# Lab 6
# Recurrent neural network

Yu-Chieh Lin

Po-Chih Kuo

# Goal

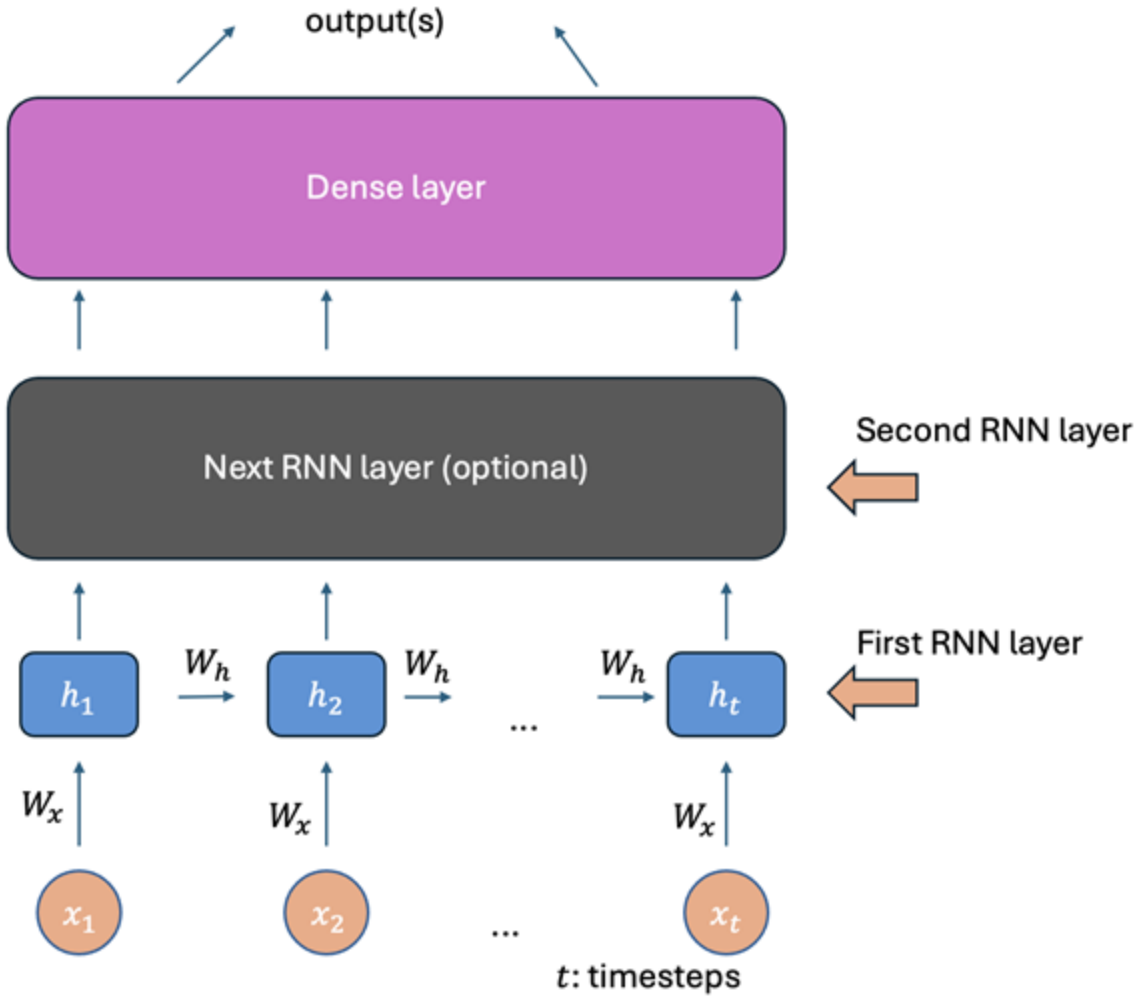
- Build your own recurrent neural network (RNN) by extending the previous Lab (Dense layer).
- Implement certain functions required to build a RNN.
- Understand how the forward propagation, backward propagation work in RNN.
- Build a model using RNN layer to (1)model a Sinusoidal wave and (2)classify human activity.

# *Grading Policy*

| Item | Score |
|---|---|
| Basic Part<br>RNN layer (30%)<br>Model class (10%)<br>Sinusoidal wave data (20%) | 60% |
| Advanced Part (human activity classification) | 35% |
| Report | 5% |

# Overview

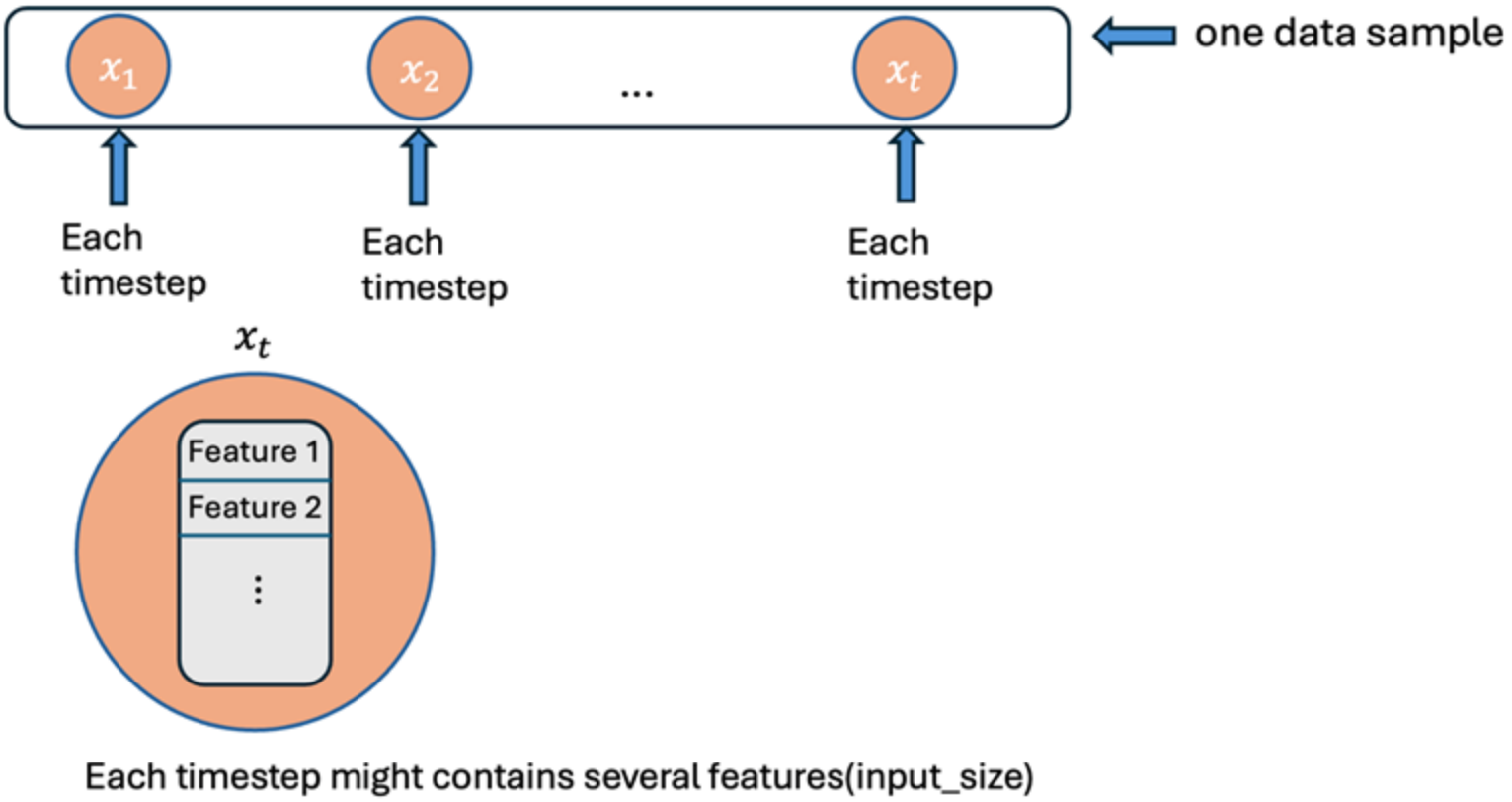
## RNN layer




$W_h, W_x, b_h$ would be initialized before first neuron ($h_1$)

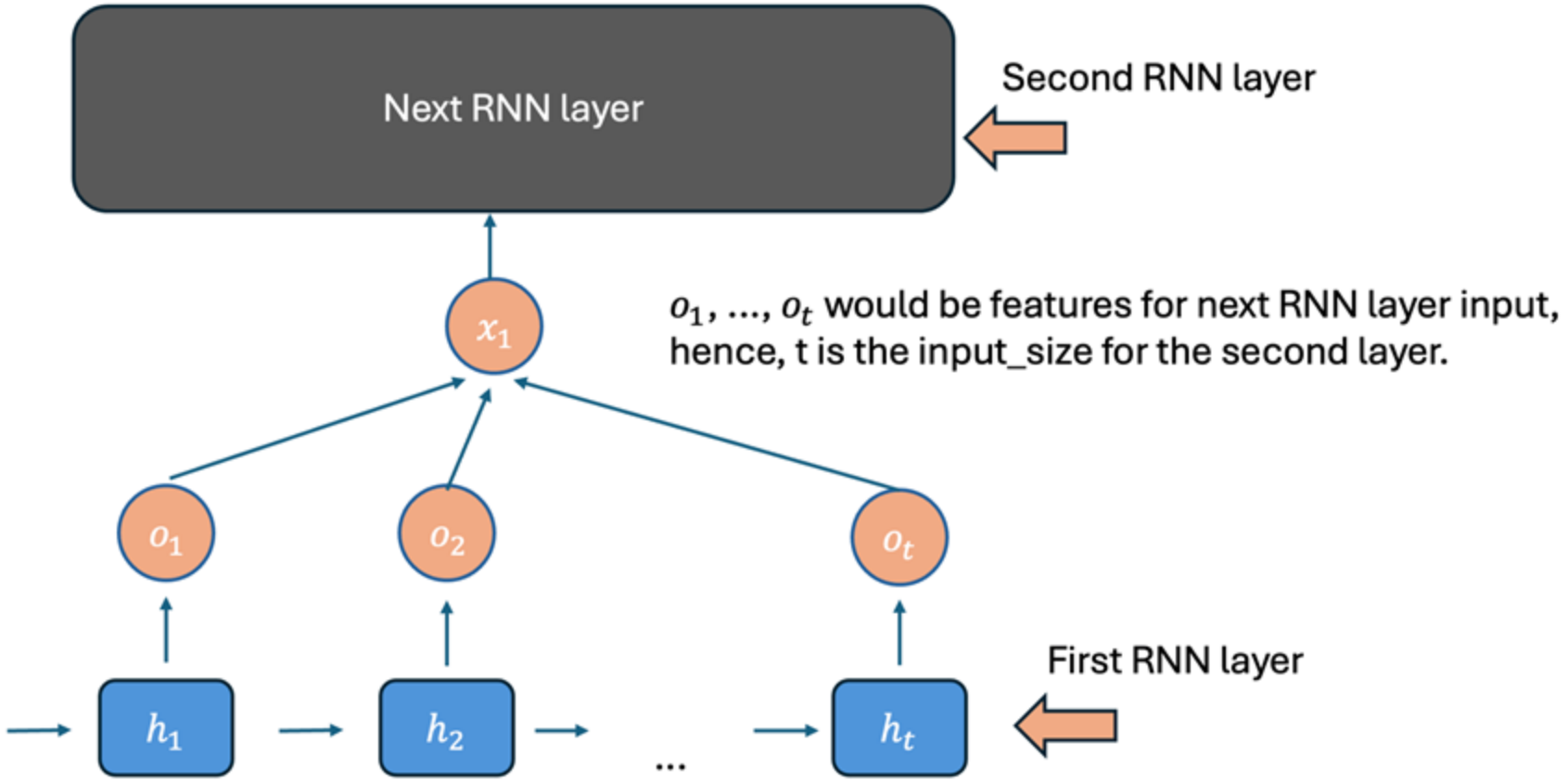
# Overview

## RNN layer (Cont'd)



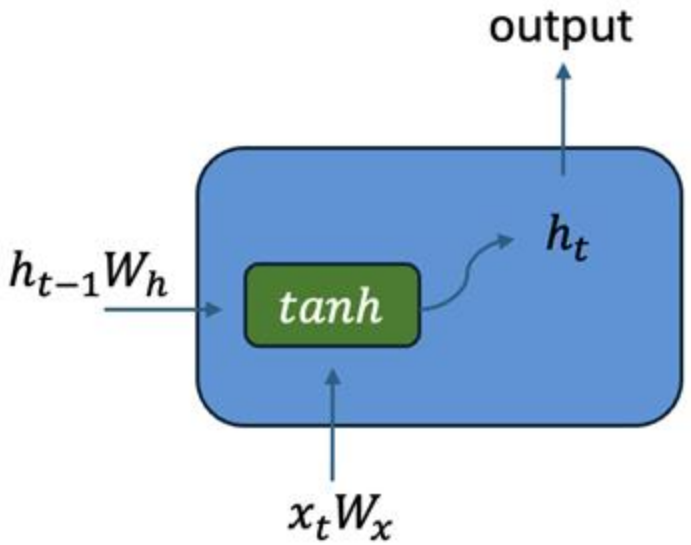
# Overview

## RNN layer connection



# Overview

**Each neuron in RNN**



# *Basic Part (60%)*

## RNN layer (30%)

1. Implement initialize_parameters function (3%)
2. Implement forward pass (12%)
3. Implement backward pass (15%)

Back propagation of RNN:

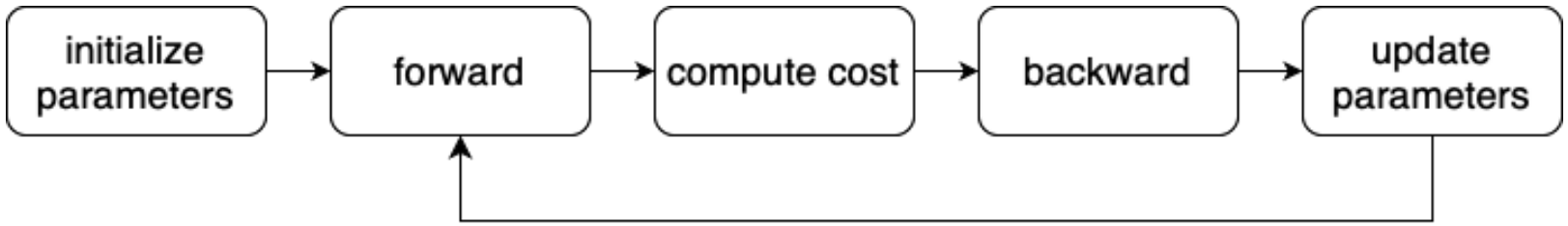
https://www.pycodemates.com/2023/08/backpropagation-through-time-explained-with-derivations.html

# *Basic Part (60%)*

## Model class (10%)

1. Implement forward pass, backward pass, update parameters and training function.

### Training process



# *Output .npy File Format*

- Named as “**Lab6_outputs.npy**”.

- This file is a dictionary that stores the output for each testing function. You can use the provided **sanity check and expected output** in the notebook to ensure nothing is missing.

# Output .npy File Format

- We will test your "**Lab6_outputs.npy**" to verify the correctness of your basic implementation.
- Submit this file to eeclass.

✓ 'RNN_initialization':
  - 'Wx_shape',
  - 'Wh_shape',
  - 'bh_shape',

✓ 'RNN_forward':
  - 'X shape',
  - 'X',
  - 'Output shape',
  - 'Output',

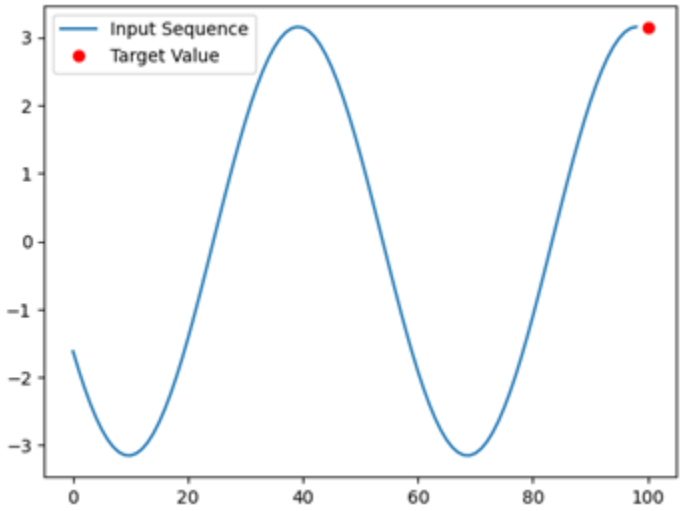
✓ 'RNN_backward':
  - 'dH shape',
  - 'dH',
  - 'dL_dX shape',
  - 'dL_dX',
  - 'dL_dWx shape',
  - 'dL_dWx',
  - 'dL_dWh shape',
  - 'dL_dWh',
  - 'dL_dbh shape'
  - 'dL_dbh'

✓ 'Model_class':
  - 'AL',
  - 'dA_prev',
  - 'Wx of RNN',
  - 'Wh of RNN',
  - 'bh of RNN'

# Basic Part (60%)

## Sinusoidal wave data (20%)

- In this dataset, we want you to build your model to predict the **last value** of given Sine wave data (sequential data).
- We provided a function *generate_sine_wave_data* that you can generate your own training data with custom frequency and amplitude.
- There are 500 samples with each 99 timesteps in testing data.



# *Basic Part (60%)*

## Sinusoidal wave data (20%)

1. You have to create your own training dataset by the given function in Lab6_template.ipynb.

2. First, try to construct the model without using RNN layer, that is, just using Dense layers to build the model. Then build another model with RNN layer to compare their performance.

3. Download the testing data(*X_test.csv*) **from Kaggle** to generate the prediction *y_pred_basic.csv.* Submit the *y_pred_basic.csv* to Kaggle for evaluation.

# *Basic Part (60%)*

## Sinusoidal wave data (20%)

**4. Kaggle link**: 2024ML-Lab6-Sinusoidal wave

5. We will use MAPE for evaluation. (On public leaderboard)

- Baseline 1: MAPE <= 17 (%) (10%)
- Baseline 2: MAPE <= 14 (%) (10%)

6. There is only public score (no private testing data) on Kaggle in this part!

# Output .csv file format (Basic part)

- Here are some output format:
  - First row is the header [Id, answer].
  - Your prediction answer should be value in *y_pred_basic.csv*.
  - There should be 500 rows output.
- Don't change the output cell in template so that the format would be correct!

- Submit the answer (*y_pred_basic.csv*) to Kaggle .

| Id | answer |
|---|---|
| 1 | -0.410314790479764 0 |
| 2 | 3.025491919814720 0 |
| 3 | -1.794977121385140 0 |
| 4 | 2.243559143572010 0 |
| 5 | -1.806936093652920 0 |
| 6 | -0.930316079378552 0 |
| 7 | -0.518172502708249 0 |
| 8 | 2.358781389023390 0 |
| 9 | -0.637970754423380 0 |
| 10 | 1.335621721987490 0 |

y_pred_basic.csv

# *Advanced Part (35%)*

## Designing a RNN Model for Categorical Classification

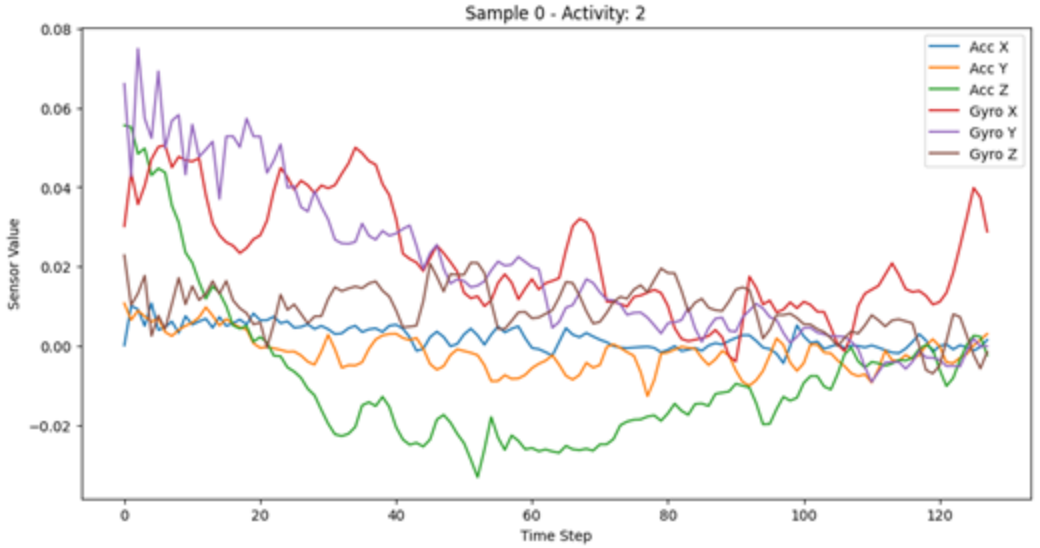
1. Design your RNN model with the given sequential data of Accelerometer and Gyroscope and predict the user's activity type. (There are 3 kinds of activity in this task)
2. You can only use the functions you implement in the basic part.
3. We will use **accuracy** to evaluate your model.
   - Baseline 1: Accuracy >= 0.65 (10%)
   - Baseline 2: Accuracy >= 0.7 (10%)
   - Baseline 3: Accuracy >= 0.75 (10%)
   - Ranking (5%)
4. We've splitted the testing data to 60% for public score & 40% for private score.

# *Data in Advanced Part*

## Categorical classification: Accelerometer and Gyroscope data

1. You will receive time-series (sequential) data of **Accelerometer and Gyroscope** each has 3-axial(x, y, z) data. For each record, there will be 128 timesteps.
2. Please download the training(X_train, y_train) and X_test from Kaggle.

   X_train would contains 7352 samples.



# *Data in Advanced Part*

**Categorical classification: Accelerometer and Gyroscope data**

3. Use the training data to classify 3 Human activities.

4. Submit the *y_pred_advanced.csv* to Kaggle, link: ML2024-Lab6-Activity classification

5. For *y_train.npy*, you should one-hot encode the labels before training the model

# *Output .csv file format (Advanced Part)*

- Here are some output format:
  - First row is the header [Id, Classes].
  - Your prediction answer should be class 0, 1, 2 in *y_pred_advanced.csv*
  - There should be 2947 row outputs.
- Don't change the output cell in template so that the format would be correct!
- Submit the answer (*y_pred_advanced.csv*) to Kaggle .

| Id | Classes |
|---|---|
| 0 | 2 |
| 1 | 2 |
| 2 | 2 |
| 3 | 2 |
| 4 | 2 |
| 5 | 2 |
| 6 | 2 |
| 7 | 2 |
| 8 | 2 |
| 9 | 2 |
| 10 | 2 |

y_pred_advanced.csv

# *Items for you*

- Template: **Lab6_template.ipynb**
- Some files: **Dense.py, Activation.py, Loss.py & Flatten.py (**You can paste the code you wrote in Lab4 & Lab5 into these files).

# *Template*

Except for the imported packages in the template, you cannot use any other packages.

Remember to save the code file to **Lab6.ipynb**

⚠ **WARNING** ⚠:

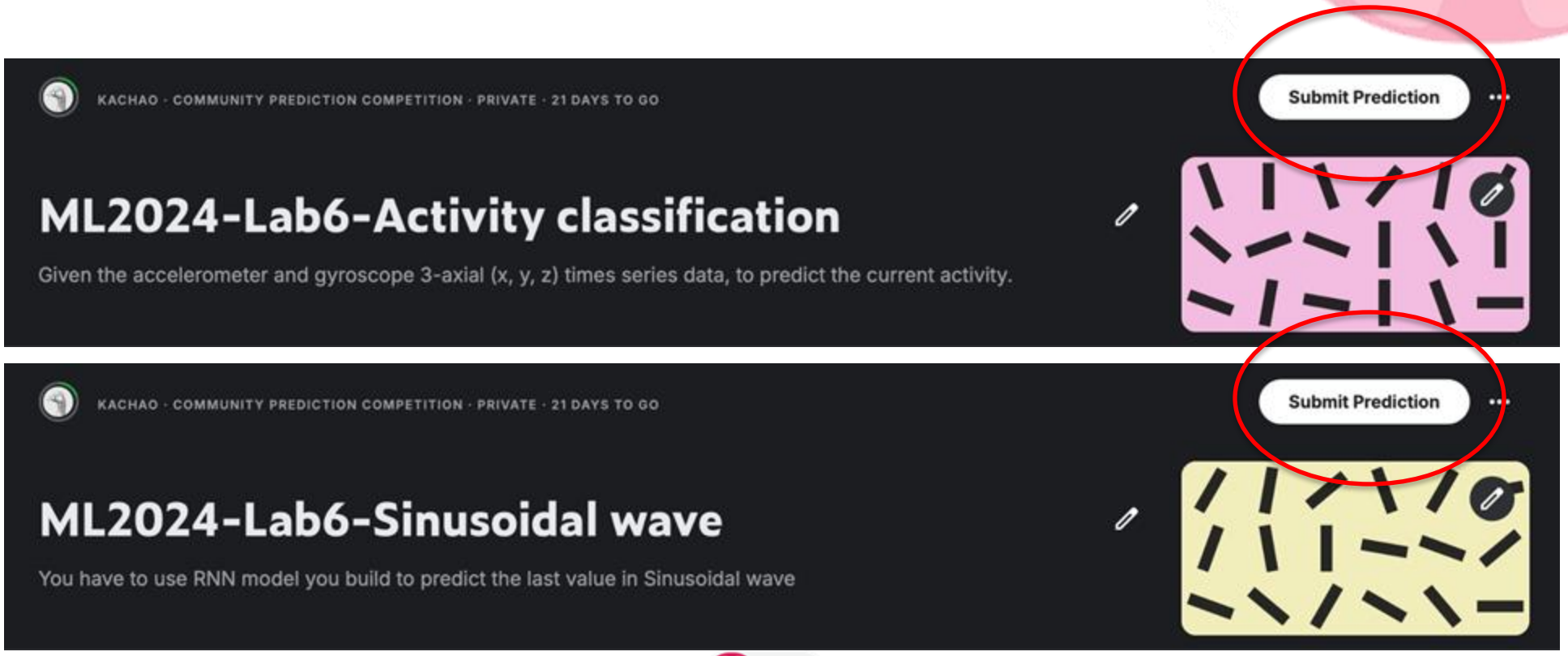
- Please do not import any other packages.
- np.random.seed(seed) is used to keep all the random function calls consistent. It will help us grade your work. Please don't change the seed.

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

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

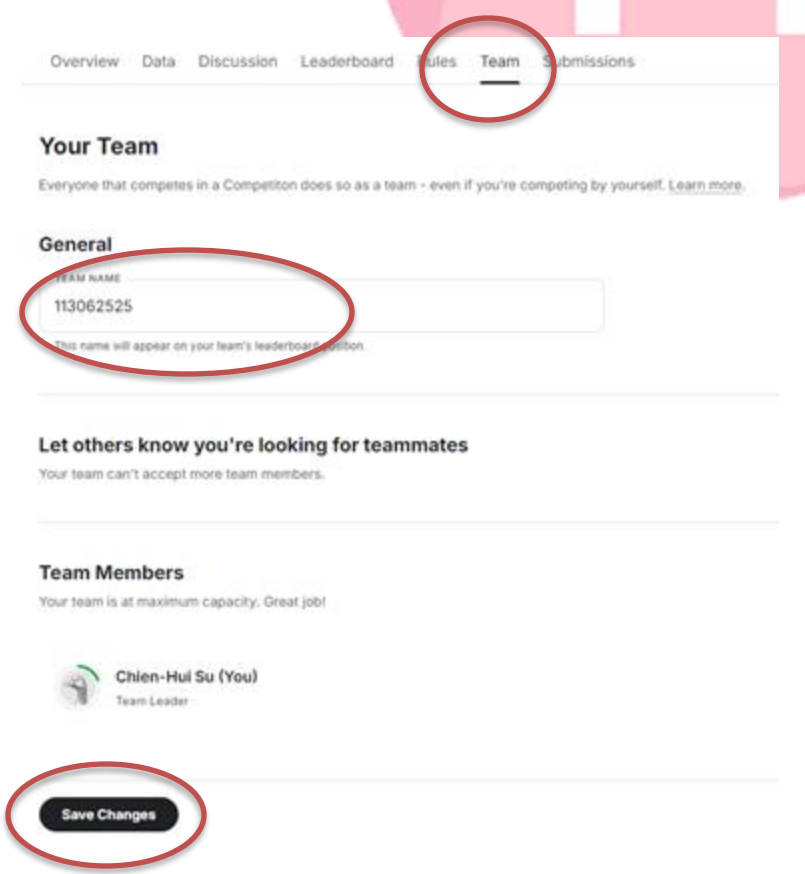
# *Kaggle*

- Please register your account.
- Click the ‘Join competition’ button to join.



# Kaggle

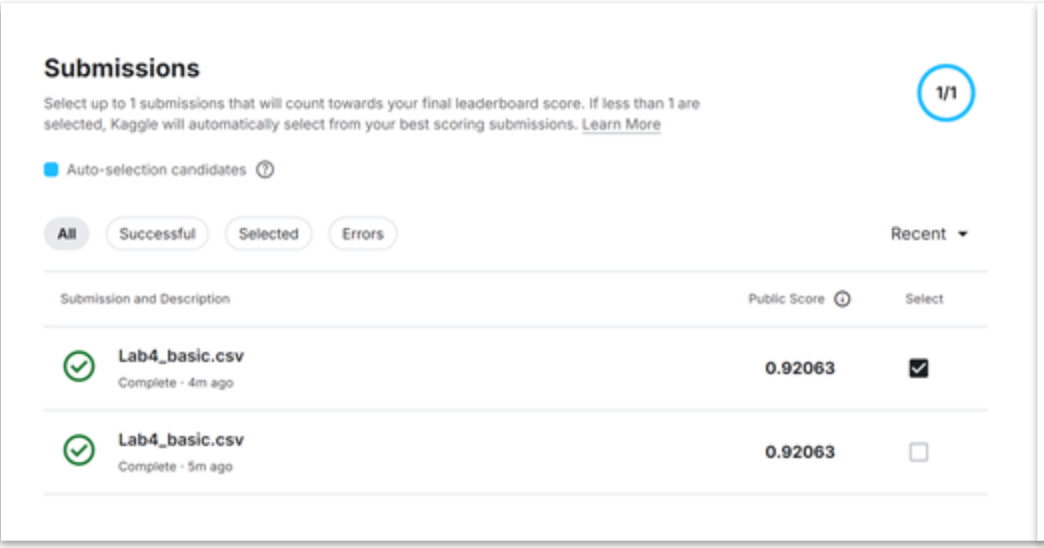
- After joining the competition, you should change your team's name (each student is a team) to your **student ID**.
- **Please remember to SAVE CHANGES**
- You can submit 100 times per day.



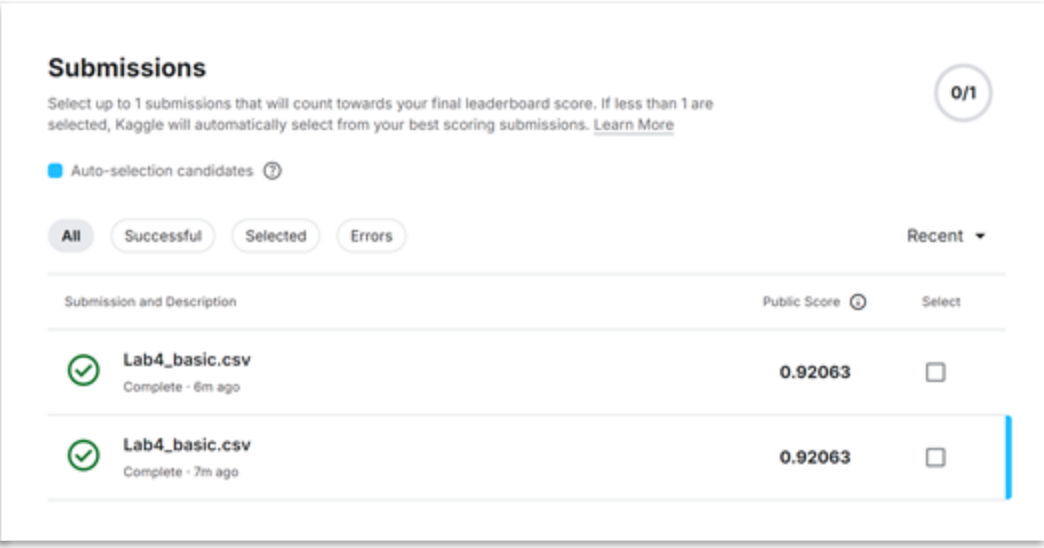

**Notes:** Please verify your team's name on the leaderboard - changing profile name does not change team name!

# Kaggle

You can manually select up to 1 submission that will count towards your final leaderboard score. If no submission is selected, Kaggle will automatically select your submission with the best public score.




Manual-selection




Auto-selection

# Report

1. When predicting values using sine wave data, is there a performance difference between the model that only contains Dense layers and one that includes an RNN layer? Which performs better? (1%)
2. Have you tried stacking two consecutive RNN layers in the model? How would you configure the parameters for the second RNN layer if the first RNN layer is defined as RNN(1, 16)? Briefly explain your reasoning.(2%)
3. What would be the effects with the larger size of hidden units in RNN layer? (2%)

Notes:

1. Do not exceed 1 page!
2. Name your report file as "**Lab6_report.pdf**".

# Requirement

- Do it individually! Not as a team! (team is for final project)
- Announce date: 2024/11/28
- Deadline: 2024/12/19 23:59 (Late submission is not allowed!)
- Submit the answers (.csv) to corresponding Kaggle competition (Ensure that your leaderboard **team name** matches your **student ID**)
  - Basic (Sinewave): ML2024-Lab6-Sinusoidal wave
  - Advance (Activity prediction): 2024ML-Lab6-Activity classification
- Hand in your files in the following format (Do not zip the files!)
  - Lab6.ipynb (Please keep your execution output)
  - Lab6_outputs.npy
  - Lab6_report.pdf

# Penalty

0 points if any of the following conditions happened:

- Plagiarism
- Late submission
- Not using a template or importing any other packages
- Incorrect input/output format
- No submission record on Kaggle
- Wrong team name on Kaggle (the team should be your **student ID**)
- No code("**Lab6.ipynb**") submission on eeclass
- Your submission on Kaggle was not generated by your code

5 Points would be deducted if your submission format or file name is incorrect

0 Points will be given in the Basic implementation if you don't submit "**Lab6_outputs.npy**"

# *Questions?*

- TA:
  - Yu-Chieh Lin (ss113062537@gapp.nthu.edu.tw)
- TA time: Thursday 17:00~18:00 at EECS 638
- TA time reservation sheet: TA reservation
- **No debugging service**