Introduction to Machine Learning (SS 2023)
Programming Project

Author 1
Last name: Ebner
First name: Matthias
Matrikel Nr.: 12115812

Author 2
Last name: Paul
First name: Prünster
Matrikel Nr.: 12107391

I. Introduction

In this report, we present an in-depth analysis of a hand gesture classification task, specifically focusing on recognizing sign languages using image-based approaches. Our objective was to classify a dataset of 9,680 hand images, each measuring 128x128 pixels, into 36 distinct classes. These classes encompassed the numbers from 0 to 9 and the letters from 'A' to 'Z', representing a comprehensive range of signs commonly used in sign language communication. Our task now was to build a machine learning model that can accurately classify images from this dataset.

II. Implementation / ML Process

We had to make sure that our chosen Machine Learning methods can work with 128x128 px images. We also had to resize them into float format and in the interval [0,1] for normalization, so we divided by 255. We identified it as a typical Convolutional Neural Network (CNN) problem quite fast. This would be the obvious choice. We also experimented with a Fully Connected Neural Networks just to compare, but definitely expected that CNN's are more suitable for image classification. Generally speaking, Neural Networks can be extremely effective and can work with raw data without prior feature extraction. However, Neural Networks are difficult to design, perfor- mance often critically depends on design details and training requires lots of data and computing power. We heard a lot about Fully Connected Neural Networks in the Lecture. Convolutional Neural Networks were also briefly covered in the VO Machine Learning.

A. Convolutional Neural Networks

Convolutional Neural Network CNNs can be used to find, extract and recognize objects. Those neuronal networks are based on two different types of layers, the convolutional layers, and the pooling layers.

1) Convolutional Layers: Convolutional layers combine the input from the previous layer or input layer and output its results to the next layer. Each convolutional neuron only has inputs for neighboring pixels. The convolutional layers are therefore not fully connected. This helps to track local

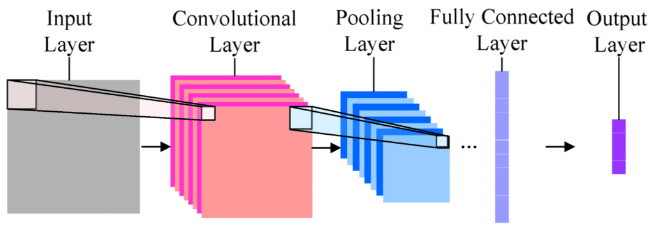

Fig. 1. Layers of a CNN

features concerning neighboring features. It also decreases the computation effort significantly. [**?**]

2) Pooling Layers: Pooling layers reduce the dimensions of the data by combining the outputs of many neurons from the previous layer to a single input to the next layer. After a mix of convolutional and pooling layers, the data goes through some fully connected layers. This makes the location, rotation, and scale of the object irrelevant. The Networks get trained through a training set and back propagation. This works by inputting the training images with already known objects and calculating the error. The error then gets back propagated, and each weight is adjusted accordingly. After a vast amount of pictures and back propagation's, the network can classify new images. This training stage is computational very demanding, but the classification can happen with relatively low effort.

B. Fully Connected Neural Networks

A Fully Connected Neural Network is as simple as it gets. There are input and output nodes. These are as the name implies, fully connected to the neurons of the hidden layers. This means they pass their value via a weight to all neurons of the next layer. All hidden layers are also connected to all nodes of the next layer, unlike in a Feedforward neural network. Here, weights can also be missing from one neuron to a neuron of the next layer. This means not all the neurons are connected. We chose a simple fully connected neural network, as we wanted to compare a complicated Network like a Convolutional Neural Network to one of the simplest networks there is to see if the effort of implementing and training a more complicated network will be worth it.

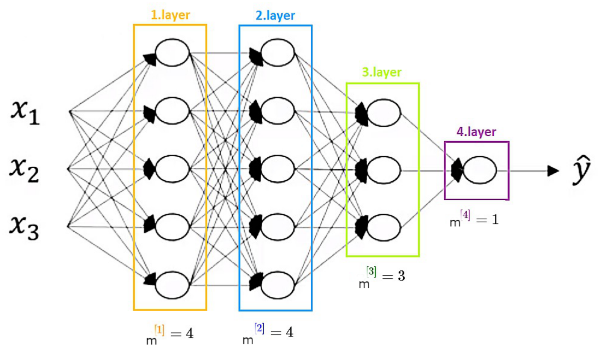

Fig. 2. A fully connected network [?]

	Training set	Validation set
Fully Connected	98.64%	80.41%
CNN	96.45%	86.71%

TABLE I
ACCURACY RESULTS

III. Results

The fully connected network has a 98% score on the training set and a 80% score on the hidden validation set on jupyterhub. The Convolutional neural netowrk does perform a bit worse on the training set but way better on the validation set. This could be because due to time constraints, we trained the Convolutional neural network less than the fully connected one. It also could mean that we overfitted the fully connected one more on the training dataset, as it gets a lower score on the validation set. This is an indicator of a bigger generalization error. As you can see in the loss plot, the convolutional neural network takes longer to train. It also has a bit more variance in the loss at the beginning.

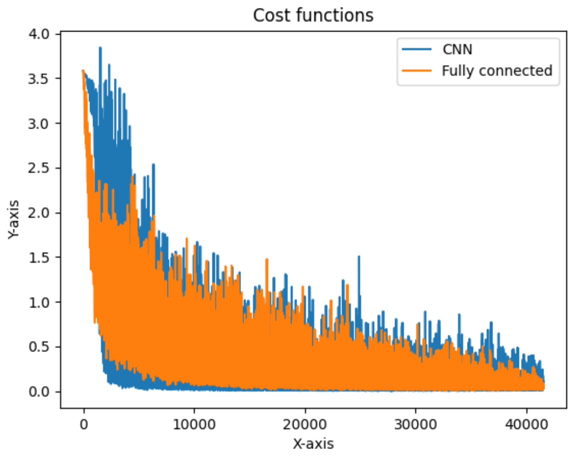

Fig. 3. Loss plot

IV. Discussion

We tried three different methods, with a Convolutional Neural Network and a Fully Connected Neural Network. In the end, we managed to get both of them running. At first, it seemed that the fully connected network would outperform the CNN. It also was way faster to train. But the test results lead to the Convolutional Neural Network being our final implementation. We also used minibatches with a batch size of 64. This was a crucial speedup, as none of us had an Nvidia GPU and we had to train on our poor Laptop CPU. This was surprisingly slow on one CPU and not feasible at all on the other one. Training on GPU is definitely the way to go, as this would be way faster.

V. Conclusion

With a lot of training with the training set, we achieved the 86.71% accuracy on the hidden test set. As expected, Convolutional Neural Networks are prob- ably the most suitable for Image Classification. It was quite surprising to see that a simple neural network is basically as good as a CNN at recognizing images, while being a bit easier and faster to train. It is probably possible to further increase the accuracy by experimenting with different settings. Another approach for better accuracy would be to extract features of the data beforehand and then train the network. Our main takeaway is how surprisingly long the training needs. With a pretty strong 16 core CPU, the CNN still took over half an hour to train for 50 epochs. The library PyTorch is also very nice and easy to use and a very powerful abstraction. All in all, we learned a lot about Neural Networks, found the project very interesting. It was fun training a model for long and then seeing the high scores.

References

Benoit Liquet Nazarath2021fullyConnected Benoit Liquet, S. M. Nazarath, Y. 2021, '4 general fully connected neural networks', https://deeplearningmath.org/general-fully-connected-neural-networks.html/. [Online; accessed June 29, 2023]. Tondak2022cnn Tondak, A. 2022, 'Convolutional neural network (cnn)', https://k21academy.com/microsoft-azure/convolutional-neural-network/. [Online; accessed July 2, 2023].