Deep Learning:
A Comparison between Deep Architectures and their Functionalities

Jussi Boersma (s2779153) Wessel van der Rest (s2873672)

University of Groningen, 9712 CP Groningen

Abstract

This paper contains a comparison of 5 different open source deep learning architectures. We compare the basic functionality of these architectures by training them on the image data set CIFAR-10. This data set contains about 60.000 images of 10 different classes on which the architectures will be trained and then tested for their accuracy of being able to identify the correct class per image. Further more the different possible functions used in these architectures will be compared on their effect on the classification accuracy.

1 Introduction

The problem of computer vision is very hard and has been researched for decades. Every year we see improvements in the ability of models to recognize real world objects from images or videos. These improvements are much needed as we implement more and more systems that require good reliable computer vision. Think of self driving cars for example. To stimulate progress and advancements in this field there are annual competitions, such as the best known: ImageNet Large Scale Visual Recognition Challenge (ILSVRC), in which teams compete to have the highest classification score on the given data set. These have been data sets with over 1.3 million images of over 1000 classes. In this paper we want to compare some of the winning architectures and also some lesser known architectures. And also look at the different functions that are used throughout the architectures and compare their effect on the classification score.

2 Methods

We will train and test our architectures on the CIFAR-10 data set. Since it contains 60.000 images but they are of very low resolution this enables our simple computers to be able to download and train the architectures on this data set without needing massive amount of memory or ram. This data set contains 10 classes of images: plane, car, bird, cat, deer, dog, frog, horse, ship and truck. We will use pytorch to create a class and run training and testing for the different networks and be able to change some of their functionalities. The five main architectures we will compare are AlexNet, ResNet, VGG, GoogleNet and ShuffleNet. We will also compare all of these to one simple CNN.

3 Results

In order to be able to compare the pre trained models on the CIFAR-10 dataset we had to make some adaptations. As these models are trained on the ImageNet data set with over 1000 classes. Amongst these 1000 classes are the also 10 parent classes found in the CIFAR-10 data set. In order to compare the model output to the correct CIFAR-10 class output we had to change the names of many output classes from the pre trained models. These models for example are able to differentiate between about 80 dog species, so we changed the output from a specific dog name to simply 'dog', as is required for the CIFAR-10 data set.

Architecture	Classification percentage (%)	Pre trained classification percentage (%)	Run Time (minutes)
AlexNet	55	18	106
VGG16	70	21	823
ResNet	73	23	232
GoogleNet	73	21	631
ShuffleNet	49	14	458
Simple CNN	64	-	7

Table 1: The classification percentage results on the CIFAR-10 data set for the six tested architectures.

Table 1 also shows the run time it took to train the architecture on the Cifar-10 data set on one of our laptops. We ran all training runs on the same laptop to ensure that the times were comparable. It is quite an old laptop so the run times might be higher than they would be on a newer computer.

Architecture	Size (mb)	Parameters
AlexNet	217	60.97 M
VGG16	512	138.36 M
ResNet	97.7	25.56 M
GoogleNet	40	7 M
ShuffleNet	5.3	4 M
Simple CNN	0.8	60.000

Table 2: This shows some of the known features of the tested architectures, their size and the amount of parameters.

For good order we want to show some of the basic features of the tested architectures as to be able to compare good accuracy against the size of the architecture for example.

Activation Function	Classification Percentage (%)
ReLu	55
ELU	62
Softplus	10
Sigmoid	9
LeakyReLu	53
Tanh	62

Table 3: Shows the classification percentage for the AlexNet architecture with the use of different activation function.

We ran different optimizers for the simple CNN because the training time for this network was significantly faster than for either of the 5 main architectures. And this simple CNN scored quite high on classification as can be seen from table 1 thus it is use full to run comparisons of different optimizers on this model and see how it affects performance.

Optimizer	Classification Percentage (%)
SGD with momentum	64
SGD no momentum	45
Adam	62
RMSprop	56

Table 4: Shows the classification percentage for the Simple CNN with different optimizers.

Weight decay values	Classification Percentage (%)
0.9	10
0.1	21
0.01	59
0.001	62

Table 5: Shows the classification percentage for the Simple CNN with weight decay activated at different values.

Our final comparison is made by augmentation of the data. We changed the brightness, contrast, saturation and hue of the images and then used the simple CNN to train on these adjusted images. We only increased the values of the four mentioned features or kept them the same, as the images are very low resolution we figured there is no advantage in decreasing the visibility of possible features. We chose the values for the increase by displaying 20 of the images after an augmentation and visually inspecting whether the classes of the images were still distinguishable. If not we would decrease the amount of the augmentation.

Augmentation method	Classification percentage (%)
no data augmentation	64
Increase brightness	59
Increase contrast	57
Increase saturation	56
Increase hue	58
Increase all	58

Table 6: This shows the classification percentage when the training images are augmented by either increasing brightness, contrast, saturation or hue.

The last comparison we do is to see if we can augment the images by changing either their brightness, contrast, saturation or hue. With the hope that these might make the features more visible and training and classification easier.

4 Discussion

There were large differences between the different activation functions used in the AlexNet architecture. ReLu, ELU, LeakyReLu and Tanh all performed well but Softplus and Sigmoid made the performance no better than random guessing. We would expect Softplus and Sigmoid to have an advantage of a rectifier because it is differentiable in all places or because it saturates less completely, but we empirically find that these expected advantages do not hold. Tanh perfomed good only was much slower in run time since Tanh saturates at very high or very low values, the slope here is very close to zero and thus slows the gradient descent compared to ReLu which slope is never close to zero and thus converges faster. The same is true for the Sigmoid function compared to ReLu.
Looking at table 3 we find that SGD with momentum is the best optimizer in this case, which makes sense because the momentum helps accelerate gradient vectors in the right directions which leads to faster converging.
The results from table 4 can be due to the simple CNN being too simple and having too little weights for weight decay to have a positive effect on the classification percentage. With a decay of 0.9 we see

that the model is just guessing which makes sense since 0.9 almost represents full decay of the weights thus no learning.
The pre trained models perform a lot worse than we would have expected but this could be due to the models being trained on quite high resolution pictures. Since CIFAR-10 consist of very low resolution images it is clearly hard for the models to correctly classify the images even tough they are all from classes it has been trained on.
We can finally conclude that for the CIFAR-10 data set the models that competed in the ImageNet challenge are somewhat overqualified for this task. They take a very long time to train have many millions of parameters thus take in more space on your computer as well (table 7). But on the CIFAR-10 data set thet is no significant improvement on performance between these architectures and a simple CNN. Three of the architectures performed somewhat better than the simple CNN but comparing this to the run time and the size of the models it is not worth using such large architectures for such low resolution images of only 10 classes.
Finally we saw that the augmentation of the images did not have a positive influence on the classification percentage. The images from CIFAR-10 are probably already optimized in this sense and further tampering with them only decreases their features.