Classification of images with Convolutional Neural Networks

This scripture documents the work done by Aditya Raj () and Sören Schleibaum (474562) for the course Neural Networks with Statistical Learning at the Technical University of Clausthal. The date of committing is the 13th of March 2017.

# Structure

[1 Structure 2](#_Toc476778445)

[2 Introduction 3](#_Toc476778446)

[3 Problem 4](#_Toc476778447)

[4 Convolutional Neural Networks 5](#_Toc476778448)

[5 TensorFlow 6](#_Toc476778449)

[6 Description of programming code 7](#_Toc476778450)

[7 Evaluation 8](#_Toc476778451)

[8 Summary 9](#_Toc476778452)

[9 References 10](#_Toc476778453)

# Introduction

* Define Deep Learning
* Define Neural Network
* Define Convolutional Neural Networks
* Define Classification (unequal to clustering)

# Problem

* Dataset from Kaggle[[1]](#footnote-1)
* 12,500 images to predict
* 25,000 images for training
* To classes for classification
  + Cat
  + Dog

# Convolutional Neural Networks

* Layers
  + Input
  + Convolutional
  + Pooling
  + Normalization
  + Local
  + Softmax linear
  + Output

# TensorFlow

## Description

The software library TensorFlow supports operation used in machine learning. It was developed by the Google Brain Team to support Deep Learning Neural Networks and released in November 2015. Multiple programming languages for instance Java and Python are supported. The library provides a support for computation on both, Central Processing Unit (CPU) and Graphics Processing Unit (GPU).

The basic concept behind TensorFlow is the usage of graphs which provide the computations inside their nodes and tensors which hold the data. The nodes are connected through edges. Tensors are a class provided by the TensorFlow library. These classes can store multi-dimensional arrays. For instance, an RGB image of height and width of 60 images is represented as a tensor of shape 60x60x3. Within the graph model a tensor can ‘flow’ inside a node. Nodes represent computations. It can have multiple inputs and outputs. The incoming data from an edge is processed within the node and passed to all its outgoing edges.

* Background
* Graph model
  + A session must run, otherwise no computation is possible
* Tensors
  + Holds the actual data
  + Everything is stored as a tensor
  + Multi-dimensional error
* Machine Learning Library
* GPU and CPU programming

## Usage

* Only CPU computation is used due to the lack of
* During this project version 1.0 together with the Python version 3 language was used.

# Description of programming code

## Graph

The design of the CNN used within this work is shown in Figure 1: *Structure of the implemented CNN*.

* Images are represented as 24x24x3
  + 3 for RGB

## UML-Diagramm

## Description of modules

# Evaluation

## Environment

The machine used to train the network is 64-bit Linux

* Describe the computer used
  + Sören
  + Linux 16.04 Machine
  + Intel Core i7-4510U CPU with 2.00 GHz x 4
  + Memory 7.7 GB
  + 64-bit

## Solutions

# Summary

# References

* TensorFlow documentation
* Stanford course for neural networks
* Python 3 documentation

# Appendix

Figure 1: Structure of the implemented CNN

input

conv

conv

pool

norm

norm

pool

local

local

Softmax linear

24 x 24 x 3

128 x 24 x 24 x 3

128 x 24 x 24 x 64

128 x 12 x 12 x 64

128 x 12 x 12 x 64

128 x 12 x 12 x 64

128 x 12 x 12 x 64

128 x 6 x 6 x 64

128 x 384

128 x 192

1. kaggle.com [↑](#footnote-ref-1)