CS6487 2019B - Assignment 2

Deep Learning - Social Checkin Prediction

Antoni Chan
Department of Computer Science
City University of Hong Kong

In this assignment you will predict check-in locations using Deep Learning.

Social Checkin Prediction

The goal is to predict the next social check-in (location and time) from the previous 3 checkins. Each checkin has 3 values: x- and y-coordinates, and the time-stamp (HH:MM:SS). Hence, the problem is a regression problem from 9-dimensional inputs (3 values x 3 previous checkins) to 3-dimensional outputs (3 values of the next check-in). For this sequence of 4 check-ins, you can assume that each checkin is within 24 hours of the previous one.

Dataset

The dataset is provided in `social-checkin-prediction.zip`. There are 3 splits of the dataset: training data (`train.csv`), validation data (`validation.csv`), and test data (`test.csv`). The training data is used for training the network, the validation data is used for checking convergence (e.g., for early stopping), and the test data should be held out and not involved in training. To prevent overfitting to each user, the data is split by user, so each data split has a distinct set of users. The data is provided as a CSV file, where each row contains: 1) user ID; 2-4) 1st check-in x, y, and time; 5-7) 2nd check-in; 8-10) 3rd check-in; 11-13) 4th check-in. The goal is to predict the 4th check-in from the 1st, 2nd, and 3rd check-ins. The check-in coordinates (x,y) are the pixel locations on the LA map (`map.png`). The time is in the format `HH:MM:SS` using the 24-hour clock. Probably you should convert this into a real number, similar to Assignment 1. The user ID is just an identifier number, and you do not to use it in your regression model.

Part 1 Baseline regression model

In the first problem, you will develop a baseline neural network model.

(a) Design a regression neural network (MLP) to predict the next check-in location and time from the previous 3 check-ins. Since the output values are real-numbers, you could use a linear activation for the output nodes, and mean-squared-error (MSE) as the loss function. However, note that the time is a circular value, and hence linear activation and MSE loss may not be the best choice for learning to predict time (e.g., 00:00:01 looks far from 23:59:59 if you directly subtract the values, but they are only 1 second apart if you consider them as times). Design a new activation and/or loss function for the time-component of the regression problem.

(b) Implement your network using a Deep Learning toolbox. You may use any toolbox that you like – high-level toolboxes, like Keras, are easy to use since they abstract the layers into different class objects that are easily concatenated together to form the network.

(c) Train and test your network using the provided training/validation and test sets. Compare your designed activation/loss function with the standard linear activation and MSE loss. Does it help improve the prediction?

.....

Part 2 Deep learning

In the second problem, you will add deep learning components to your model to improve the prediction accuracy.

(a) To improve the accuracy, test various types of regularization, e.g., L2, L1, drop-out, ensembling, early-stopping, data augmentation, etc. Which types of regularization can yield improvements in the test error? Why?

(b) To improve the accuracy, add various types of architecture components to your baseline model, e.g., batch norm, ResNet, DenseNet, attention, etc. Which types of architectures can yield improvements the test error? Why?

.....

- **What to hand in** – You need to turn in the following things:

  1. Answers, plots, analysis, discussion, etc. for the above problems.
  2. Source code files.

  You must submit your Assignment using the Canvas website.

- **Plagiarism** – You are allowed to use any deep learning toolbox for this assignment. However, you should design and code the network by yourself.

- **Grading** – The marks for this assignment will be distributed as follows:

  – 15% – Derivation and implementation of a new activation node and/or loss function (1a).
  – 15% – Design and implementation of the baseline model (1a, 1b).
  – 20% – Training and testing of the baseline model (1c).
  – 20% – Testing various regularization methods (2a).
  – 20% – Testing various architecture components (2b).
  – 10% – Quality of the written report. More points given to insightful observations and analysis.