Uppsala University

Introduction to Machine Learning, Big Data, and AI

Assignment 3

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 can 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 first to knit your document to Word or HTML and then print the assignment as a PDF from Word or HTML if you find it difficult to get TeX to work.
- You should submit the report to **Studium**.
- To pass the assignments, *you should answer all questions not marked with **, although minor errors are ok.
- To get VG on the assignment, *all questions should be answered, including questions marked with a **, although minor errors are ok.
- A report that does not contain the general information (see the **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.
- Instead of R markdown, you can use other software to make the pdf report, but you should use the same instructions for formatting. 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**.
- Deadlines for all assignments are **Sunday 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 **here**, or submit a pull request to the public repository.
     ```

Contents

1 General Questions

You will be able to answer the following questions based on the reading assignments for this assignment. See the course plan for detailed reading **here**.

1. Describe the output layer in a neural network. What does it do? (max 1 paragraph)
2. What is the benefit of using (negative) log-likelihood as loss functions according to Goodfellow et al. (2016)? (max 1 paragraph)
3. Describe early stopping regularization (max 1 paragraph).
4. Describe the relationship between bagging and dropout regularization (max 1 paragraph).

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 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.

(a) Print your `keras` model and include it in your report.

(b) How many parameters do your model have?

(c) What is the input layer in the model? How many parameters does the input layer have?

(d) What is the output layer in the model? How many parameters does the output layer have?

(e) What do you get as classification accuracy? Include the validation accuracy or error per epoch figure/plot that is automatically generated. *Hint!* The classification accuracy should be around 0.85. *Hint!* In multi-class classification, if the model classifies a digit to any other class than the valid class, it is a false negative or a false positive. I.e. we can still use the same definition for accuracy, precision and recall as in the binary case in multi-class classification.

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/error and the validation accuracy/error per epoch for each step in a figure/plot. Please change this sequentially 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 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 (or how low error) can you get? What is the architecture used for the best model?

6. Identify two digits that the network has classified incorrectly. Visualize them. Why do you think the network made the classification mistake?

7. 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 (or higher error), why is this the case?

3 * A simple neural network from scratch

This task is only necessary to get one point toward a *pass with distinction* (VG) grade. Hence, if you do not want to get a *pass with distinction* (VG) point, you can ignore this part of the assignment.

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`, a vector `c`, a vector `w` and the scalar b. It should work as in Ch. 6.1. Here is an example of how it should work.

```
W <- matrix(1, nrow = 2, ncol = 2)
c <- matrix(c(0, -1), ncol = 2, nrow = 4, byrow = TRUE)
X <- matrix(c(0,0,1,1,0,1,0,1), ncol = 2)
w <- matrix(c(1, -2), ncol = 1)
b <- 0

mini_net(X, W, c, w, b)
```

```
##      [,1]
## [1,]    0
## [2,]    1
## [3,]    1
## [4,]    0
```

```
mini_net(X, W*0.9, c, w, b)
```

```
##      [,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. What is the current output function? Is the network output function reasonable? Do you have any other output function that might be better?

4. Now implement a mean squared error loss function for the network. It should work as follows.

```
y <- c(0,1,1,0)
mini_net_loss(y, X, W, c, w, b)
```

```
## [1] 0
```

```
mini_net_loss(y, X, 0.9*W, c, w, b)
```

```
## [1] 0.015
```

5. Again, change the value $W_{1,1}$ to 0. What is the value of the loss function?

References

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