Department 07
Master Computer Science

Deep learning - Dog Breed Classification

Realization of an native Android app using deep learning algorithms

Alice Bollenmiller, Andreas Wilhelm
WS 17/18 IG
January 22, 2018

Contents

1 Introduction

In this capitel, a short introduction leads to the subject of Deep learning. Furthermore, the scope of this work and its points of referencee are described and localized.

1.1 Deep learning

In 2015, AlphaGo - a computer program developed by Google's DeepMind Group that was trained to play the strategic board game Go - was the first program to defeat the multiple European champion Fan Hui under tournament conditions. Back then, terms like Artifical Intelligence (AI), Machine Learning and Deep Learning were talked of, because those were the reason why a computer program was able to defeat a human being. One of the most frequent used techniques of AI is Machine Learning. It uses algorithms to parse data, process it and learn from it. The result is a prediction or determination as a conclusion of what was learnt from the dataset. Deep Learning - which is part of the Machine Learning techniques - sells its application particularly in the field of language and image processing.

In 1958, Frank Rosenblatt introduced the concept of the perceptron which is the fundamental idea of all Deep learning approaches. It consists of multiple artifical neurons which are coupled with weights and biases. In the case of a single-layer perceptron, the input nodes are fully connected to one or more output nodes. During the learning process the weights are adapted according to the learning progress. When such a structure is extended with layers, it becomes a multi-layer perceptron (MLP). This basic structure can be found in special neuronal network architectures e.g. Convolutional Neuronal Networks (CNN). CNNs which are frequently used for object detection and image/ audio recognition etc. consists of multiple neurons. When a neuron receives an input, it performs a dot production and adapts the weights. Because of their special structure, CNNs are able to detect local properties of an image. Basically, the network represents a differentiable score function which is applied on the raw image pixels and computes the class scores as an output.

1.2 Terms of Referencee

The problem of Deep learning architectures is their high performance requirements regarding computational power. Common frameworks for open source development in the field of deep learning which are discussed in —- introduced several models for the integration in mobile applications.

In order to overcome the above mentioned problem, optimizing methods will be combined with appropriate models which require less computing power and are suitable for mobile application integration. As a result of this work, a dog breed analyzer will be implemented. This mobile app will take a live stream as an input and determine the breed of the focused dog. The three highest probabilities of breeds will be shown by the app.

2 Methodological fundamentals

2.1 Common Frameworks for Deep Learning Applications

- some examples, tensorflow (tensorflow slim -¿ High level api for easier use, tensorflow lite), Caffe, Keras, Torch, PyTorch, ...
https://datahub.packtpub.com/deep-learning/top-10-deep-learning-frameworks/

2.2 Common Models in Deep Learning Applications

- short differences between different architecuteres (?, CNN, RNN)
- AlexNet, Mobilenet, Inception, VGG, -¿ short decsription, useCases, important things, differences

2.3 Qualified Models for mobile App Integration

- Mobilenet, Inception etc -¿ short decsription, useCases, important things, differences

2.4 Key requirements for an appropriate dataset

- generall why you need a huge dataset -¿ different backgrounds
- self trained needs a huge dataset, a lot of computing performance and time
-¿ so use pre trained, if small dataset.
-¿ pretrained used millions of pictures (e.g. ImageNet)

3 Concept

3.1 Frameworks

- tensorflow -¿ why

3.2 Model based Architectures

- general architectures of models -¿ Mobilenet, Inception

3.3 Application based Architecture

4 Realization

4.1 dataset

4.2 hardware environment

used CPU, GPU -¿ NVIDIA, handys

4.3 software environment

- Bazel, Java, Android Studio, Python, Operating System
- Android system

4.4 installation of software

- software environment

4.4.1 Tensorflow based on Python

4.4.2 Tensorflow based on Bazel

- e.g. Workspace changes for Android SDK, msse4.2

4.4.3 Installing Android Studio and its Delevopment Kit

- also possible with bazel but easier Android studio (needs correct versions of sdk, ndk)
- SDK, NDK
- IMPORTANT: tf versions updaten (same as trained)

4.5 building the models

-¿ evtl extra subsubsection:
- execution methods -¿ Bazel and Python (incompatible versions)
- Mobilnet -¿ steps, optimierung
- Inception -¿ steps, optimierung
- time related differences of execution
-¿ time CPUs/GPU

4.6 Output Tests and Validation

- test pictures and if it works -¿ label image
- validation script?!

4.7 Implementation of an native Android App

- list all necessary things to do (e.g. tensorflow version, Interpreter -¿ load Model)

4.8 Deployment and Validation

5 Evaluation

- prio von nierdig zu hoch
- regarding implementation time
- regarding performance
- regarding quality in accuracy
- handy perfomance?

6 Conclusion

- tutorials not complete, different
- which model is better
- prospects, improvements, Recommendations