Machine Learning Nanodegree

Capstone project

author: Michael Kögel
date: 28th July, 2018

Project Proposal

Domain Background

A main application for the machine processing of image and sound data are convolution neuronal networks (CNN). It is a concept inspired by biological processes in the field of supervised machine learning, but is far from plausibly modeling neural processing. Therefore we use the method for image and video recognition, recommender systems and natural language processing.

For the following capstone project of the machine learning Nanodegree from Udacity I train a CNN to classify the species of mushrooms from images. The idea is based on the project Deep Shrooms from the University of Helsinki, in which three students developed a smartphone application to classify wild mushrooms to edible/not edible based on image input from the users camera. As you can imagine, the classification result should be treated with caution. False positive (positiv = edible) assessment may result in serious health consequences for the user and is therefore ethically unacceptable (see the link for a blog post to this topic).

The project is not about the decision whether a mushroom is edible or not. Instead the application should help to understand the biodiversity of mushrooms in Central European forest areas.

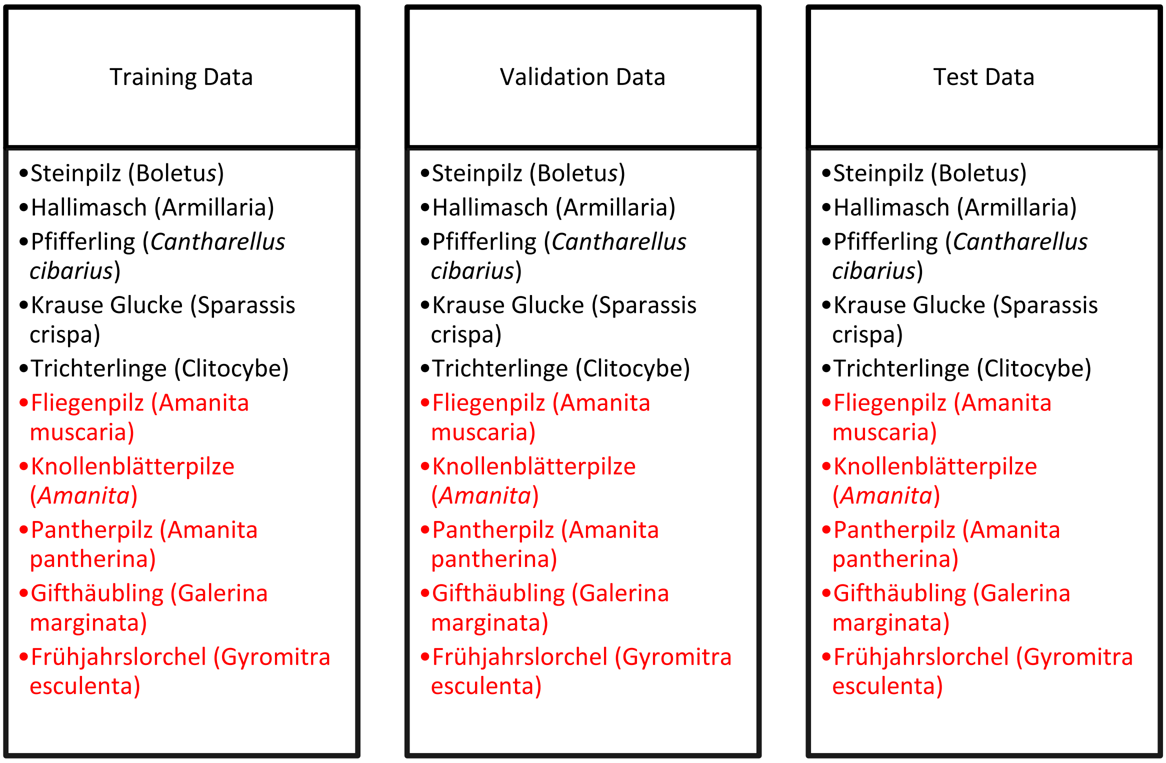
Problem Statement

Identifying mushroom species requires a basic understanding of their macroscopic structure. Separating species requires meticulous attention to detail. There is no single trait that mushrooms can be separated into single species. For this purpose, a recognition system is needed.

The main task of my project is to create an application which classifies user inputted pictures of wild mushrooms according to their species, using a machine learning classifier based on a convolutional neural network. In result the application calculates the probability for each species of mushrooms, the image may belongs to.

Datasets and Inputs

I want to classify at least 10 different species of mushrooms and I will use only very few training examples to build the image classifier. The data directory contains three subdirectories for training, validation and test. Each subdirectory has 10 subfolders for each type of mushroom. In the following graphic (Figure 1: Directory composition of dataset) the data directory composition is displayed.



Training Data

- Steinpilz (Boletus)
- Hallimasch (Armillaria)
- Pfifferling (*Cantharellus cibarius*)
- Krause Glucke (Sparassis crispa)
- Trichterlinge (Clitocybe)
- Fliegenpilz (Amanita muscaria)
- Knollenblätterpilze (*Amanita*)
- Pantherpilz (Amanita pantherina)
- Gifthäubling (Galerina marginata)
- Frühjahrslorchel (Gyromitra esculenta)

Validation Data

- Steinpilz (Boletus)
- Hallimasch (Armillaria)
- Pfifferling (*Cantharellus cibarius*)
- Krause Glucke (Sparassis crispa)
- Trichterlinge (Clitocybe)
- Fliegenpilz (Amanita muscaria)
- Knollenblätterpilze (*Amanita*)
- Pantherpilz (Amanita pantherina)
- Gifthäubling (Galerina marginata)
- Frühjahrslorchel (Gyromitra esculenta)

Test Data

- Steinpilz (Boletus)
- Hallimasch (Armillaria)
- Pfifferling (*Cantharellus cibarius*)
- Krause Glucke (Sparassis crispa)
- Trichterlinge (Clitocybe)
- Fliegenpilz (Amanita muscaria)
- Knollenblätterpilze (*Amanita*)
- Pantherpilz (Amanita pantherina)
- Gifthäubling (Galerina marginata)
- Frühjahrslorchel (Gyromitra esculenta)

Figure 1: Directory composition of dataset

The table entries contain the german words for the mushrooms species. In parentheses you can find the respective scientific names. The font color indicates whether the mushroom is edible (black color) or not (red font color), just for interest.

I need to collect the data by myself. That is why I want to use only a few examples to train, valid and test the model. The plan is to use between 60 and 100 images of each specie for training. Another bunch of 10 to 20 images for each specie should be used for validation and verification.

Solution Statement

The goal of the project is to develop an application that is able to classify mushrooms according to their species based on images. To do this, the project presents a few effective methods that will be used to build a powerful image classifier, using only very few training examples.

To build the application I need to collect images from databases / websites and subdivide them into groups of their species. After the data collection and data preprocessing I want to figure out, how well an Deep Residual Network is able to detect mushrooms in my collected images (regarding to the chapter 5 project from the course).

In the first analytical part of my project we use the preprocessed and transformed data to train a small convolutional neuronal network and see how this network performs on the small dataset. In the second part the approach would be to leverage a network, pre-trained on a large dataset. I am using the bottleneck features of a pre-trained network and build a resulting more complex CNN. In the last step I apply fine-tuning the fully connected layer and illustrate how the model can be further improved.

Benchmark Model

I couldn't find a project that classifies images of mushrooms by their species. As an orientation, I will use the results from a codelabs project. In this project the user learns how to run TensorFlow on a single machine, and will train a simple classifier to classify images of flowers. In the verification part, the model achieves an accuracy of over 90%, which should also be the goal in my project.

Another benchmark is to run a simpler model (from the first solution part) on the same data and compare the results with our more complex model (from the second solution part). Therefore the simple model from scratch contains a stack of 3 convolution layers with a ReLU activation and followed by max-pooling layers. On the top I will put two fully-connected layers with sigmoid activation function for binary classification.

Also interesting could be the comparison to a similar project, like the resulting output of the model from the project Deep Shrooms, I mentioned at the beginning of the proposal. It will be interesting to see whether the subdivision into subcategories of mushroom species achieves better results than the subdivision between edible and non-edible.

Evaluation Metrics

The model predications for this classification problem can be evaluated in several ways. I calculate the accuracy for the models based on the verification dataset. When adapting the hyperparameters of the pre-trained CNN I can also use the accuracy value to see the improvements.

To visualize the performance of the algorithm a confusion matrix for the tests of the CNNs will be used. This allows more detailed analysis than mere proportion of correct classifications. There is also a statement about the cross sensitivity between the species of mushrooms. An interesting evaluation would also be to distinguish the sensitivity and specificity between edible and non-edible mushrooms in the result.

Project Design

Collecting data

Data can be found using databases like: http://www.mushroom.world/ or this Finnish website: http://www.funga.fi/linkkeja/#sienisanasto. I will also collect images by myself using the chrome extension Fatkun Batch Download Image.

Residual Network

I will extract predictions to object classes of the images from the Deep Residual Network ResNet50. How many of the images can be predicted to the object class "mushrooms". The results can be used to provide information about the quality of the data set.

Data preprocessing

With the ImageDataGenerator class from Keras, the data will be "augmented" via a number of random transformations. The size of the images will be reduced for all images to a fixed size for the CNN input. Colored images will be used, because they contain a high value of self-information for the defined problem.

Small convolutional neuronal network

As an initial baseline, I want to build a small layered convolutional network from scratch. To reduce the effect of overfitting I will use different ways to modulate entropic capacity by adding dropout layers, optimize the number of layers and the size of each layer and use weight regularization (such as L1 or L2 regularization).

Using the bottleneck features of a pre-trained network

A more refined approach would be to leverage a network pre-trained on a large dataset. I will use the VGG16 architecture, pre-trained on the ImageNet dataset. This model will already have learned features that are relevant to our classification problem.

I will only instantiate the convolutional part of the model (up to the last fully-connected layers). We will then run this model on our training and validation data once, recording the output. Then we will train a small fully-connected model on top of the stored features.

Fine-tuning the hyper-parameters of the network

The focus is to increase the accuracy of the model by fine-tuning the last block of the pre-trained network and the top-level classifier. I will apply different techniques like using slow learning rates, reducing entropic capacity of the network and use the SGD optimizer rather than an adaptive learning rate optimizer.

Evaluation

The main comparison criterion is the accuracy of the trained CNNs. I will also show the confusion matrix of a set of verification data and show a possibility to evaluate the sensitivity and specificity.

Additional

If sufficient time should be available, I will retrain a different convolutional network like MobileNet on the training data and compare the results to the VGG16 bottleneck approach. A further addition would be porting the application to the raspberry pi with camera extension.

Links and Sources

https://tuomonieminen.github.io/deep-shrooms/

https://pdfs.semanticscholar.org/8efa/ea4085e64785143e21f1797e9c2c95c8f2f7.pdf

http://www.mushroom.world/

http://www.image-net.org/

https://www.theverge.com/2017/7/28/16054834/mushroom-identifying-app-machine-vision-ai-dangerous

http://www.funga.fi/teema-aiheet/sienten-tunnistaminen/

https://chrome.google.com/webstore/detail/fatkun-batch-download-ima/nnjjahlikiabnchcpehcpkdeckfgnohf/RK%3D2/RS%3DPnB3CMxxSoOYRnLD3KKFviCVQvs-

https://keras.io/preprocessing/image/

https://keras.io/

https://www.kaggle.com/keras/vgg16

https://keras.io/applications/#mobilenet