [2]. For these reasons, it is always considered as the first option for the activations function. The other assessed parameter is the regularization method. In this way, L2 worked better than L1. This is mainly because that L2 distributes the error terms in all weights and is not resistant to the outliers [3]. The other issue was to select an appropriate regularization rate. I found that the rate of 0.001 is a good choice and provided me with descent loss values. As understood from [4], regularization rate should be selected in the range of [0-0.1] and 0.001 s a god choice as to if we choose the regularization rate with a high value; then this causes underfitting, which increases the loss values and prevents to make a generalized

model. Overall, I achieved the best performance on the model with six hidden layers (8 neurons in each layer), lr:0.01, the activation function of ReLU, and L2 regularization with the rate of 0.001. in this case, the loss values for the 100 epochs achieved about 0.001 in training loss and 0.002 in test loss values.

Suggestion and Filling the Gaps

I found the interactive link very interesting and helpful in learning-by-practice the basic concepts in a neural network. I suggest that the developed add some explanations to the proposed website to guide the authors. They can use reference [5] for providing their materials to the website.

Assessed link 2: Gradient toys with TF.js

Self-reflection

- **Short Summary**

The website provides interactive graphs to understand concepts and differences between activation functions, Norm Clipping, Stopgrad, inverse kinematic chain toy, and fractal trees. By changing the location of different graphs, we can understand how they affected each other or the rationale behind each concept.

- **Hypothesis**

As a related graph to this week's materials, I hypothesized that when I increase the positive values of x, ReLU demonstrates a monotonous increasing speed; swish provides the same trajectory except for early positive values. The clip has the most, and the sigmoid indicates the slowest speed. On the other hand, for the

negative values, the ReLU and swish can’t go beyond zero values, and clip and sigmoid demonstrate the same sped and trajectory compared to the positive values. These can be translated into how this activation function affects gradients based on their values.

- **Testing the hypothesize and what I have learned**

By considering the mathematical model of the sigmoid function, it was expected that it could be +0.73 when the x reached out to the +1 and -0.73 when x reached out to -1. This demonstrates the symmetric trajectory and pattern that this function can render based on the input values. For the Clip function, when the x values become more than 0.5 or -0.5, they can’t go beyond 0.5 or -0.5, respectively. This shows that this function renders outputs faster than the other activation functions, and it acts symmetrically for the positive and negative input values.

The ReLU and Swish are the ones that have a smoother trajectory (this affects the rendered output). The ReLU is only working with the positive x values. However, Swish values are lower than ReLU until x reaches 0.5, and then both show a similar trajectory and speed. When the x values decrease in the negative direction, swish slightly goes less than zero, and after reaching x to values of -0.5, it returns to zero. Beyond that, it remains at zero (here, it reminds me of the similarity with leaky ReLU). The swish acts similarly or outperforms ReLU due to its non-monotonic smooth attribute and benefits when optimizing the model for convergence towards the minimum loss [6]. Also, due to the achieved trajectory of the sigmoid, we are more interested in using it when we have to predict the probability as an output [7]. As mentioned above, our first choice in implementing the networks is using ReLU due to its simplicity, sparsity, and prevention of gradient descent vanishing.

Suggestion and Filling the Gaps

Similar to what I mentioned before, I found the interactive link very interesting and helpful in learning-by-practice the basic concepts in a neural network. I suggest that the developed add some explanations to each of the proposed interactive tools. For example, they can use the materials in the reference [8] to explain the differences between activations functions.

Self-evaluation:

In this intuition report, I have gone through an in-depth analysis of two of the provided links. In my assessment, I have completely considered the required expectation for deep exploration, including proposing a hypothesis and what I expected, reporting on what I achieved and explored, in-depth discussion of what I have learned, and providing gaps and recommendations to fill them. Considering the quality and assessment level, I deserve to get the full mark (4 points) for this intuition report.

In advance, thank you very much for your time and consideration of this report.

Best regards,

Ramtin

References

[1] Zulkifli, H. (2019, March 28). *Understanding Learning Rates and How It Improves Performance in Deep Learning*. Medium. https://towardsdatascience.com/understanding-learning-rates-and-how-it-improves-performance-in-deep-learning-d0d4059c1c10

[2] K, B. (2020, November 7). Understanding ReLU: The Most Popular Activation Function in 5 Minutes! Medium. https://towardsdatascience.com/understanding-relu-the-most-popular-activation-function-in-5-minutes-459e3a2124f

[3] Tyagi, N. (2021). L2 vs L1 Regularization in Machine Learning | Ridge and Lasso Regularization. L2 and L1 Regularization in Machine Learning. https://www.analyticssteps.com/blogs/l2-and-l1-regularization-machine-learning

[4] Brownlee, J. (2020, August 25). *How to Use Weight Decay to Reduce Overfitting of Neural Network in Keras*. Machine Learning Mastery. https://machinelearningmastery.com/how-to-reduce-overfitting-in-deep-learning-with-weight-regularization/

[5] Stewart, M. P. R. (2020, July 30). *Simple Guide to Hyperparameter Tuning in Neural Networks*. Medium. https://towardsdatascience.com/simple-guide-to-hyperparameter-tuning-in-neural-networks-3fe03dad8594

[6] C. (2020, February 2). *Why swish could perform better than ReLu –*. MachineCurve. https://www.machinecurve.com/index.php/2019/05/30/why-swish-could-perform-better-than-relu/

[7] Wikipedia contributors. (2021, September 17). *Sigmoid function*. Wikipedia. https://en.wikipedia.org/wiki/Sigmoid_function

[8] Jain, V. (2020, March 28). *Everything you need to know about "Activation Functions" in Deep learning models*. Medium. https://towardsdatascience.com/everything-you-need-to-know-about-activation-functions-in-deep-learning-models-84ba9f82c253