CVDL flower project

Sravani Dhara[1] and Dheeraj Tippani[2]

[1]sravani.dhara@st.ovgu.de
[2]sai.tippani@st.ovgu.de

January 2020

Abstract

The purpose of this project is to build a classifier for 'Oxford flowers102' data set. This is achieved using CNN (Computational Neural Networks). This report will give a comprehensive explanation of the approach, methods and problems faced during the process.

Contents

1 Introduction

The goal of this project is to achieve more than 25 percent accuracy in classifying the 'Oxford flowers102' image data set. This section will be followed by methodology, problems faced during implementation, results and interpretations, and future scope.

2 Methodology

The images from the data set are organised into respective class labels, which are in turn organized for training, validation and testing purposes. Computation Neural Networks are known and used for extracting effective features from the image data set. The hyper parameter choices and justification will be given in further sections.

The software tools and packages used for this project are presented below.

- Programming language: Python 3.6
- Integrated Development Environment: Pycharm, Google Collab notebook
- Libraries:
  - TensorFLow: An end-to-end open source machine learning platform.
  - Keras: Keras is an open-source neural-network library written in Python.
  - matplotlib: for visualisation of results and other important.
  - OS: operating system interface.

2.1 Oxford flowers102 Data set

The data set contains 8189 images of flowers belonging to 102 species/classes, with aspect ratio ranging aspect ratio (width/height) from 1 to 1.5. Most of the images of flowers contain cluttered backgrounds, wide range of shapes, colours and postures, in both inter-species and intra-species. The computational challenge of figuring out feature space which can uniquely distinguish across the data set of arbitrary images is extremely difficult, let alone with limited samples for several classes.

From each class 60% of the images are organised in training, 20% in validation and 20% in testing directories. The images are resized, augmented and re-scaled to normalize the images among all classes.

2.2 Hyper parameters

The following are the hyper parameters of the Convolution neural network that is built for the classification. At every stage of the classifier there are certain decisions taken, this section will be explaining and justifying the said decisions and choices.

- Input: Oxford flowers102 Data set contains images of flowers both in raw form and segmented form. The segmented images are not chosen because of the following reasons.
  - Not all flowers are well segmented.

- many flowers are not centered in the images, the data augmentation of segmented images might lose certain important portions.
- The segmented images are simply not yielding as better results as raw images.

- Data Augmentation: Extremely basic data augmentations are performed on data sets so as to increase the size of data set for certain classes with fewer images, without losing important information. This will help in reducing over fitting. Said augmentations include horizontal flipping, width shift (10%), height shift (10%). Augmentation is also necessary because the network runs for around 50 epochs, which could lead to over fitting.

- CNN: Convolutional Neural Networks are shift invariant and space invariant neural networks. Hand crafted features can not effectively represent the images and also cannot uniquely distinguish between classes, because many of the classes have drastically different looking flowers in colour, shape and orientation. Many inter specie flowers look similar and many intra specie flowers look different.

- Layers: The layers of the network are deeper than wider. A wider network will easily over-fit, where as a deeper network is needed for a data set like 'Oxford flowers102'. Deeper networks learn more interesting features from their previous layers.

- Activation function: Relu activation function is used for all the layers except for the final output layer, where Soft-max' activation function is used. In a deep neural network like CNN, back propagation has to reach to the very initial layers, for that relu is the best activation function.

- Dropout: 40% dropout is added at each convolution layer to reduce the over fitting happening during the training. This specific value is tweaked by trial and error.

- Epochs: The number of epochs depend on both the network and data set. We chose to run the network for 50 epochs with drop out of 40%.

3 Classification

3.1 Training

The network is trained with 4874 training images of respective classes. The image size at the first layer is 80 in width and 60 in height, which is around 1/8th the original size of the image. The scaled down version reduces the computation time. The first layer has 12 kernels of size (3, 3) followed by a 2D max-pool layer of size (2, 2).

There are further 4 such convolution layers which are all followed by a max-pooling layer after each of them. Drop out of 40% will help in reducing over fitting after each of these convolution layers. The last convolution layer is flattened to give 1600 inputs for next Dense layer. There are 2 dense layers, one of which is output layer with soft-max as activation function.

Learning rate is dynamic with adam optimiser. A training accuracy of 86% is achieved. Although, the validation accuracy is upto 62% which is indicating over fitting.

3.1.1 CNN Architecture

3.2 Results

Accuracy of the test data is 57%.

3.2.1 cross validation

3.2.2 per class precision

4 Problems faced

- Resnets: test accuracy achieved was 3%
-