**I Computational intelligence**

Indoor scene prediction Computation

Team Members:

1. Ahmed Araby Kamel (Sec1)
2. Ahmed Ragab Saleh (Sec1)
3. Ahmed El-Sayed Salama (Sec1)
4. Mahmoud Helmy Mohamed (Sec5)
5. Adel Mabrouk (sec )

**Data Set Analysis**

At the beginning we tried to explore the data and we get theses statistics, among all the images we have

* Dimensions

Min hight is 150

Min width is 174

Max height is 2607

Max width is 3460

Which tell us that the data have huge variation in it’s dimension so we took this into account when we feed the images into the CNN as resizing the images

Could waste information so we have to find dimension that keep as much info as possible and also give us good training time.

So first we followed the dimensions in the paper of VGG, Alex net ,

227 \* 227 for both of them, then we tried to change theses dimensions and made them 300 \* 300 and it did give a very little enhancement in the validation accuracy so we choose to commit to the dimensions specified in the papers.

* Classes unbalanced

Some classed have number of images greater then other

For example,

airport\_inside class have 487 example

and green house class have 81 example

and this could make the model biased towards the classed with higher number of examples during the training as they will produce high loss value if they ware classified wrong so the model will try to learn their feature in specific instead of learn the general important feature

to solve this problem we did apply data augmentation on the classes that have less number of examples more than others and some times we did just duplicate the images in this classes to give them importance respect to the model, and we used transfer learning.

Also there was a corrupted data and we get rid of them but there number was small so there will not affect our results.

DataPreprocessing

at first

we followed the method was specified in the alexnet paper

as we did

1- center the pixels around mean 0 by subtracting the mean of the pixels

across the whole data set from each pixel, channel wise.

2- normalized the pixels by using this formula

(Pixel\_value – min\_val) /(max\_val – min\_val)

after the centering step as we squash all the values to be in [0 , 1].

But at the end

we did not apply theses methods as feeding the images as it’s get the highest accuracy of ours !!!!.

Data Agumentation

As we said earlier the data was unbalanced with it self and in addition to this

The amount of all the give data is to small to train a CNN with, so we did follow some method to trying to increase our data size with preserving the image in a state that still been valid in it’s respective class, and don’t act as a noise for the Network, we were to use

**horizontal flipping,** rotation, bluring using gaussian noise, add some salt and peper noise but we ended up using only horizontal flipping as the other were acting as noise for the Network and the performance of the Network was better without them, expect **horizontal flipping**.

**Architectures**

We were trying to gain some intuition about how different architecture will behave

So we tried different architecture even if we did now that some of them are too simple to accommodate the problem in hand.

We tried :

1. Simple architecture that was used for Mnist data set (simple one)
2. Alex net (with out transfer learning ).
3. VGG12 (with out transfer learning ).
4. VGG16 (with and without transfer learning )

And the last architecture will be considered as our main architecture that we will discusses our work throw it.

**VGG16\_Architecture**

the idea of the architecture is:

to make the network deep as deep networks provide more activation function layers which mean more ability to fit complex function,

and at the same time it keeps the filters of the convolutional layers simple and small to reduce the time of training.

A picture containing text

Description automatically generated

We did use pretrained weights of the network (transfer learning ) that was trained on ImageNet Data set, as this data set consist of objects and we could say but not for sure that identifying scene could relay on identifying the object in it so this analogy made theses pre-trained weights good for our problem beside that there is a paper \*\*\*\*\*\*\*\*\*\*\* that work in “Places Data set” that classify 1000 different indoor scenes and it did use this Architecture as one of it’s trials.

We also needed to edit the Architecture a bit so we changes the last dense layer ( output layer ) to work with 10 classes instead of 1000

And one of our trials that did not go well was to feed the images to CNN as 300 X 300 so we did change the last MaxPool layer to be

(6 X 6) window with stride = 2 to keep the output for the fully connected layers 7 X 7 X 512 so we still able to initialize them using the pretrained weights.

**hyper parameters**

our hyper parameters settings that we used to achieve 87 % accuracy on Kaggle :

Learning rate =0.001

Batch\_size = 256

Activation function for conv layers and FC layers is “RELU”

Dropout = 0.7

L2 regularization = 0.005

Validation set was 1% of all the data

Number of Epochs = 60

Optimizer is Momentum

Momentum = 0.9

As we used transfer learning we initialized all the layers using the pretrained weights and we did freeze all the layers

expect last conv layer and the fully connected layers.

From the table below test accuracy make it evident that our model is overfitting and this was evident to us also during the training and we tried a lot of methods to reduce this overfitting but it did just help a little with the validation accuracy from 80% to 93% but not with the test accuracy

We applied dropout = 0.7 and L2 regularization and we also

tried to reduce the complexity of the network but reducing the network complexity give us underfitting.

**Results**

|  |  |
| --- | --- |
| Accuracy on the train data set | 100% |
| Accuracy on the validation Data set | 93% |
| Accuracy on the test data set on Kaggle | 87.046% |