Use of Convolution Neural Networks in Scene Recognition
6.819 Miniplaces Project Fall 2017

Team Name: 26ers   Team Code: vdNinu4xEIJBHAuM3gWO

Ekin Karasan
Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge, MA 02139
ekarasan@mit.edu

Dimitris Koutentakis
Massachusetts Institute of Technology
77 Massachusetts Avenue
Cambridge, MA 02139
dkout@mit.edu

Abstract

Convolutional Neural Networks (CNNs) are a powerful tool for learning features in various different types of images and have been used in tasks such as image recognition, object detection and scene recognition. In this paper, we analyze the use of CNNs in the task of scene recognition for a subset of the Places2 dataset consisting of over 10 million images from 400 unique scene categories. We evaluate the performance of two different architectures, AlexNet [1] and ResNet [4], and attempt to improve their performance with various embellishments. We obtain an accuracy of above 70%.

1. Introduction

In the past 5 years, there have been many developments in the use of CNNs in image recognition. The first demonstration of the success of CNNs was with AlexNet [1]. Using their deep convolutional neural network, they were able to win the 2012 ILSVRC (ImageNet Large-Scale Visual Recognition Challenge), a very highly-reputed computer vision challenge. AlexNet not only utilized new techniques such as dropout, local response normalization and data augmentation, but also led to a focus in the field that established CNNs to be the leading class of models for image recognition. Figure 1 shows an overview of AlexNet's architecture.

The following years, several influential and highly respected efforts have been made in order to further improve the accuracy of Convolutional Neural Networks. Such networks include ZFNet [6], which won the ILSVRC the next year using a very similar architecture to AlexNet that was trained for twelve days, and VGGNet [5], that demonstrated the success of using a deep network with many layers.

In even more recent times, the boundaries of image recognition through CNNs have been pushed even further with architectures such as GoogleNet [3] and ResNet [4] that use depths much larger than what had been used before. GoogleNet was the first to demonstrate that stacking layers sequentially on top of each other was not the only approach. They used a structure called an Inception Module, as displayed in Figure 2, in which various portions of the network are happening in parallel. This structure allowed them to utilize very complex and deep architectures while also remaining computationally effi-

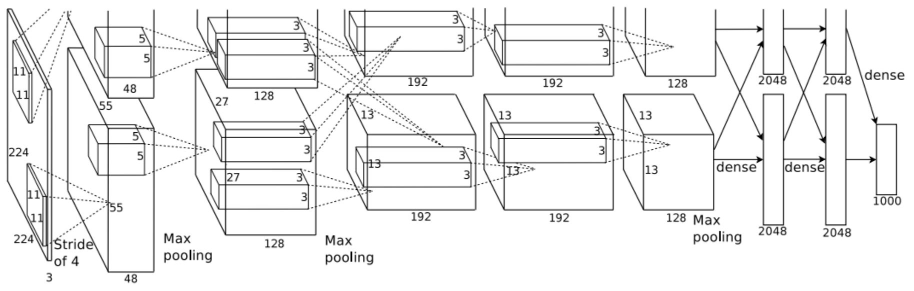

Figure 1. AlexNet architecture. [1]

cient. The ResNet consisted of a 152 layer network architecture and achieved an error rate as low as 3.6% on the ImageNet dataset. This network used a structure called a residual block in which the original input was added to the result of the block as shown in Figure 3.

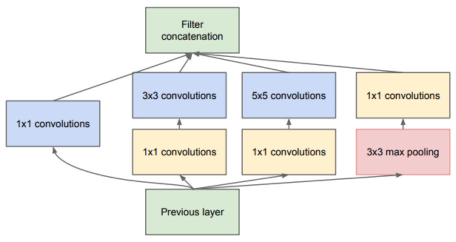

Figure 2. The inception module. [3]

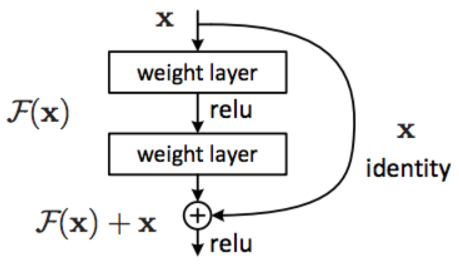

Figure 3. A residual block. [4]

In our work, we explored deep convolutional neural networks for image recognition on the Mini Places Challenge. The goal of the challenge was to identify the scene category depicted in a given photograph. The Mini Places dataset is a subset of the Places2 Dataset [2], and consists of 100,000 training images across 100 different scenes. Due to relatively small size of the training set, we face challenges with over fitting, while still being able to accurately predict the scene of an image.

Our approach will be discussed in further detail in Section 2, where we will explain our methods and thought process behind our approach. In Section 3 we will describe our experiments and our results.

1.1. Division of Work

In order to efficiently produce results, we both worked on all aspects of this project, including research, implementation submission and writeup. We each had an Amazon Web Services (AWS) instance virtual server where we tested our theory which was implemented by making changes in our shared repository code. After each run, we would assess our models and discuss on further ideas to bring down the error.

2. Model & Approach

In this section we discuss the methods used to improve the accuracy of our model in the Mini-places challenge. More specifically, the three main goals we had were:

1. Implementing AlexNet
2. Dealing with over-fitting
3. Implementing ResNet

2.1. Implementing AlexNet

Once we set up the AWS instances, and installed all the necessary dependencies. We started by implementing a working version of the AlexNet code offered in the github repository with the sizes given. That architecture alone gave us a relatively good accuracy of over 60%.
Just by tweaking the parameters such as:

- Learning Rate η
- Batch Normalization decay and
- Dropout factor

We were able to reach accuracies of around 70%. However, even though we trained the model for long times, it was hard to go past that value. We soon realized that our model was over fitting to the relatively small dataset, as the training accuracy was much higher that our validation accuracy.

2.2. Dealing with Over-Fitting

Dealing with over-fitting can be challenging, especially when working with smaller datasets, like the one provided.

In order to deal with the over-fitting, we tried varying the dropout factor and learning rate, but decided to focus on image augmentation. The image augmentation provided with the code is quite basic, resulting in pictures that are very similar to one another. As a consequence it is easy to over-fit after some amount of training. In order to go around that problem, we changed the image augmentation part of the DataLoader python file. We first tried improving it on our own by adding more randomizations and more changes, but quickly decided on going with third party libraries. This is why we installed an open-cv dependent library named imgaug. Imgaug has several built-in functions that allow a large amount of image augmentation in order to avoid over-fitting. We used some of those functions, such as rotating, cropping, changing the image, adding gaussian noise and changing the color of image. This helped us reduce our error.

2.3. Implementing ResNet

Even though the strategies discussed above improved our classification accuracy, we were not able to achieve a very high accuracy with AlexNet and therefore decided to use the deep, complex structure of ResNet to get an increase in our performance. We incorporated the ResNet architecture to our dataset. However, due to its very deep structure, training this network was much slower compared to AlexNet. Therefore, we were unable to tweak the parameters enough times to obtain maximum accuracy.

3. Experiments & Results

3.1. Experimentation Set Up

In order to test our models, we used Amazon Web Services EC2 Instances. We chose the p2.2xlarge AMIs and installed all the necessary dependencies which include:

```
CUDA 8.0
CudNN 6
python3
Tensorflow-gpu
ImgAug
```

Each one of us would make modifications to our code shared on a github repository and then run the modified CNN on their AWS Instance for several epochs. We then tested the test-set accuracy of our model.

Method	Test Set Top 5 Error
AlexNet Base	0.305
AlexNet Optimized	0.29
AlexNet with Image Augmentation	0.28

Table 1. Results for test set accuracy.

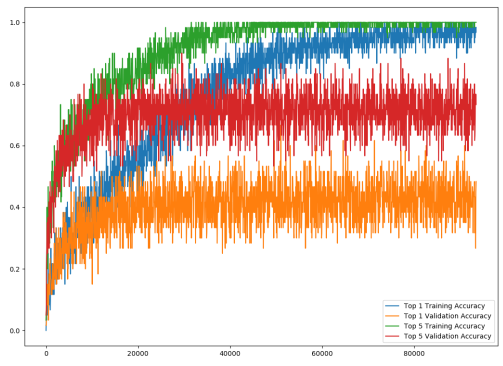

Figure 4. AlexNet with learning rate = 0.0005 batch size = 60, decay=0.9, dropout=0.5

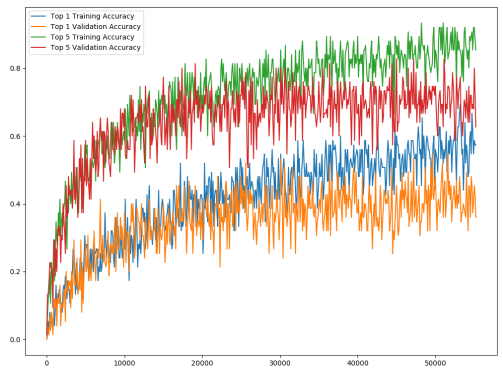

Figure 5. AlexNet with batch size of 75, learning rate of 0.005, dropout=0.3, decay =0.8

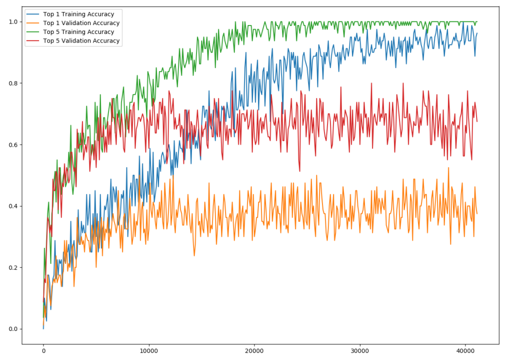

Figure 6. AlexNet with learning rate = 0.001 batch size =80, decay=0.9, dropout=0.7

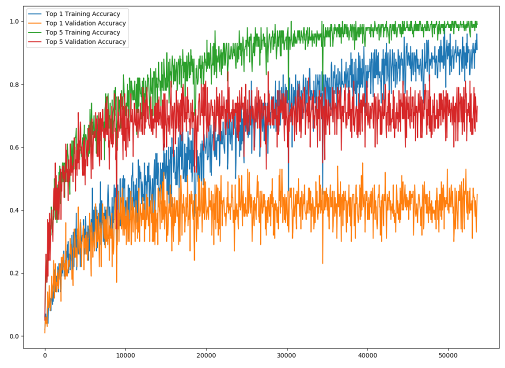

Figure 7. AlexNet with learning rate = 0.005 batch size = 100, decay=0.9, dropout=0.4, improved image augmentation

3.2. Discussion

It is obvious from the results that we received that CNNs can in fact be used to classify image scenes and environments according to a given set of scenes. Given a large enough data set and computing power, training neural network architectures can be a simple process that can then classify all sorts of images.

In our work it was obvious how architectures like AlexNet can be used in such circumstances, but unfortunately, due to factors such as limited computing power and time, we didn't have time to fully explore and optimize the parameters of deeper architecture, such as ResNet and GoogleNet. The research we examined suggests that for this type of model architectures like ResNet should perform better, but we never exceeded our AlexNet accuracy.

References

[1] I. S. A. Krizhevsky and G. E. Hinton. Imagenet classification with deep convolutional neural networks, 2012.

[2] B. Z. et al. Places: An image database for deep scene understanding. *Computer Vision and Pattern Recognition (CVPR)*, 2016.

[3] C. et al. Going deeper with convolutions. *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[4] K. H. et al. Deep residual learning for image recognition, 2015.

[5] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *Computer Vision and Pattern Recognition (CVPR)*, 2015.

[6] M. D. Zeiler and R. Fergusr. Visualizing and understanding convolutional networks. *Computer Vision and Pattern Recognition (CVPR)*, 2013.