Assignment 3
Neural Network

Jia He Lim
Po-Chih Kuo

Goal

- Build your own deep neural network step by step
- Implement all the functions required to build a deep neural network
- Understanding forward propagation, backward propagation and update
- Implement Binary Cross-Entropy loss (basic part) and Categorical Cross-Entropy loss (advanced part)
- Implement binary classifier (basic part) and multi-class classifier (advanced part)

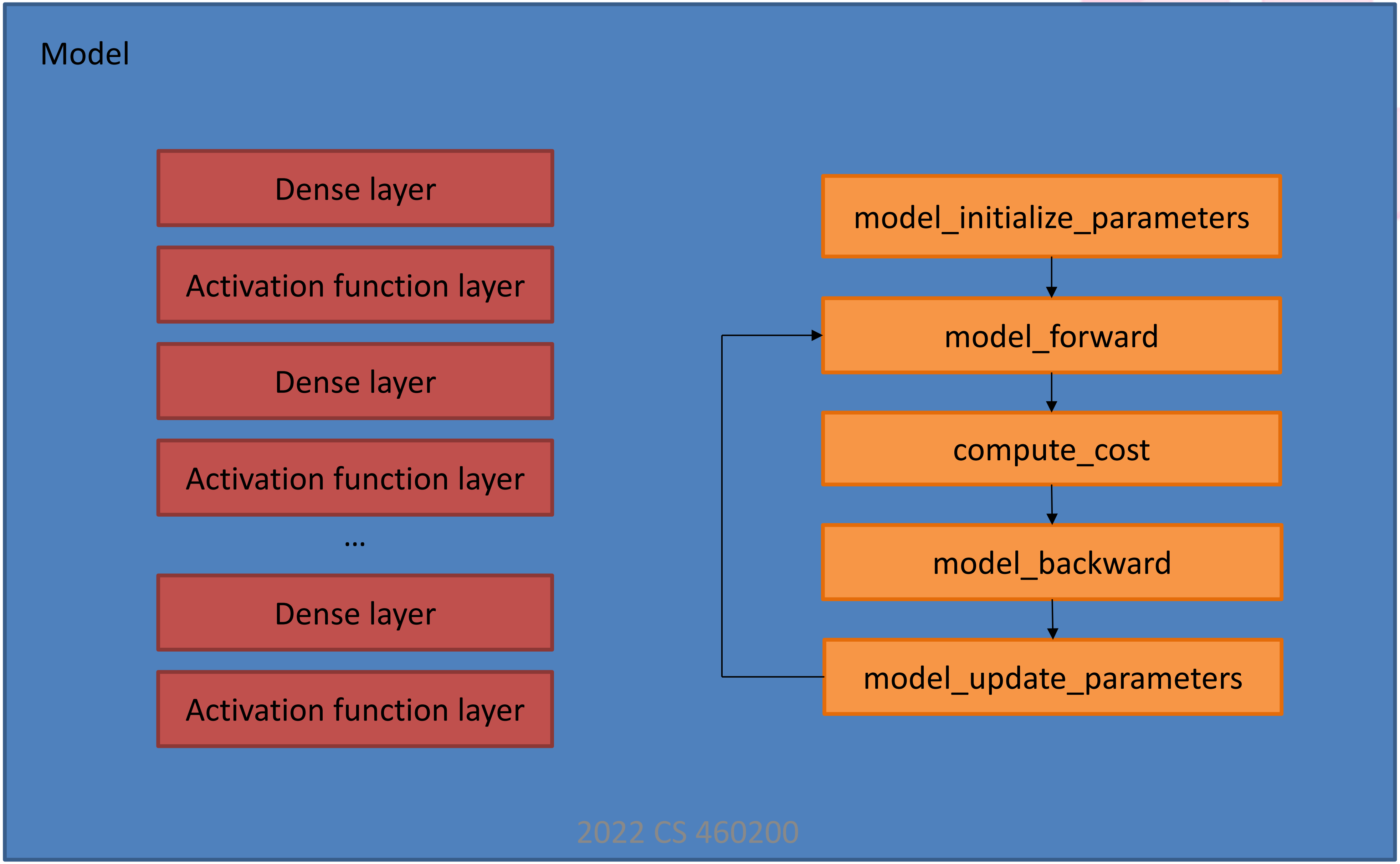
Grading Policy

Item	Score
Basic Implementation	65%
Advanced Implementation	30%
Basic & Advanced Report	5%

Overview

- Layer with parameters
  - Dense layer
    - initialize_parameters
    - forward
    - backward
    - update
- Layer without parameters
  - Activation function layer
    - forward
    - backward

Overview


Model
Dense layer
Activation function layer
Dense layer
Activation function layer
...
Dense layer
Activation function layer
model_initialize_parameters
model_forward
compute_cost
model_backward
model_update_parameters

Basic Implementation (65%)

Dense layer

1. Create and initialize parameters of a linear layer with Glorot uniform initialization. (5%)
2. Implement linear forward. (5%)
3. Implement linear backward. (5%)
4. Implement linear update parameters. (5%)

Activation function layer

1. Implement activation forward. (5%) (Sigmoid and ReLU)
2. Implement activation backward. (5%) (Sigmoid and ReLU)

Basic Implementation (65%)

Model

1. Implement model initialize parameters. (5%)
2. Implement model forward. (5%)
3. Implement model backward. (5%)
4. Implement model update parameters. (5%)
5. Compute the binary cross-entropy cost. (5%)
6. Implement a binary classifier and tune hyperparameter to get a good rank. (10%)

Advanced Implementation (30%)

Activation function layer

1. Implement activation forward. (5%) (Softmax)
2. Implement activation backward. (5%) (Softmax+CCE_loss)

Model

1. Compute the categorical cross-entropy cost. (5%)
2. Implement a multi-class classifier and tune hyperparameter to get a good rank. (15%)

Basic & Advanced Report (5%)

- Describe what problems you encountered and how did you solve them when implementing the basic and advanced functions. (2%)
- Briefly describe the structure of your binary and multi-class classifiers. (2%)
- Describe effort you put to improve your model (e.g., hyperparameter finetuning). (1%)
- Do not exceed 1 page!
- Name your report file as “**report.pdf**”.

Data

Binary classification: Breast Cancer Data Set

The data set consists of 569 samples (357 benign and 212 malignant). Ten real-valued features are computed for each cell nucleus: radius, texture, perimeter, area, smoothness, compactness, concavity, concave points, symmetry, and fractal dimension. The mean, standard error, and "worst" or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features.

The details of the training set and validation set are shown below:

- shape of X: (500, 30); shape of y: (500, 1)
- shape of X_train: (400, 30); shape of y_train: (400, 1)
- shape of X_val: (400, 30); shape of y_val: (100, 1)

Data

Multi-class classification: MNIST handwritten digit dataset

The MNIST dataset is an acronym that stands for the Modified National Institute of Standards and Technology dataset. It is a dataset of 70,000 small square 28×28 pixels grayscale images of handwritten single digits between 0 and 9.

The details of the training set and testing set:

- shape of X_train: (60000, 28, 28)
- shape of y_train: (60000,)
- shape of X_test: (10000, 28, 28)

Items for you

- Template: HW3_Neural_Network.ipynb (basic input data inside) (You are encouraged to use Colab!)
- Advanced data: advanced_data.npz
- Sample output: sample_output.npy

Template

Except for the imported packages in the template, you cannot use any other packages.
Remember to save the code file to **HW3_Neural_Network.ipynb**

1. Introduction

Welcome to your third assignment. In this assignment, you will build a deep neural network step by step. In this notebook, you will implement all the functions required to build a neural network.

After finishing this assignment, you will have a deeper understanding of the process of training a deep neural network, which only consists of three steps: forward propagation, backward propagation and update.

2. Packages

All the packages that you need to finish this assignment are listed below.

- numpy : the fundamental package for scientific computing with Python.
- matplotlib : a comprehensive library for creating static, animated, and interactive visualizations in Python.
- math : Python has a built-in module that you can use for mathematical tasks.
- sklearn.datasets : scikit-learn comes with a few small standard datasets that do not require to download any file from some external website. You will be using the breast cancer wisconsin dataset to build a binary classifier.

⚠️ **WARNING** ⚠️:

- Please do not import any other packages.
- np.random.seed(1) is used to keep all the random function calls consistent. It will help us grade your work. Please don't change the seed.

❗ **Important** ❗ : Please do not change the code outside this code bracket.

```
### START CODE HERE ### (≈ n lines of code)
...
### END CODE HERE ###
```

Output NPY File Format

- Named as “**output.npy**”
- This file is a dictionary that stores your output for each function and the binary and multi-class classification prediction. Note that the hyperparameter and weights of both classifiers will also be stored in the output file to check whether your predictions come from the same classifier you submitted.
- The dictionary should have the following keys: 'linear_initialize_parameters', 'linear_forward', 'linear_backward', 'linear_update_parameters', 'sigmoid', 'relu', 'softmax', 'sigmoid_backward', 'relu_backward', 'softmax_CCE_backward', 'model_initialize_parameters', 'model_forward_sigmoid', 'model_forward_relu', 'model_forward_softmax', 'model_backward_sigmoid', 'model_backward_relu', 'model_update_parameters', 'compute_BCE_cost', 'compute_CCE_cost', 'basic_pred_val', 'basic_layers_dims', 'basic_activation_fn', 'basic_model_parameters', 'advanced_pred_test', 'advanced_layers_dims', 'advanced_activation_fn', 'advanced_model_parameters'

Output NPY File Format

Expected output:

```
linear_initialize_parameters :  <class 'dict'>
linear_forward :  <class 'tuple'>
linear_backward :  <class 'tuple'>
linear_update_parameters :  <class 'dict'>
sigmoid :  <class 'tuple'>
relu :  <class 'tuple'>
softmax :  <class 'tuple'>
sigmoid_backward :  <class 'numpy.ndarray'>
relu_backward :  <class 'numpy.ndarray'>
softmax_CCE_backward :  <class 'numpy.ndarray'>
model_initialize_parameters :  <class 'tuple'>
model_forward_sigmoid :  <class 'tuple'>
model_forward_relu :  <class 'tuple'>
model_forward_softmax :  <class 'tuple'>
model_backward_sigmoid :  <class 'tuple'>
model_backward_relu :  <class 'tuple'>
model_update_parameters :  <class 'dict'>
compute_BCE_cost :  <class 'numpy.ndarray'>
compute_CCE_cost :  <class 'numpy.ndarray'>
basic_pred_val :  <class 'numpy.ndarray'>
basic_layers_dims :  <class 'list'>
basic_activation_fn :  <class 'list'>
basic_model_parameters :  <class 'list'>
advanced_pred_test :  <class 'numpy.ndarray'>
advanced_layers_dims :  <class 'list'>
advanced_activation_fn :  <class 'list'>
advanced_model_parameters :  <class 'list'>
```

Requirement

- Do it individually! Not as a team! (team is for final project)
- Announce date: 2022/11/10
- Deadline: 2022/11/23 23:59 (Late submission is not allowed!)
- Hand in your files in the following format (Do not compressed!)
  - HW3_Neural_Network.ipynb (Please keep your execution output)
  - output.npy
  - report.pdf
- Assignment 3 will be covered on the next exam.
- Note that Assignment 4 will be based on Assignment 3; it is recommended to put much effort into this assignment so you will have a good start for your next assignment.

The Evaluation Metric

In the basic and advanced function implementations, you will get a full score if your output is exactly the same as the standard answer, or else you will get a zero mark for each function implementation.

For the binary classifier and multi-class classifier, we will use **accuracy** to evaluate your classifier.

- Binary classification: You will be compared with others who submit the basic prediction.
- Multi-class classification: You will be compared with others who submit the advanced prediction.

Penalty

0 points if any of the following conditions:

- Plagiarism
- Late submission
- Not using template or import any other packages
- Incorrect input/output format
- Incorrect submission format
- Predictions mismatch (your predictions did not come from the same classifier you submitted)

Questions?

- TA: Jia He Lim (ivanljh123@gmail.com)
- No debugging service