Practical #3: Predicting Ethernet network configuration correctness using feedforward Neural Networks

Version: 10/12/2019

Instructors: Nicolas Navet (nicolas.navet@uni.lu), Long Mai (long.mai@uni.lu)

Technological context

Ethernet networks are starting to be used for in-vehicle communications. The main reason is the important bandwidth offered by Ethernet, which is needed to implement modern automotive functions such as automated/assisted driving and infotainment services.

In the early stage of the design of a car, the network architect defines the architecture of the network (e.g., topology and link speeds). The question that we want to answer is to predict whether a given set of flows (making up the traffic exchanged between the Electronic Control Units) will meet their performance constraints on the candidate architecture. Performance constraints can be for instance deadlines ("the engine speed value must be received within 5ms after its production by the suspension") or throughput ("at least 7Mbps must be available for software update"). If all performance constraints of the flows are met, the network configuration (i.e., the architecture plus the flows) is said *feasible*. In the following, we will consider the network topology shown in Figure 1.

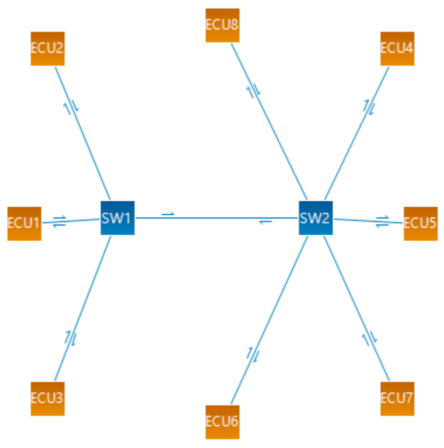

Figure 1: Topology of the network. There are 9 links, all operating at 100Mbps except the inter-switch link whose data rate is 1Gpbs.

There are algorithms to know whether a given set of flows will be feasible on a given topology. Those algorithms are however very compute intensive and cannot be used in design-space exploration when many design options (i.e., candidate solutions) have to be considered. Using classification, we would like to avoid running these algorithms and predict whether a network configuration will be feasible or not based on a training set (8000 configurations, and 1996 for the testing set).

Data set and template

The content of the dataset, in file “data.csv” available on Moodle, is as follows:

- Column 1: number of critical streams (e.g. dynamic of the vehicle, braking, suspension, etc)
- Column 2: number of audio streams
- Column 3: number of video streams
- Column 3: number of best-effort streams (code upload, etc)
- Column 5: maximum network load on any of the network links
- Column 6: Gini index (how well the load is distributed over the links)

The labels of the samples are indicated in file “labels.csv”, the first column of row *k* indicates whether the *k*-th configuration in the data.csv file is feasible or not.

Nb:

- there is a header in “data.csv” but not in “labels.csv”,
- the second column in “labels.csv” can be ignored (it is just the opposite of the first column).

Out of the 9996 data samples, the 8000 first samples should be reserved for the training set and the last 1996 for the testing set.

Objectives and evaluation

The objectives of the practical are twofold:

- Provide you with some practical experience in using neural networks for prediction tasks, the process of designing a neural network and tuning its hyperparameters to get the maximal performance.
- Give you some experience in using standard ML tools: Keras, Tensorflow and Anaconda.

The practical is very open on purpose. Your final objective it to obtain the maximal accuracy on the testing set. You have a complete freedom in terms of the shape, size, activation functions, etc, of the feedforward neural network.

50% of the grade will be allocated based on the accuracy of the prediction and the amount of overfitting. The quality of the report will determine the rest of the grade.

Some guidelines and hints:

- Start with simple solutions, like the ones experimented in the classes, before considering more complex ones,
- Follow a systematic approach when tuning the network: do not set values at random, do not try to optimize all parameters at once (you may change two at once to save some time),
- Overfitting will be detrimental to the performance on the testing set, hence you should pay extra attention to this phenomenon and try to mitigate it (e.g., with dropout),
- As most of the grade will be determined based on one execution on our machine, you should make sure that the accuracy is stable. One step in that direction is to set the seeds of the random generators to a fixed value as done in the template provided.

The model that will be graded is the best one that should be saved in a variable called best_model.

What is expected from you in the report:

- Explain, e.g. with respect to what we have seen during the classes or read outside the class, the design decisions for:
    1. the shape of the neural networks and its activation functions,
    2. the value of all hyperparameters.
- Explain the successive experiments having led to the final solution.
- For the final solution, please provide:
    1. a plot of the loss score over the training epochs on the training set,
    2. the prediction accuracy on the training set and testing set.

Restrictions: you must use Keras with Tensorflow as the backend, as done before in the course.

So that the results are reproducible, you must set a random seed at the beginning of the script, as follows:

library(keras)
set.seed(0)
use_session_with_seed(0)

Additional information:

You must submit on Moodle an archive file containing both your report in pdf format and the R script for the final solution. The deadline is December 21, 2019 at 23:59. This is an individual project and solutions must not be shared among you. **Solutions which we consider too similar will have their grade divided by two.** Do not hesitate to ask us for guidance during the class or by email, our role is to help you achieve the objectives**.** The weight of this practical in the final grade will be one.

The grade will consider the correctness of the answers to the questions, the quality of the code and the report (writing, presentation). An individual oral presentation of the results may be organized.