FIT3181 Deep Learning – S2 2023

Assignment 1 – Machine Learning, Deep NNs, and CNNs

Purpose	The theme of this assignment is practical machine learning knowledge and skills in deep neural networks, including feedforward and convolutional neural networks. In this assignment, you will demonstrate your knowledge of deep neural networks, CNNs, model robustness, and that you can build a simple image classification for a real-world application. The assignment relates to Unit Learning Outcomes 1, 2, 3, and 4.
Your task	Complete the individual tasks as detailed in the instructions below.
Value	**25%** of your total marks for the unit The assignment is marked out of 100 marks.
Due Date	**11:55 pm Friday 08 September 2023**
Submission	● Via Moodle Assignment Submission. ● Turnitin will be used for similarity checking of all submissions. ● This is an individual assignment (group work is not permitted). ● DRAFT submission is not assessed. ● In this assessment, you must **not** use generative artificial intelligence (AI) to generate any materials or content in relation to the assessment task.
Assessment Criteria	Marks are awarded for the *understanding* and *correctness* of your work, including but not limited to your code, calculations, and explanations to respond to required tasks. The instructions contain an individual marks breakdown for these components.
Late Penalties	**20%** deduction per calendar day
Support Resources	See Moodle Assessment page
Feedback	Feedback will be provided on student work via: ● general cohort performance ● specific student feedback ten working days post submission

INSTRUCTIONS

This assignment has three (3) parts:

- In part I, you will be assessed on theory and knowledge of machine learning and deep learning. More specifically, you will be tested on activation functions and how DNNs, forward propagation and backward propagation work.
- In part II, you will be asked to build, train, test and improve a basic DNN for handwritten letter recognition. You will need to load and prepare data, visualize data for inspection, build your DNN and tune model hyperparameters, and then further improve your DNN with label smoothing trick.
- In part III, you will be required to build a real-work image classifier with CNNs. Then, you will need to implement code to improve your model robustness against adversarial attacks.

Make sure you read the instructions carefully.

You need to **submit through the Moodle** Assignment activity a single ZIP file, name **xxx_A1_solution.zip**, where xxx is your student ID. The ZIP should contain:

1) Jupyter notebooks with answers to the questions and your work. They should be named **A1_Part1_Solutions.ipynb**, **A1_ Part2_Solutions.ipynb**, and **A1_ Part3_Solutions.ipynb** corresponding to part 1, part 2, and part 3 respectively.

2) A copy of your solution notebooks exported in HTML format.

3) (Optional) Any extra file or folder needed to complete your assignment (e.g., images used in your answers).

Part 1: Question on theory and knowledge (30 points)

The first part of this assignment is for you to demonstrate your knowledge in deep learning that you have acquired from the lectures and tutorials materials. Most of the contents in this assignment are drawn from the lectures and labs from weeks 1 to 2. Going through these materials before attempting this part is highly recommended.

In this part of the assignment, you will demonstrate your knowledge of activation functions, and how DNN, forward propagation and backward propagation work.

Question 1.1 Activation Functions (8 points)

Activation function plays an important role in modern Deep NNs. In this question, we will explore some of them to get a deeper understanding of their characteristics and their advantages.

a) Given the Exponential Linear Unit activation function:

$$\mathrm{ELU} = \begin{cases} \alpha(e^x - 1) & \text{if } x < 0 \\ \mathrm{x} & \text{otherwise} \end{cases}$$

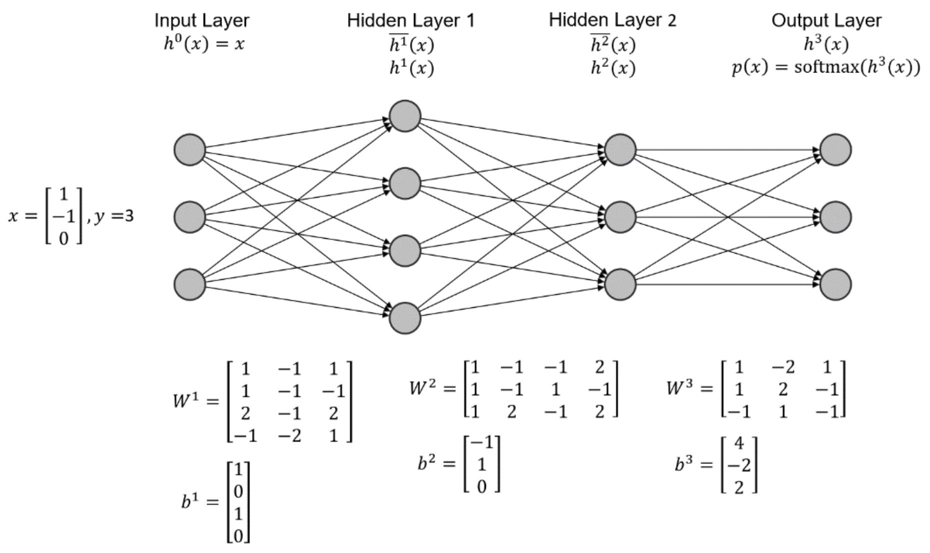
State its output range, find its derivative (show your steps), and plot the activation function and its derivative. (2 points)

b) In literature, there are a wide range of activation functions that have been proposed. Do a research and select two (2) activation functions that were not discussed in the lecture (including *ReLU*, *sigmoid*, and *tanh*). For each of the selected activation function, do the following (3 points total for each activate function):

- Find the research paper which proposes the activation function.
- Write a summary of the author's motivation which leads to the activation function (max 150 words).
- Write a summary of advantages of the activation function (max 150 words).

Question 1.2 Feed-Forward Neural Networks (8 points)

Assume that we feed a data point x with a ground-truth label $y = 3$ (with index starting from 1 as in the lecture) to the feed-forward neural network with the ReLU activation function at hidden layers as shown in the following figure:


Input Layer
$h^0(x) = x$
Hidden Layer 1
$\overline{h^1}(x)$
$h^1(x)$
Hidden Layer 2
$\overline{h^2}(x)$
$h^2(x)$
Output Layer
$h^3(x)$
$p(x) = \mathrm{softmax}(h^3(x))$
$x = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}, y = 3$
$W^1 = \begin{bmatrix} 1 & -1 & 1 \\ 1 & -1 & -1 \\ 2 & -1 & 2 \\ -1 & -2 & 1 \end{bmatrix}$
$b^1 = \begin{bmatrix} 1 \\ 0 \\ 1 \\ 0 \end{bmatrix}$
$W^2 = \begin{bmatrix} 1 & -1 & -1 & 2 \\ 1 & -1 & 1 & -1 \\ 1 & 2 & -1 & 2 \end{bmatrix}$
$b^2 = \begin{bmatrix} -1 \\ 1 \\ 0 \end{bmatrix}$
$W^3 = \begin{bmatrix} 1 & -2 & 1 \\ 1 & 2 & -1 \\ -1 & 1 & -1 \end{bmatrix}$
$b^3 = \begin{bmatrix} 4 \\ -2 \\ 2 \end{bmatrix}$

a) What is the numerical value of the latent presentation $h^1(x)$? (1 point)

b) What is the numerical value of the latent presentation $h^2(x)$? (1 point)

c) What is the numerical value of the logit $h^3(x)$? (1 point)

d) What is the corresponding prediction probability $p(x)$? (1 point)

e) What is the predicted class? Is it a correct or incorrect prediction? (1 point)

f) What is the cross-entropy loss caused by the feed-forward neural network at (x, y)? (1 point)

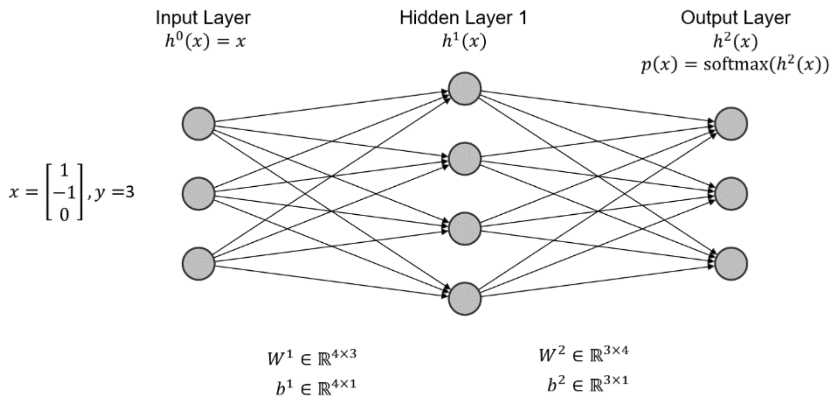
g) Assume that we are applying the label smoothing technique[1], with $\alpha = 0.1$. What is the relevant loss caused by the feed-forward neural network at (x, y)? (2 points)

Note: *You need to show both formulas and numerical results for earning full marks. Although it is optional, it is great if you show your **numpy** code for your computation.*

Question 1.3 Back propagation (10 points)

Assume that we are constructing a multilayered feed-forward neural network for a classification problem with three classes where the model parameters will be generated randomly.

The architecture is as follows:


Input Layer
$h^0(x) = x$
Hidden Layer 1
$h^1(x)$
Output Layer
$h^2(x)$
$p(x) = \text{softmax}(h^2(x))$
$x = \begin{bmatrix} 1 \\ -1 \\ 0 \end{bmatrix}, y = 3$
$W^1 \in \mathbb{R}^{4\times3}$
$b^1 \in \mathbb{R}^{4\times1}$
$W^2 \in \mathbb{R}^{3\times4}$
$b^2 \in \mathbb{R}^{3\times1}$

We then feed a feature vector $x = [1 \quad -1 \quad 0]$ with ground-truth label $y = 3$ to the above network.

a) Suppose that we use cross-entropy (CE) loss. What is the value of the CE loss l? (2 points)

b) What are the derivatives $\frac{\partial l}{\partial h^2}$, $\frac{\partial l}{\partial W^2}$, and $\frac{\partial l}{\partial b^2}$? (3 points)

c) What are the derivatives $\frac{\partial l}{\partial h^1}$, $\frac{\partial l}{\partial \bar{h}^1}$, $\frac{\partial l}{\partial W^1}$, and $\frac{\partial l}{\partial b^1}$? (3 points)

d) Assume that we use SGD with learning rate $\eta = 0.01$ to update the model parameters.

[1] Link for main paper from Goeff Hinton: https://papers.nips.cc/paper/2019/file/f1748d6b0fd9d439f71450117eba2725-Paper.pdf

What are the values of W^2, b^2 and W^1, b^1 after being updated? (2 points)

Note: *You need to show both formulas, numerical results, and your* ***numpy*** *code for earning full marks.*

Question 1.4 Optimization with Gradient Descent (4 points)

In this question, we will take a step further on getting deeper understanding about gradient descent, one of the most important optimization techniques in deep learning.

a) Write the pseudo-code to implement the gradient descent algorithm and explain in your words what each line of the code does (2 points)

b) Explain in your own words why we should update the parameters in the opposite direction of the gradient (2 points)

Part 2: Deep Neural Networks (30 points)

This part of the assignment is for you to demonstrate your basis knowledge in deep learning that you have acquired from the lectures and tutorials materials. Most of the contents in this assignment are drawn from the tutorials covered from weeks 1 to 4. Going through these materials before attempting this assignment is highly recommended.

You are going to work with the **EMNIST** dataset for *image recognition task*. This dataset can be installed with the command `pip install emnist`. It has the exact same format as MNIST (grayscale images of 28 × 28 pixels), but the images represent handwritten letters rather than handwritten digits, so the problem is more challenging than MNIST.

Question 2.1 Load the EMIST Datasets and Process Data (4 points)

In this question, we use functions in the package **emnist**, namely `extract_training_samples` and `extract_test_samples`, to load the training and testing sets. We also want to encode labels using an ordinal encoding scheme.

The shape of training and testing data are (num_train, 28, 28) and (num_test, 28, 28), where num_train and num_test are number of training and testing images respectively. We next convert them to arrays of vectors which have shapes (num_train, 784) and (num_test, 784).

Question 2.2 Split Data into Training, Validation, and Testing Datasets (2 points)

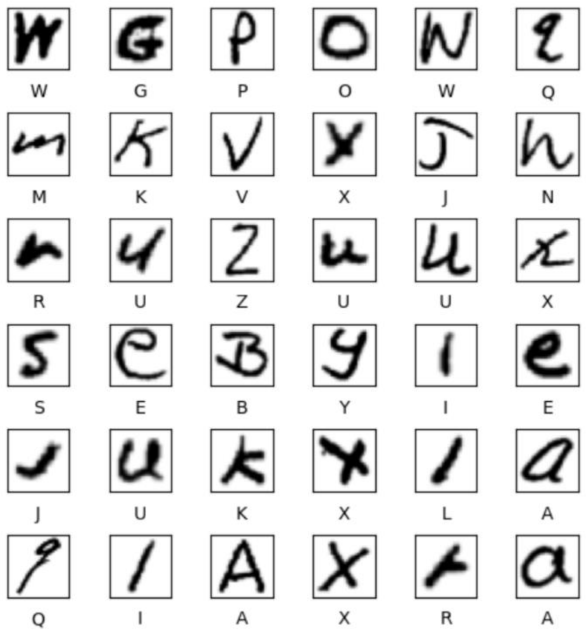
You need to write the code to address the following requirements:

- Use 10% of training data for validation and the rest of training data for training.
- Scale the pixels of training, validation and testing data to [0,1].

You have now the separate training, validation, and testing sets for training your model.

Question 2.3 Visualize Some Images in the Training Set with Labels (5 points)

You are required to write the code to randomly show 36 images in training data (which is an array of images) with labels as in the following figure.



Question 2.4 Write Code for the Feed-Forward Neural Net Using TF 2.x (5 points)

We now develop a feed-forward neural network with the architecture 784$\rightarrow$ 20(ReLU) $\rightarrow$ 40(ReLU) $\rightarrow$ 10($softmax$). You can choose your own way to implement your network and an optimizer of interest. You should train model in at least 20 epochs and evaluate the trained model on the test set.

Question 2.5 Tune Hyperparameters with Grid Search (6 points)

Assume that you need to tune the number of neurons on the first and second hidden layers $n_1 \in \{20, 40\}$, $n_2 \in \{20, 40\}$, and the used activation function $act \in \{sigmoid, tanh, relu\}$. The network has the architecture pattern 784 $\rightarrow$ n1(act) $\rightarrow$ n2(act) $\rightarrow$ 10($softmax$), where n1, n2, and act are in their grides. Write the code to tune the hyperparameters n1, n2, and act. Note that you can freely choose the optimizer and learning rate of interest for this task.

Question 2.6 Experimenting with the Label Smoothing Technique (8 points)

Implement the label smoothing technique by yourself. Note that you cannot use the built-in label-smoothing loss function in TF 2.x. Try the label smoothing technique with α = {0.1, 0.15, 0.2} and report the performances. You need to examine the label smoothing technique with the best architecture obtained in Question 2.5.

Part 3: Convolutional Neural Networks and Image Classification (40 points)

This part of the asssignment is designed to assess your knowledge and coding skill with Tensorflow as well as hands-on experience with training Convolutional Neural Network (CNN).

The dataset we use for this part is the STL10 dataset[2] which consists of 5,000 training images of airplane, bird, car, cat, deer, dog, horse, monkey, ship, truck; each of which has 500 images.

You are provided with a base code which downloads the dataset and train a default model.

Question 3.1 Observe the Learning Curve (4 points)

Run the provided cells to train the default model and observe the learning curve. Then, report your observation (i.e. did the model learn well? If not, what is the problem? What would you do to improve it?).

Note: for questions 3.2 to 3.9, you'll need to *write your own model* in a way that makes it easy for you to experiment with different architectures and parameters. The goal is to be able to pass the parameters to initialize a new instance of `YourModel` to build different network architectures with different parameters. You can investigate the code of the class `BaseImageClassifier` before writing your own model.

Question 3.2 Define Your CNN (4 points)

Write the code of the `YourModel` class. Note that this class will be inherited from the `BaseImageClassifier` class. You'll only need to re-write the code for the `build_cnn` method in the `YourModel` class.

Question 3.3 Experiment with Skip Connection (6 points)

Once writing your own model, you need to compare two cases: (i) using the skip connection, and (ii) not using the skip connection. You should set the instance variable `use_skip` to either True or False. For your runs, report which case is better and if you confront overfitting in training.

Question 3.4 Tune Hyperparameters with Grid Search (4 points)

Now, let us tune the $num\_blocks \in \{2, 3, 4\}$, $use\_skip \in \{True, False\}$, and $learning\_rate \in \{0.001, 0.0001\}$. Write your code for this tuning and report the result of the best model on the testing set. Note that you need to show your code for tuning and evaluating on the test set to earn the full marks. During tuning, you can set the instance variable verbose of your model to False for not showing the training details of each epoch.

Question 3.5 Apply Data Augmentation (4 points)

We now try to apply data augmentation to improve the performance. Extend the code of the class `YourModel` so that if the attribute `is_augmentation` is set to True, we apply the data augmentation. Also, you need to incorporate early stopping to your training process. Specifically, you early stop the training if the valid accuracy cannot increase in three consecutive epochs.

Hint that you can rewrite the code of the fit method to apply the data augmentation. In addition, you can copy the code of `build_cnn` method above to reuse here.

[2] https://cs.stanford.edu/~acoates/stl10/

Question 3.6 Observe Model Performance with Data Augmentation (4 points)

Leverage your best model with the data augmentation and try to observe the difference in performance between using data augmentation and not using it. Write a report of your observation.

Question 3.7 Explore Data Mixup Technique (4 points)

In this question you will explore Data Mixup Technique for improving generalization ability. Data mixup is another super-simple technique used to boost the generalization ability of deep learning models. You need to incorporate data mixup technique to the above deep learning model and experiment its performance. There are some papers and documents for data mixup which you can refer to:

- Main paper for data mixup: https://openreview.net/pdf?id=r1Ddp1-Rb
- An article on data mixup: https://www.inference.vc/mixup-data-dependent-data-augmentation/

You need to extend your model developed above, train a model using data mixup, and write your observations and comments about the result.

Question 3.8 Attack Your Model (5 points)

Attack your best obtained model with PGD, MIM, and FGSM attacks with $\epsilon = 0.0313$, $k = 20$, $\eta = 0.002$ on the testing set. Write the code for the attacks and report the robust accuracies. Also choose a random set of 20 clean images in the testing set and visualize the original and attacked images.

Question 3.9 Train a Robust Model (5 points)

Train a robust model using adversarial training with PGD ϵ=0.0313, k=10, η=0.002. Write the code for the adversarial training and report the robust accuracies. After finishing the training, you need to store your best robust model in the folder `./models` and load the model to evaluate the robust accuracies for PGD, MIM, and FGSM attacks with ϵ=0.0313, k=20, η=0.002 on the testing set.

Question 3.10 (bonus question) (5 points)

Read the SAM paper[3] and try to apply this technique to the best obtained model and report the results. For the purpose of implementing SAM, we can flexibly add more cells and extensions to the *model.py* file.

END OF ASSIGNMENT

GOOD LUCK WITH YOUR ASSIGNMENT 1!

[3] https://openreview.net/pdf?id=6Tm1mposlrM