Practical Introduction to Neural Networks
Homework 1
Due: Monday April 15, 2019

Overview

Before starting with the homework please make sure that you have TensorFlow and Jupyter installed on your laptop. For each problem in this homework we have provided a Jupyter notebook template with which you can get started.

Templates are available at the course Github Repository:
`https://github.com/shlizee/PracticalIntroductionNN`

Please submit Jupyter notebooks with your code (and outputs) by the due date to your Github repository and place a link to your repository in the canvas assignment.

Problem 1: XOR gate implementation using perceptrons

One of the examples in the lecture showed how to implement a single perceptron neuron acting as logical AND gate for binary inputs. In this problem, you will implement a more complex logical gate: XOR. Since XOR gate is not linearly separable, unlike AND and OR gates, (see the figure 1) the task is to implement a multi-perceptron with nonlinear activation function.

Specifically, implement from scratch the following neural network: one-hidden layer with two neurons (i.e, you can't use tf.layers.dense function). Use the cross entropy as the cost function. To train the neural network, use the standard gradient descent (tf.train.GradientDescentOptimizer). Your answer should print the accuracy at each epoch, the outcome for each input state for the final step and to plot the decision boundary. For further details, please refer to Jupyter notebook template.

Problem 2: Iris flower dataset

Iris flower dataset contains a set of 150 records of three species (setosa, virginica and versicolor) with measurements of four features: sepal width,sepal length, petal width and petal length (figure 2). The dataset is in csv format. Please make sure you have installed **pandas** and **scikit-learn** and if not, install these packages in the Anaconda virtual environment:

```
conda install pandas
conda install scikit-learn
```

The the Jupyter notebook template includes the preprocessing of the dataset. For further details, go over the preprocessing steps line by line.

Part a:

Implement a single perceptron to perform a binary classification of setosa and non-setosa classes, using only petal width and petal length features. Use an appropriate learning rate (e.g. 0.01) and number of training

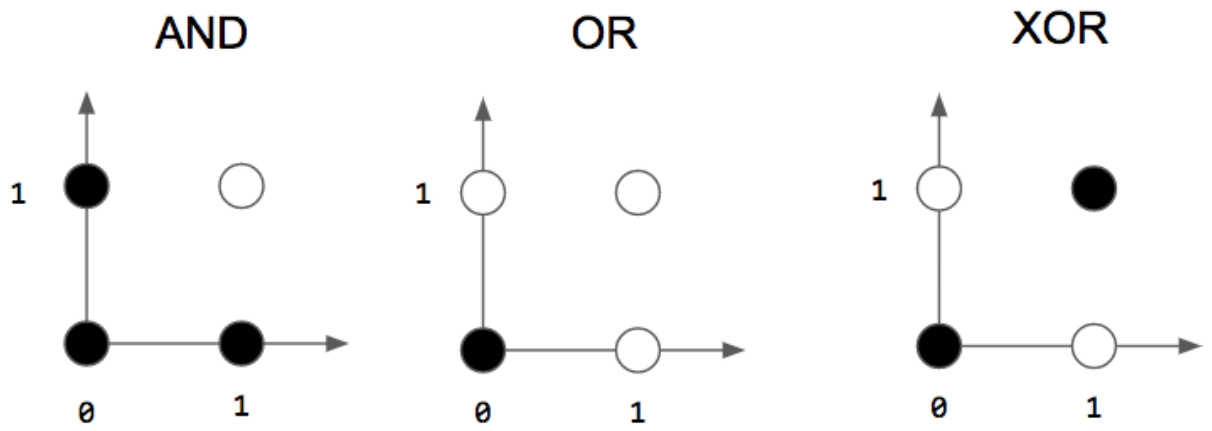

Figure 1: Logic gates

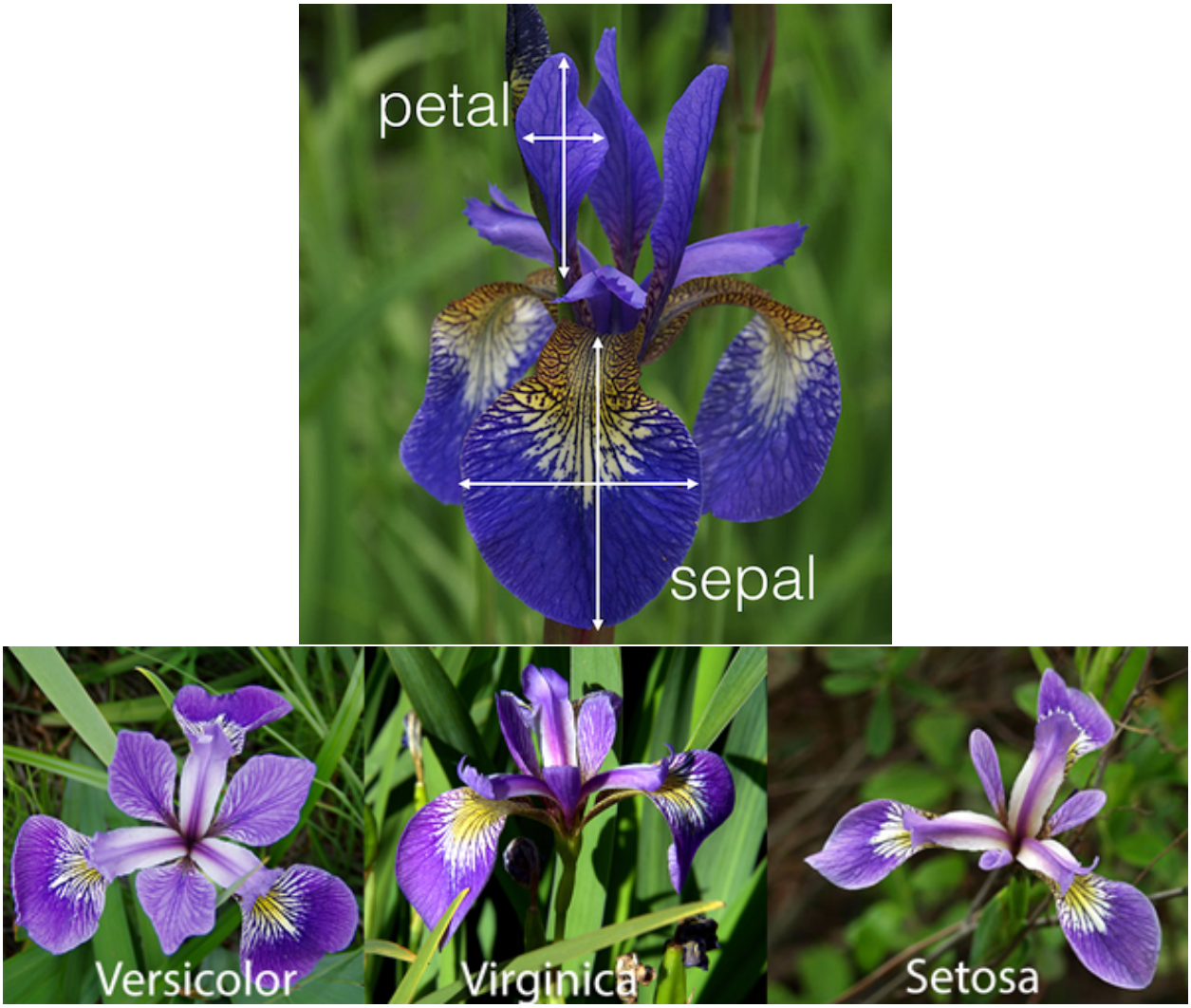

Figure 2: iris flowers

epochs (e.g. 100). Plot the accuracy curve versus epoch number, print out the test accuracy and plot the decision boundary. Which accuracy are you getting? What does it tell you about setosa w.r.t other two species? (Write your textual answers in a markdown cell).

Part b:

Implement another single perceptron to classify: virginica versus non-virginica using only petal width and petal length features. Use the same learning rate and number of epochs as in part a. Print similar measures of the accuracy as in part a. Which accuracy are you getting now? What does it tell you about the classification? (Write your textual answers in a markdown cell).

Part c:

Implement a neural network with two hidden layers to perform classification on the three species using all four features. Use the following setting: Use 256 neurons for the first hidden layer, 128 neurons for the second hidden layer. Use the ReLU function as the activation function in each hidden layer and use the softmax function as the activation of output layer. Use cross entropy function as the cost function. Plot and print out the accuracy.

Problem 3: Improving the NN for the MNIST dataset

In lecture 2, we showed a toy example for MNIST handwritten digits. However, the classification accuracy that we were able to achieve was low. Make modifications to the model based on the materials that we covered in class. The goal is to reach testing accuracy of at least 96%. Try to reach the highest accuracy as possible without drastically amending the network's overall architecture. Among the different parameters, you can vary the learning rate, activation functions, number of neurons, number of layers, number of epochs, batch size, etc. Make a table of your investigations and the performances of the different settings. Explain your results in a markdown cell. In particular, include a summary of parameters that contributed the most to improvement in performance and which did not change it. See if you can explain some of these results.