Johannes Gutenberg-Universität Mainz
Institut für Informatik
Data Mining
Prof. Dr. Stefan Kramer

Machine Learning - Sheet 8

16.06.2017
Deadline: 23.06.2017 - 23:55

Task 1: Handwritten Digit Recognition (Part 2) *(12 Points)*

We now want to experiment with an actual implementation of the backpropagation algorithm. We will use TensorFlow to train a tiny neural network that is supposed to classify MNIST digits.

(1) Read the TensorFlow's MNIST tutorial .

(2) Use TensorFlow Python API to implement the network sketched in Figure 1.

(3) Visualize the weights $W^{(1)}$ as follows: right after the training, take a few (at least 10) columns of the matrix $W^{(1)}$, reshape each 784×1 column into a 28×28 matrix, and render it as a little image.

(4) Train the neural network on the MNIST dataset. Experiment with different settings for the gradient descent, vary the dimensions of the hidden layer.

(5) Report your results (∗).

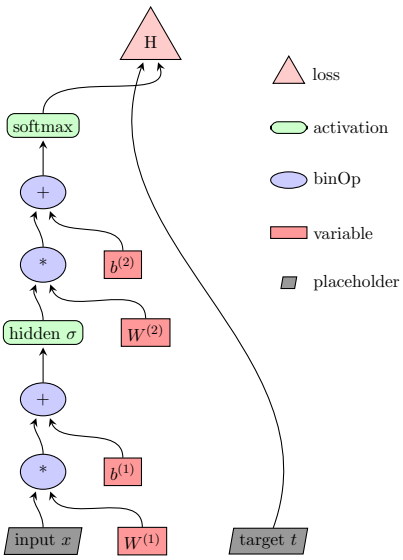

Figure 1: Simple ANN for handwritten digit classification (softmax and cross-entropy loss are shown as separate nodes, in the actual implementation they should be merged).

(∗) *Neural networks can take more than just a few minutes to train, so it will not be possible to re-run your code. Therefore, submitting only code is not sufficient: a short PDF document with the relevant code snippets and a few plots is expected instead.*

Task 2: Voronoi Diagram *(8 Points)*

Given are the following instances (attributes from $\mathbb{R}^2$, class label from $\{\oplus, \ominus\}$):

$$(3,1)\ominus, (9,2)\ominus, (5,3)\ominus, (8,5)\ominus, (7,7)\ominus, (1,4)\oplus, (3,4)\oplus, (1,8)\oplus, (4,9)\oplus, (5,6)\oplus$$

(1) Draw the points in the $[0,10] \times [0,10]$ square.

(2) Draw the Voronoi diagram associated with the set of points.

(3) Draw the decision boundary of the 1-NN classifier between the two classes.

If you want, you can automate some of the steps. Here is the data as Java code snippet:

```
int[][] points = {
  {3,1}, {9,2}, {5,3}, {8,5}, {7,7},
  {1,4}, {3,4}, {1,8}, {4,9}, {5,6}
};
boolean[] labels = {
  false, false, false, false, false,
  true, true, true, true, true
};
```