Uppsala University

Introduction to Machine Learning, Big Data, and AI

Assignment 4

General information

- The recommended tool in this course is R (with the IDE R-Studio). You can download R **here** and R-Studio **here**. You are allowed to use Python and Jupyter Notebooks, although the assignments may use data available only through the R package, a problem you would need to solve yourself.
- Report all results in a single, `*.pdf`-file. *Other formats, such as Word, Rmd, Jupyter Notebook, or similar, will automatically be failed.* Although, you are allowed to first knit your document to Word and then print the assignment as a PDF from Word if you find it difficult to get TeX to work.
- The report should be submitted to the **Studium**.
- To pass the assignments, *all questions should be answered*, although minor errors are ok.
- A report that do not contain the general information (see template) will be automatically rejected.
- When working with R, we recommend writing the reports using R markdown and the provided **R markdown template**. The template includes the formatting instructions and how to include code and figures.
- If you have a problem with creating a PDF file directly from R markdown, start by creating an HTML file, and then just print the HTML to a PDF.
- Instead of R markdown, you can use other software to make the pdf report, but the same instructions for formatting should be used. These instructions are also available in **the PDF produced from the R markdown template**.
- The course has its own R package `uuml` with data and functionality to simplify coding. To install the package just run the following:

  1. `install.packages("remotes")`
  2. ```
     remotes::install_github("MansMeg/IntroML",
     subdir = "rpackage")
     ```

- We collect common questions regarding installation, and technical problems in a course Frequently Asked Questions (FAQ). This can be found **here**.
- Deadline for all assignments is **Sunday at 23.59**. See the course page for dates.
- If you have any suggestions or improvements to the course material, please post in the course chat feedback channel, create an issue, or submit a pull request to the public repository!
     ```

1 Neural Network Basics

As a first step, we want to implement one of the most straightforward neural networks possible.

1. Implement the neural network in Goodfellow et al. (2016, Ch. 6.1) as a function that takes as an input a 2×2 weight matrix `W`. It should work as in Ch. 6.1. Here is an example of how it should work.

```
mini_net(X, W, c, w)
##      [,1]
## [1,]    0
## [2,]    1
## [3,]    1
## [4,]    0

mini_net(X, W*0.9, c, w)
##      [,1]
## [1,]  0.0
## [2,]  0.9
## [3,]  0.9
## [4,]  0.2
```

2. Now test changing the value $W_{1,1}$ to 0. What do you get for the results of the network?

3. Is the network output function reasonable? Do you have any other output function that might be better?

2 Feed-Forward Neural Network using Keras and TensorFlow

As a first step, you need to install `tensorflow` and `keras` on your local computer. You can find detailed information on how to install `tensorflow` and `keras` **[here]**. Details on different architecture components can be found in the `keras` reference found **[here]**.

Note! Running Keras can be computationally heavy and I suggest not to run the code in `markdown`. Instead, run the code in R and copy in the results as output (see the assignment template for an example).

If you have installed `keras`, you can load the MNIST dataset. This dataset contains data on handwritten digits that we want to classify. To access the data, we use `keras` as follows.

```
library(keras)
mnist <- dataset_mnist()
mnist$train$x <- mnist$train$x/255
mnist$test$x <- mnist$test$x/255
```

1. As a first step, we visualize a couple of digits. Below is an example.

```
idx <- 3
im <- mnist$train$x[idx,,]
# Transpose the image
im <- t(apply(im, 2, rev))
image(1:28, 1:28, im, col=gray((0:255)/255), xlab = "", ylab = "",
        xaxt='n', yaxt='n', main=paste(mnist$train$y[idx]))
```

2. How large is the training and test set?

3. Implement a feed-forward neural network in R using TensorFlow and Keras. Start by following the introduction here. Start with one hidden layer with 16 units and the sigmoid as the activation function. Do not use any regularization for now. Print your `keras` model. What do you get as classification accuracy? Include the validation error per epoch figure that is automatically generated. *Hint!* The classification accuracy should be around 0.85.

4. Now, we are going to play around with the essential concepts in feed-forward networks. Do the steps below, and report the *validation* set accuracy and the validation error per epoch figures for each step. Please change this sequantially, so you build a more and more complex network.

   (a) Increase the number of hidden units to 128.

   (b) Change the activation function to reLU.

   (c) Change the optimizer to RMSprop.

   (d) Try to run the net for ten epochs. If you would use early stopping for regularization, what would you do? *Note!* You only need to describe what you would do, not actually do it.

   (e) Add a second layer with 128 hidden units. How many parameters do you have in the net right now?

   (f) Add dropout with 0.2 and 0.5 dropout probability. You can choose what layers you would like to introduce the dropout.

   (g) Add batch normalization. You can choose where you want to add it.

5. Now try to improve the network architecture yourself (but still only use a feed-forward network). How good validation set accuracy can you get? What is the architecture used for the best model?

6. Use your model to compute the accuracy, precision, and recall on the hold-out test set included with the MNIST data. If you get a lower accuracy than on the validation set, why is this the case?

References

Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.