# Practical work 11 – Recurrent Neural Networks

### Objectives

The objective of this PW is to practice applications of simple Recurrent Neural Networks (RNN).

### Submission

— **Deadline** : W12 (in 1 week), before the start of the lecture.

— **Format** : Zip with the jupyter notebooks.

## Exercise 1 Language Classification by Last Names

In this example, you will develop a character-based model for detecting the language (or country of origin) for given last names.

a) Download the zip-file `name_classification.zip` with a jupyter notebook and a data folder, unzip the file, open the jupyter notebook.

b) The first cells contain some helper functions to load the training and test data. Complete the generation of the alphabet (characters to be represented) and the functionality to create the vector representation for the characters. Use a one-hot-vector representation.

c) Implement a many-to-one model consisting of a single $SimpleRNN$ and a $softmax$ for the classification (by using tensorflow/keras). Train the model - try different numbers of hidden units (dimension of hidden state vector) and play with different batch sizes. Report about your findings in the notebook.

d) Figure out a treatment of class imbalance, implement it and apply it to the given example. What improvement can you achieve ?

e) Explain why a treatment of class imbalance is important.

f) Implement a model with more than one $SimpleRNN$ layer and a final $softmax$ for the classification. Train the model - try different numbers of hidden units (dimension of hidden state vector) and play with different batch sizes. Report about your findings in the notebook.

## Exercise 2 Human Activity Recognition

In this example, you will study models for human activity recognition from accelerometer and gyroscope data recorded by smartphones (as presented in class).

The goal is to explore the hyperparameter/model space to find the best suited model for the given problem.

a) Download the zip-file `activity_recognition.zip` with a jupyter notebook and a data folder, unzip the file and open the jupyter notebook.

b) The first cells contain some helper functions to load the training and test data.

c) Now implement various different models (all are many-to-one) with different hyperparameter settings :

- Layer type : Only consider $SimpleRNN$, $Conv1D/Conv2D$, $Dense$ (with softmax) and other Helper structures such as $InputLayer$, $Flatten$, $Reshape$.
- Number of layers : 1+
- Different hyper parameters : batchsize, nepochs, number of hidden units
- With/without regularisation (e.g. dropout)

d) Compare at least 5 different architectures (pure RNN, pure CNN, pure MLP, mixtures) quantitatively and qualitatively. Spot potential issues for specific settings / models. Make sure that the results are robust against different initialisation seeds.

**Try to beat the test accuracy reported in class (on the slides) !**

## Exercise 3 Optional : Review Questions

a) What are the different forms of sequence *mapping* allowed by recurrent neural networks ? Give for each form an example of application.

b) Compute the number of parameters to be trained for a two-layer $SimpleRNN$ and $softmax$ with hidden state dimensions 32 and 64, respectively, 10 classes to classify in the softmax and inputs given by sequences of length 100 and each element a vector of dimension 30.

c) Why is gradient clipping rather needed in long than in short sentences ?

d) Describe why SimpleRNNs have problems in learning long-term dependencies.