Character Recognition Using Artificial Neural Networks

Michael Single
University of Berne
08-917-445

Stefan Moser
University of Berne
09-277-013

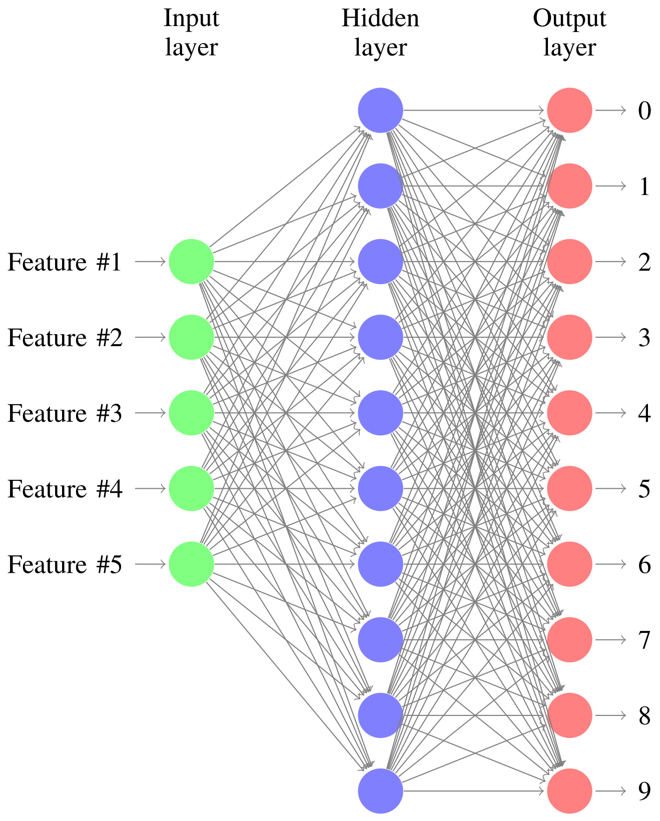


Fig. 1. Structure of a artificial neuronal network with 5 input features, a hidden layer with 10 nodes and our 10 output classes, the numbers from 0 to 9. Every node is connected to all nodes of the next level.

I. Artificial neural networks

Every neuron as a number of input signals x that are weighted with the neurons weights w. The *activation* a of a neuron is computed as

$$a = \sum_{i=1}^{n} x_i \cdot w_i \tag{1}$$

The *output signal* y further refines the activation by applying a bias w_0 and applying a non linear function f

$$y = f(a - w_0) \tag{2}$$

For the results obtained in this report, we used the sigmoid function for f

$$f(t) = \mathrm{sig}(t) = \frac{1}{1+e^{-t}} = \frac{1}{2} \cdot \left(1 + \tanh\frac{t}{2}\right) \tag{3}$$

The weights w of each neuron are learnt using *reinforcement learning*. Since we never received any source code for the library we used, we can not comment on how exactly this is done, except that the initial weights are chosen randomly, leading to a slightly different outcome in every run.

II. Results

The structure of the neural network largely decides its performance. An example structure can be seen in Figure 1. The main focus of this report is to analyse the results produced by different structures and different input features. Our first experiments were done using the pixel values[1] directly as features.

One observation we made was, that the learning rate must be set dependent on the the number of neurons used. A plot comparing these two properties can be seen in Figure 2.

We could also observe the most prevalent disadvantage of artificial neural networks. While the performance with large datasets is quite good, they take a long time to learn. If the training set size is reduced, accuracy drops considerably. Some trials showing this behavior can be seen in Figure 3.

A. Advanced Features

To decrease the time needed for training, we reduced the dimensionality of the problem by decreasing the number of features. We did this by simply downscaling the image to a fourth. As expected, this brought a great improvement in learning time, as the dimensionality of the problem was brought down from 748 to 49. But at the same time accuracy decreased for high number of neurons. The effect was larger, the smaller the learning rate was, as can be seen in Figure 4. With only few neurons used in the hidden layer, the accuracy was about the same.

[1]Normalized to the range $[-1, 1]$

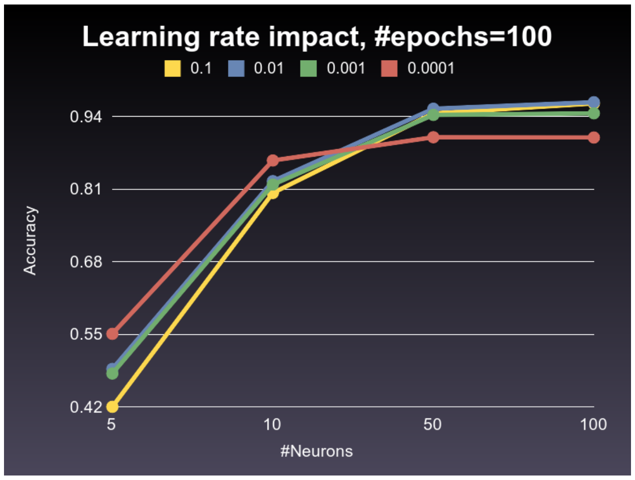


Fig. 2. Different learning rates with different number of neurons. It becomes apparent, that very low learning rates work well with low numbers of neurons, but high learning rates work well with higher number of neurons.

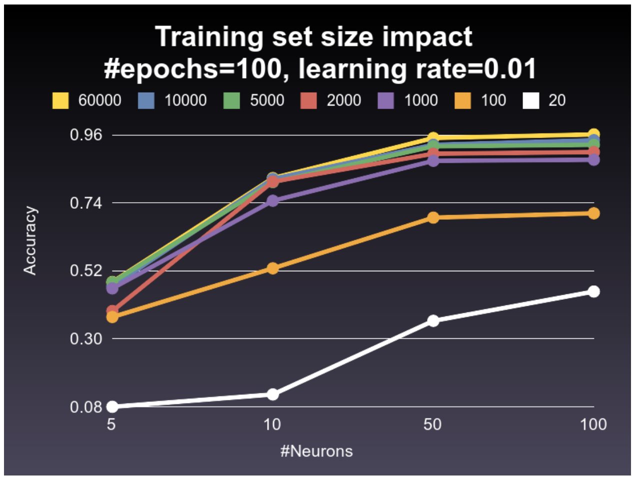
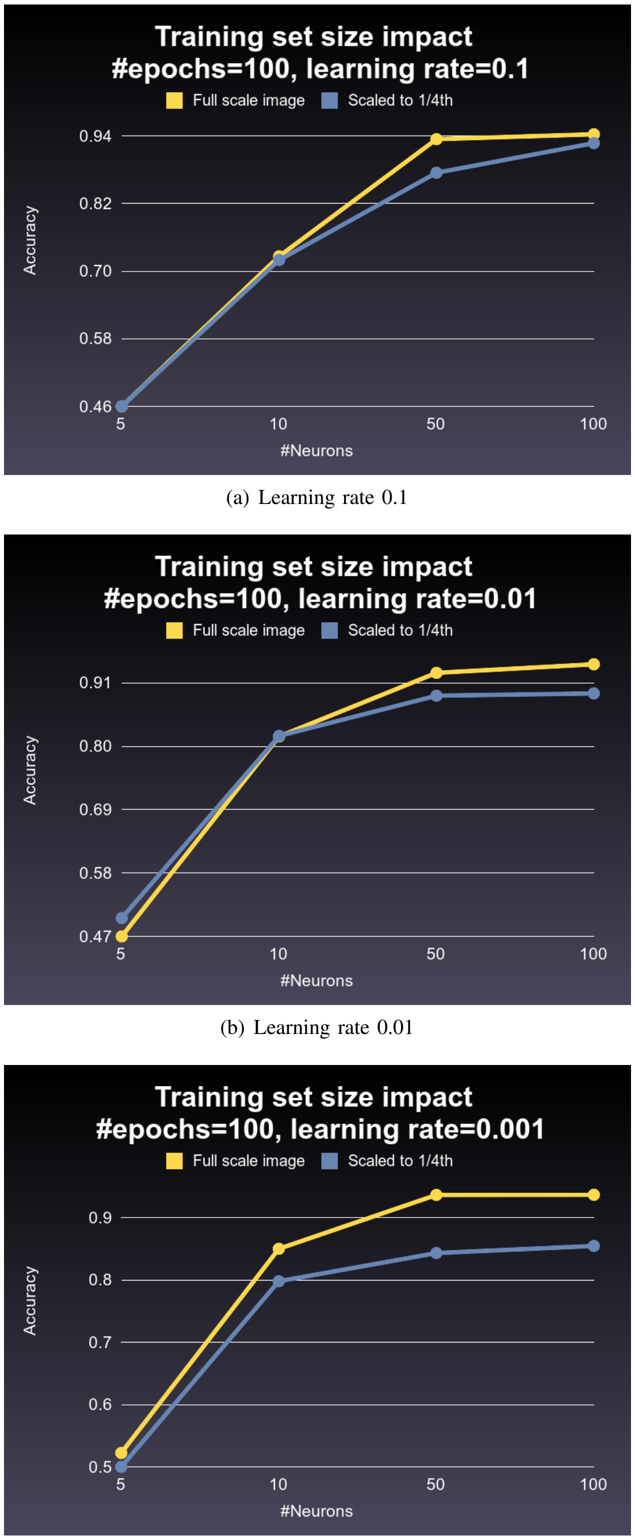


Fig. 3. Different sizes of training sets with different number of neurons. As expected, the more training samples available, the better the results. However, the time used for learning does also grow with training set size.



(a) Learning rate 0.1

(b) Learning rate 0.01

(c) Learning rate 0.001

Fig. 4. Comparisons between our two tested features: In yellow the pixel values of the full image (748 scalar values), in blue the pixel values of the image scaled to one fourth (49 scalar values).