

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	1
1.1	Deep learning	1
1.2	Terms of Referencee	1
2	Methodological fundamentals	1
2.1	Common Frameworks for Deep Learning Applications	1
2.2	Common Models in Deep Learning Applications	1
2.3	Qualified Models for mobile App Integration	1
2.4	Key requirements for an appropriate dataset	1
3	Concept	1
3.1	Frameworks	1
3.2	Model based Architectures	1
3.3	Application based Architecture	2
4	Realization	2
4.1	dataset	2
4.2	hardware environment	2
4.3	software environment	2
4.4	installation of software	2
4.4.1	Tensorflow based on Python	2
4.4.2	Tensorflow based on Bazel	2
4.4.3	Installing Android Studio and its Delevopment Kit	2
4.5	building the models	2
4.6	Output Tests and Validation	2
4.7	Implementation of an native Android App	2
4.8	Deployment and Validation	2
5	Evaluation	2
6	Conclusion	3

1 Introduction

1.1 Deep learning

- what is deep learning -> purpose, usage, current research projects, state of the arts

1.2 Terms of Reference

- dog breed analyzer -> goals, purpose,
- > high performance computing but native android 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 architectures (?, CNN, RNN)
- AlexNet, Mobilenet, Inception, VGG, -> short description, useCases, important things, differences

2.3 Qualified Models for mobile App Integration

- Mobilenet, Inception etc -> short description, useCases, important things, differences

2.4 Key requirements for an appropriate dataset

- generally 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 niedrig zu hoch
- regarding implementation time

- regarding performance
- regarding quality in accuracy
- handy performance?

6 Conclusion

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