Dog Breed Identification using Deep Learning (Image Classification)

Muhammed Zahid Bozkuş
Department of Computer Science
Turkish-German University
e170503104@stud.tau.edu.tr

1 Introduction

Today, deep learning is used in many different fields to perform different operations. One of the most frequently encountered areas is image classification. In general terms, image classification can be defined as a process that allows images to be separated into different classes according to their visual content. The main purpose of this project, in which image classification will be used, is to create a system to distinguish different dog breeds. For the development phase, the use of Convolutional Neural Networks is the obvious choice.[1]

2 Dataset and Features

The dataset to be used for this project is the "Stanford Dogs Dataset". This dataset contains 20.580 images of dogs, which come from 120 different breeds. There are 150 images per breed on average.[2] Here are some examples from the dataset:

Figure 1: A walker hound

Figure 2: A japanese spaniel

Figure 3: A redbone

As it can be expected, the total number of images may not be sufficient for proper learning. For this reason, methods like data augmentation will be used.[3] Approximately 80% of the dataset will be used for training purposes. The remaining 20% will be used for validation.

3 Preprocessing

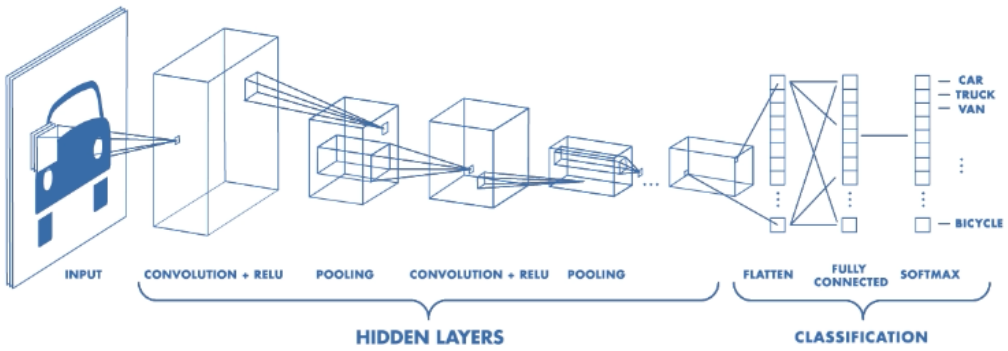
The images in the dataset have different sizes. It is necessary to pre-process them to fit the network architecture that will be used. Hence, they will be resized to 128x128 pixels. The pictures will also be normalized, so they are in similar ranges, which will make the learning process much faster. Like it is mentioned before, we also need to do data augmentation, so we have sufficient data to work with. For this purpose rotated versions of the pictures will also be used.

4 Related Work

There are many similar projects that have been completed with either the specific database that will be used in this project or similar ones. The links to the related works are given in the references section. Mostly what makes a difference are the methods and models that have been used (ResNet, AlexNet etc.), which is another motivational factor for this project. What I will try in this project is to initially build a classification machine by using a specific model. In the following stages, other models will be used for the same database to make a comparison between different models. By doing this comparison, we will have a better idea at the end about what works better for a multiclass classification problem such as this one.

5 Methods and Approach

In the first phase of the project, we are going to build a CNN-Model to train. Like in many similar Convolutional Neural Networks, we are going to have multiple convolutional layers, pooling layers and dense layers. The model can be described with the following image:



A Convolutional Neural Network Example [4]

In the beginning we have an image as the input for our machine. The input will be first processed in the hidden layers. We start with a convolutional layer, where filters are applied to the image. This is where we extract features from the images. After that an activation function is being used, in this case ReLU. The gradient computation is being done with this function. In many CNN projects, the convolutional layer is then followed by a pooling layer, which helps to reduce the size of the input.[5] This layer also helps with the overfitting problem. These first two steps of applying the convolutional and pooling layer can be repeated multiple times. As we apply more layers, the network becomes much deeper. How deep it will become depends on our problem. After we are done with the hidden layers, the actual classification occurs. In this final stage of our model we flatten the inputs and create a set of fully connected layers.[5] As we can see in the image, each parameter is then connected to one another. To get the outputs, we apply the softmax function. This function is espacially to be used when we have a multiclass classification, as we have in our project with 120 different dog breeds. In the end we get the probability that an image belongs to a certain category for every single input.[6]

After we build our model, we need to determine some hyperparameters to train our model with. For the optimization we use the Adam optimization algorithm. Adam is straightforward to implement, computationally efficient, has little memory requirements and some ohter advantages.[7] For the loss function we will use the categorical cross-entropy, as we have a multiclass classification problem.[8]

6 First Model and Results

The first model has been built by using the methods mentioned in the Methods and Approach section. For the initial phase, a pre-trained model with a good accuracy has not been used. In other words, the model started to learn from zero, as the accuracy was nearly zero. With more and more epochs, the loss started to get lower and the accuracy to get higher. The first results after running 10 epochs can be seen in the following image:

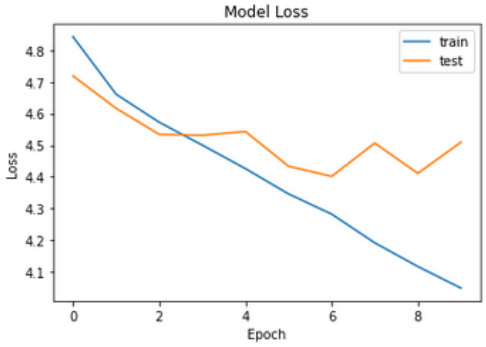


Figure 4: Loss for the Training and Test Set

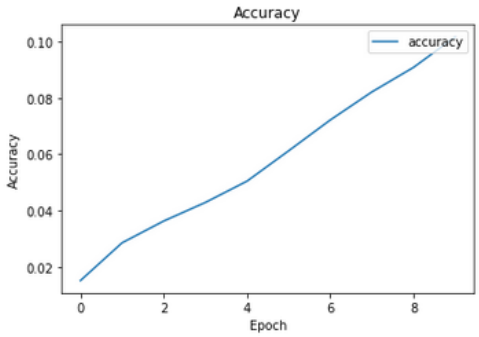


Figure 5: Accuracy of the model

As we can see the loss for both the training and validation set is quite high in the beginning. The loss for the training set goes down from 4.9 to 4.0 in these 10 epochs. At the same time we can observe the the accuracy increases. This can be interpreted as a good sign, but it is obvious that there is potential for improvement.The process could probably be much faster with different parameters or a better architecture.
The loss for the test set is also an issue to be worked on. As the loss gets lower for the training set but not for the test set, we could interpret that the model is overfitting to the training set and should do some changes, like simplifying our model, adding some dropout layers or similar methods to prevent the overfitting.[9]

7 Evaluation

The evaluation methods are to be determined.

GitHub Repository of the Project

https://github.com/chevamikado/ImageClassificationCNN

Related Work and Literature

ImageNet Classification with Deep Convolutional Neural Networks,
http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networ
https://arxiv.org/abs/1312.5402

Some Improvements on Deep Convolutional Neural Network Based Image Classification,
https://arxiv.org/abs/1312.5402

Dog Breed Classification using Deep Learning: a hands-on approach,
https://towardsdatascience.com/dog-breed-classification-hands-on-approach-b5e4f88c333e

How to easily build a Dog Breed Image classification model,
https://medium.com/nanonets/how-to-easily-build-a-dog-breed-image-classification-model-2fd214419cde

References

[1] The Complete Beginner's Guide to Deep Learning: Convolutional Neural Networks and Image Classification,
https://towardsdatascience.com/wtf-is-image-classification-8e78a8235acb

[2] Stanford Dogs Dataset,
http://vision.stanford.edu/aditya86/ImageNetDogs/

[3] The Effectiveness of Data Augmentation in Image Classification using Deep Learning,
https://arxiv.org/abs/1712.04621

[4] MathWorks, Introducing Deep Learning with MATLAB,
https://www.mathworks.com/content/dam/mathworks/tag-team/Objects/d/80879v00_Deep_Learning_ebook.pdf

[5] Image Classification in 10 Minutes with MNIST Dataset,
https://towardsdatascience.com/image-classification-in-10-minutes-with-mnist-dataset-54c35b77a38d

[6] Understand the Softmax Function in Minutes,
https://medium.com/data-science-bootcamp/understand-the-softmax-function-in-minutes-f3a59641e86d

[7] Gentle Introduction to the Adam Optimization Algorithm for Deep Learning,
https://machinelearningmastery.com/adam-optimization-algorithm-for-deep-learning/

[8] Understanding Categorical Cross-Entropy Loss, Binary Cross-Entropy Loss, Softmax Loss, Logistic Loss, Focal Loss and all those confusing names,
https://gombru.github.io/2018/05/23/cross_entropy_loss/

[9] Don’t Overfit! — How to prevent Overfitting in your Deep Learning Models, https://towardsdatascience.com/dont-overfit-how-to-prevent-overfitting-in-your-deep-learning-models-63274e552323