**Homework 2**

1. **Section 1 – MNIST Dataset**
   1. **Network architecture and results**
      1. **Model**

Below is the final configuration used for the convolutional network model implemented in the experiments. The inputs of the model are the 28x28 pictures provided by the MNIST dataset.

|  |  |  |
| --- | --- | --- |
| **Layer type** | **Layer’s details** | **Output Shape** |
| Convolutional input layer | 4 filters  Filter size: 3x3  Stride: 1x1 | 26,26,4 |
| Max pooling layer | -- | 13,13,4 |
| Convolutional layer | 8 filters  Filter size: 3x3  Stride: 1x1 | 11,11,8 |
| Max pooling layer | -- | 5,5,8 |
| Convolutional layer | 16 filters  Filter size: 3x3  Stride: 1x1 | 3,3,16 |
| Flatten layer | -- | 144 |
| Fully connected layer | 64 neurons | 64 |
| Fully connected output layer | 10 neurons | 10 |

The training hyperparameters are presented in the table below:

|  |  |
| --- | --- |
| **Optimizer** | Adam |
| **Loss** | Sparse Categorical Cross Entropy Loss |
| **Epochs** | 5 |
| **Activation of layers** | ReLU |

* + 1. **Stride and filter sizes**
    2. **Learning curve plot**
    3. **Accuracy of training and test sets**
    4. **Distribution of weights and biases**
  1. **Examples**
     1. **Correctly classified examples**
     2. **Incorrectly classified examples**
     3. **Discussion**
  2. **Feature maps**
  3. **Regularization**

1. **Section 2 – CIFAR-10 Dataset**
   1. **Network architecture and results**
      1. **Model**
      2. **Stride and filter sizes**
      3. **Learning curve plot**
      4. **Accuracy of training and test sets**
      5. **Distribution of weights and biases**
   2. **Examples**
      1. **Correctly classified examples**
      2. **Incorrectly classified examples**
      3. **Discussion**
   3. **Feature maps**
   4. **Regularization**
   5. **Pre-processing of images**

**Main References**

* Andrew Ng, *Neural Networks and Deep Learning* [MOOC]. Coursera. <https://www.coursera.org/learn/neural-networks-deep-learning/>
* Andrew Ng. *Machine Learning* [MOOC]. <https://www.coursera.org/learn/machine-learning>
* Ian Goodfellow, Yoshua Bengio, & Aaron Courville (2016). Deep Learning. MIT Press.
* Peter Roelants, *How to implement a neural network (1/5) - gradient descent*, 2015. <https://peterroelants.github.io/posts/neural-network-implementation-part01/> (Accessed November 2021).
* Matt Mazur, *A Step by Step Backpropagation Example*, 2015. <https://mattmazur.com/2015/03/17/a-step-by-step-backpropagation-example/>. (Accessed November 2021).
* Piotr Skalski, *Let’s code a Neural Network in plain NumPy*, 2018. <https://towardsdatascience.com/lets-code-a-neural-network-in-plain-numpy-ae7e74410795>. (Accessed November 2021).
* Zhuochen Wu, *Feature Selection in Convolutional Neural Network with MNIST Handwritten Digits*, 2018.

<http://users.cecs.anu.edu.au/~Tom.Gedeon/conf/ABCs2018/paper/ABCs2018_paper_156.pdf> (Accessed November 2021).

Arseny Turin, Gradient Descent From Scratch, 2020. <https://towardsdatascience.com/gradient-descent-from-scratch-e8b75fa986cc> . (Accessed November 2021).