CITS5017 Deep Learning
Semester 2, 2023

Project 2

Assessed, worth 20%. Due: 11:59pm, Friday, 20th October 2023

1 Outline

In this project you will deal with a small pedestrian trajectory dataset extracted from the ETH/UCY dataset that had been widely used in pedestrian trajectory forecasting. The pedestrian trajectories were annotated as 2D coordinates extracted from the videos captured by fixed overhead cameras. The particular scene that you will work on for the project is *Zara* (source: https://github.com/crowdbotp/OpenTraj) as shown in Figure 1.

Figure 1: A frame sampled from a video of the Zara scene.

To make it easier for you, all the trajectories in the dataset have been normalised so that both the x- and y-coordinates are within the $[-1, +1]$ range. So, you should not need to perform any additional normalisation. Furthermore, all the trajectories are 20 frames long (i.e., 20 timesteps).

Your tasks for this project are to train two LSTM models to predict the trajectories of pedestrians and to train a variational autoencoder to synthesize new trajectories. This project is a good practical exercise to test your understanding of the techniques covered in Chapters 15, 16, and 17.

2 Submission

Name your Jupyter-notebook file as **Surname_FirstName-proj2.ipynb** (please replace **Surname** and **FirstName**[1] by your own surname and first name).

NOTE 1: You must submit your Jupyter-notebook file to **cssubmit** (https://secure.csse.uwa.edu.au/run/cssubmit). You will need to login using your Pheme account to do the submission. You should reserve

[1]You can include your middle name if you like. It does not matter. We just want to be able to distinguish students' files when we unzip the whole class's submissions together in one directory. **NOTE:** If your surname and/or given name contain special characters, such as apostrophes, '/', etc, you may have to omit them or replace them by underscores or hyphens as these characters have special meanings in Python: apostrophes are considered as single quotes; '/' is used to denote directories.

at least half an hour for the submission procedure as cssubmit carries out virus checking (which can't be disabled) on each submitted file. You need to ensure that your internet connection is stable during the entire uploading process. If possible, have your computer wire-connected to your modem before you start your submission.

You should wait until you see an acknowledgement screen confirming that your submission has completed successfully. You can take a screen shot and keep the image as evidence of your submission. Do **not** submit your Jupyter-notebook file via email to the Unit Coordinator. Do **not** upload it onto Google Drive or OneDrive and then send the link to the Unit Coordinator. **Only files submitted to cssubmit will be marked.**

NOTE 2: Do not blindly copy examples that you found on the internet. Many of these examples are too complex and do not match the project specification. You should be able to complete this project by following examples in the lectures, textbook, and sample codes from the author Géron.

3 Data Download and Preparation

The dataset for the project has been preprocessed as a numpy file, named **zara1.npy**, which can be downloaded from the LMS page of the unit.

Use the following function to help you retrieve the training set and validation set:

```
def load_data(filename):
    with open(filename, "rb") as f:
        train = np.load(f)
        val = np.load(f)
    return train, val

train_set, val_set = load_data("zara1.npy")
```

You should find that `train_set` is a 3D tensor of the shape $(26507, 20, 2)$ and `val_set`, also a 3D tensor, of the shape $(2315, 20, 2)$. This means that there are 26507 trajectories available for training and 2315 trajectories available for validation. All the trajectories have been trimmed to be 20 frames long. The last axis denotes that the datum for each trajectory in each frame is a $2D$ point (x, y).

4 Some background information about the loss function

In pedestrian trajectory prediction, our aim is to train a neural network that learns the trajectories of pedestrians over a number of frames so that it can predict the trajectories in future frames. For instance, given the 2D coordinates of N trajectories over 12 frames, we are interested in predicting the 2D coordinates of these trajectories in the next 8 frames. Let $\{(\bar{x}_{k,i}, \bar{y}_{k,i}) \,|\, \text{for } i = 1, \cdots, M; k = 1, \cdots, N\}$ and $\{(\hat{x}_{k,i}, \hat{y}_{k,i}) \,|\, \text{for } i = 1, \cdots, M; k = 1, \cdots, N\}$ be, respectively, the ground truth and predicted 2D coordinates of the N trajectories in the M future frames (in the example here, $M = 8$). The loss function $\mathscr{L}_{\text{ADE}}$ that the network should minimize is:

$$\mathscr{L}_{\text{ADE}} = \frac{1}{MN} \sum_{k=1}^{N} \sum_{i=1}^{M} \sqrt{(\bar{x}_{k,i} - \hat{x}_{k,i})^2 + (\bar{y}_{k,i} - \hat{y}_{k,i})^2}. \tag{1}$$

In the literature, this is called the **average displacement error** (or **ADE** for short). This is, in fact, just the average Euclidean distance of the ground truth and predicted points of all the N trajectories over the M future frames.

Another evaluation measure commonly used in the literature is the **final displacement error** (of **FDE** for short), defined as follows:

$$\mathscr{L}_{\text{FDE}} = \frac{1}{N}\sum_{k=1}^{N}\sqrt{(\bar{x}_{k,M} - \hat{x}_{k,M})^2 + (\bar{y}_{k,M} - \hat{y}_{k,M})^2}. \tag{2}$$

Here, only the points in the final frame of the N trajectories are used in the calculation.

5 Tasks

(i) Split the trajectories into two parts: the *observed* part should contain the trajectory coordinates for the first 12 frames; the *future* part should contain the trajectory coordinates for the remaining 8 frames.

(ii) Write a small function that takes in appropriate arguments so that it would randomly sample 30 trajectories and display them in 3 subplots (10 trajectories per subplot, to avoid crowdedness). The *observed* part and the *future* part of each trajectory should be shown in different colours.

Call the function twice – for the training set then the validation set.

(iii) **Implementation of the first LSTM model (30% (the mark includes parts (i) and (ii) above))**

- Design an LSTM that has 1 to 2 layers and an appropriate number of neurons per layer. This LSTM should be trained to take in the *observed* part of the trajectories and predict the pedestrians' 2D coordinates for just the **first** frame in the *future* part (i.e., predict one timestep ahead). Show the architecture of your LSTM network.
- You will need to write a function using the formula given in Equation (1) and use it as the loss function for compiling and training your network, e.g., `def ADE(y_true, y_pred)`.
- Train your network for 100 epochs but use an early stopping callback and the validation set to prevent over-training.
- Show the learning curves plot.
- As this LSTM network can predict only the pedestrians' coordinates for one timestep ahead, use a *for* loop to recursively predict the remaining 7 frames in the *future* part of the trajectories.
- Report the ADE and FDE of this network.
- Illustrate 30 randomly sampled trajectories in some subplots. The *observed* part, the ground truth *future* part and the predicted *future* part of these trajectories should be shown in different colours.

  If you have the function for part (ii) above designed well, you should be able to reuse the function here.

(iv) **Implementation of the second LSTM model (30%)**

Repeat the same steps above (the seven bullet points under the first LSTM model) for a second LSTM, which should follow the *encoder-decoder* network architecture.

(v) **Comparison (20%)**

- Compare the two LSTM models in terms of their numbers of parameters and performances on the ADE and the FDE.

- Change the lengths of the *observed* and *future* parts to $10/10$, $8/12$, and $6/14$[2] and compare how much the performances of both models drop with fewer *observed* timesteps and more *future* timesteps. Note that you should not need to change the architectures of the two LSTM networks, but you would need to recompile and retrain them. You can put the code in a function so that it can be reused for different split ratios of the trajectories.

(vi) **Trajectory generation** (20%)

Implement a variational autoencoder (VAE) that can generate trajectories of 20 frames long. Here, you don't need to split the trajectories into *observed* and *future* parts. Use the training set to train your VAE model and the validation set for validation. You loss function should include the reconstruction loss ($\mathscr{L}_{\text{ADE}}$ in Equation (1)) and the latent loss. Show the architecture of your VAE network. You can try training the network for 100 epochs with early stopping or try other settings. Show the learning curves plot.

Illustrate 30 generated trajectories from your model in subplots. Comment on the quality of the generated trajectories.

6 Markdown cells

Your Jupyter-notebook file should include markdown cells that explain the steps you carry out for various parts of the project. Some brief comments can (should) be inserted within the code cells (but try not to over-comment your code). Try to organize the tasks in different sections (with appropriate section headings) and number the sections to improve the overall presentation of your project.

7 Google Colab

The training of your models for this project should take less time than that for project 1. It should be very easy to complete your project in Google Colab also. Upon completing the project, you will need to download your Notebook file and submit it to **cssubmit**.

By default, GPU is not available on Colab. To specify that you need a GPU, select the menu item *Runtime* and then the option *Change runtime type*. In the popped-up window, select "GPU" for *Hardware accelerator*.

8 Penalty on late submissions

See the URL below about late submission of assignments:

https://ipoint.uwa.edu.au/app/answers/detail/a_id/2711/~/consequences-for-late-assignment-submission

Late submissions will be calculated as 5% of the total mark per day for the first 7 days (including weekends and public holidays) after which the project is not accepted.

[2] $10/10$ means "observe 10 timesteps and predict 10 timesteps"; $8/12$ means "observe 8 timesteps and predict 12 timesteps"; and so on.

9 Plagiarism

You should attempt the project by yourself. Collusion with (an)other student(s) is considered to be serious academic misconduct and can cause you to be suspended or expelled from the unit. Please see https://www.uwa.edu.au/students/my-course/student-conduct for more details.