Report

Contents

1 Overview of Convolutional Neural Networks

Convolutional Neural Networks (CNN) are a certain kind of neural networks, mostly used for image, text and speech classification. Like the name says, a CNN uses convolution in at least one of its layers. The architecture of a CNN has usually the form Convolution - Pooling - ... - Convolution - Pooling - Fully connected layers. It is also possible that after a convolutional layer, another convolutional layer follows directly before pooling is applied.

The convolution operation is done by using filters, also known as kernels. These filters are not manually defined, but rather learned by the network during the network is trained. What has to be defined manually is the (randomly) initialization of the weights for the filters, define how many filters with which dimension have to be used and further things like padding, stride size etc. A convolution operation is followed by a non-linear activation function, typically ReLU. This function decides if the a certain feature is used or not. In the case of pixels of an image, the non-linear activation function sets the pixel value either to zero or it takes the computed (positive) value.

Pooling has the goal of combining the most important features by applying the maximum or the average over a certain number of values. In the case of an image, a certain number of pixels is pooled by taking the maximum or average pixel value.

The training of a CNN is quite demanding, as the loss function, which has to be optimized, is not nicely convex (as it is the case for neural networks in general). This requires some sophisticated techniques like early stopping, dropout and trying out different learning rates.

2 CIFAR-10

Mini project

The goal of this mini project is to train a Convolutional Neural Network (CNN) to classify the images of the CIFAR-10 dataset. The framework we used is **Tensorflow** with **Keras** as API. In the following we will describe the data and the approach used to train the classifier.

Model "Simon"

The general code structure is from https://keras.io/examples/cifar10_cnn/. The CNN architecture is inspired by http://parneetk.github.io/blog/cnn-cifar10/. This network achitecture gave a test accuracy of 0.8069 after 30 epochs.

- **Optimizer**: We used RMSprop as an optimizer.
- **Learning rate**: set to 0.0001.
- **Activation function**: Like usually applied for CNN, we used the state-of-the-art ReLU activation function.
- **Dropout**: We used for dropout 0.25, so that randomly 25% of the neurons/units were set to zero.
- **Padding**: For the first layer in the hierarchical convolution, we used zero padding. For the second layer, i.e. the layer before the pooling is applied, we did not use padding.
- **Number of filters**: In the first two layers I used 48 filters, in the next two layers 96 filters, and in the last two layers 192 filters.
- **Kernel sizes**: For the kernel size, usually a small kernel/filter is used. We used for all layers a 3×3 kernel.
- **Pooling**: I used the classical max pooling, as this is the state-of-the-art for text classifications for CNN.