Image Classification on Caltech101 Dataset

Karthik Kadajji, Kartik Prabhu, Ravi Theja Gunti

18th June 2019

Abstract

This report details the work done by our team on the task of Image classification on Caltech101 data set. We considered Deep Convolutional Neural Network with Confusion Matrix for implementation and evaluation of the dataset. The dataset is highly unbalanced and also contains class imbalance problem. The goal is to be able to find suitable methods to classify the images.

1 Introduction

The motivation of the project is to reach accuracy of above 30% in more than 50% of class label and display a confusion matrix for the same. Section 2 gives an overview of the dataset and a few observations. In section 3, we give the details of the approach and ideas we had during implementation. Section 4 has the results including visualization of certain layers and confusion matrix. We conclude the project in section 5.

2 Data Set:

Caltech101 is a dataset with pictures of objects belonging to 101 categories. Each category has 40 to 800 images. Most categories have about 50 images. The dataset was prepared in September 2003 by Fei-Fei Li, Marco Andreetto, and Marc 'Aurelio Ranzato. The size of each image is approximately 300 x 200 pixels. Below are a few observations from dataset.

2.1 Opportunities

Following are the observations that model can take advantage of:

- Uniform size and presentation: Most of the images within each category are uniform in size and in the relative position of interest objects. Hence, there is no need to crop or scale images before use.
- Low level of clutter/occlusion: Algorithms usually store features unique to the object. However, most images taken have varying degrees of background clutter, which means algorithms may build incorrectly.
- Fixed point of view for most of the classes

2.2 Challenges

- Class imbalance: A few classes do not have enough images to train a model.
- A few classes have images that are very similar (Human vs Human faces, Bonsai vs Brain, Butterfly vs dragonfly)
- A few images are rotated (Example:nautilus) and scaled(Least height = 92 Least width = 80, Max height = 3999 Max width = 3481)
- A few images have multiple view points (example: Brain)
- The class "Background-Google" is a mixture of different images which do not contribute to the precision of the model.

2.3 Summary

Most of the scenarios are in favor of classification, and some of the challenges can be overcome by minor pre-processing of images.

- Number of Instances: 9146
- Number of labels : 101

3 Implementation

Initially to get a gist about the features we tried extracting SURF(Speeded-Up Robust Features). And using BagOfvisuals extracted from 7748 images, 3,83,62,468 features. Finally this was reduced to a feature vector(7748 X 500) with clustering. This gave us an approximate accuracy of 40 percent for a 1-fold hold out evaluation.

Coming to our core implementation with CNN architecture, the network is as shown in the below figure.

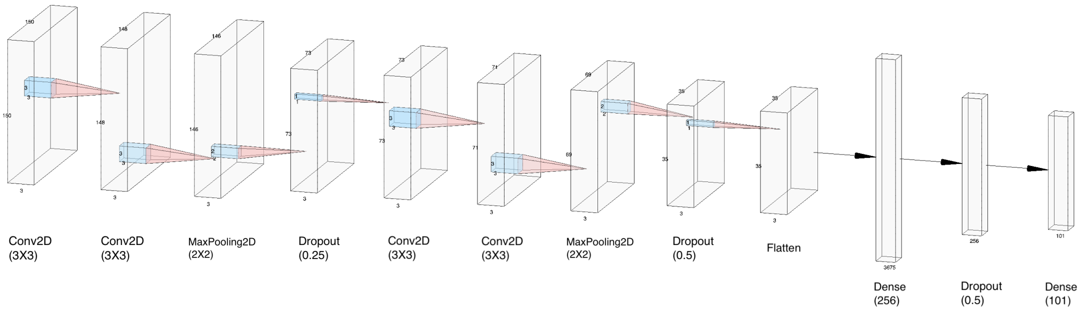


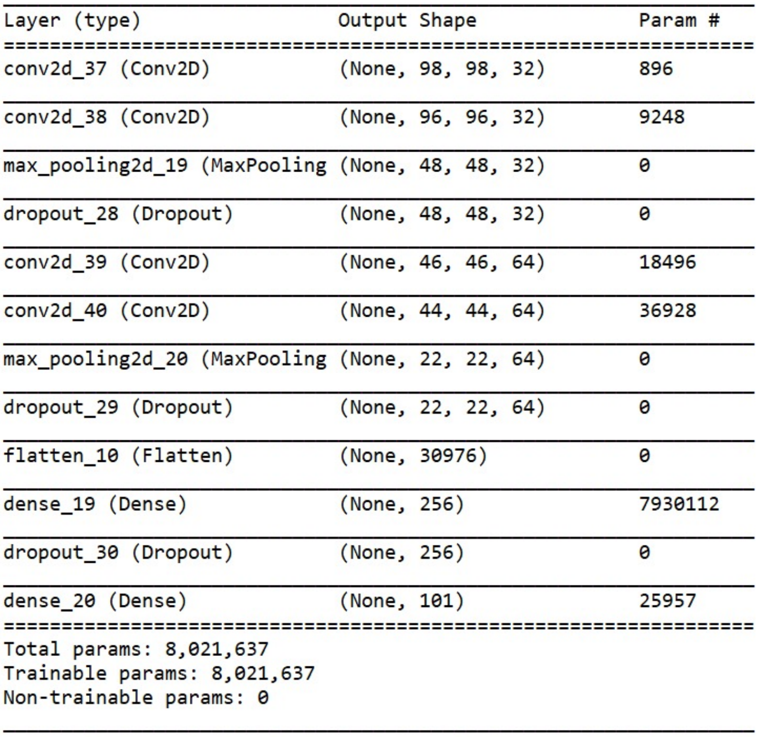
Figure 1: CNN architecture

3.1 Defining CNN model

The CNN model is composed of convolutional layers followed by maxpooling and dropout. A 100X100 image will be processed and the flatten and dense output will give us a 101 classification bits. A total of 9,021,637 trainable parameters are obtained as indicated in the Model figure. Average size of images in the dataset was found to be 244.64 X 301.65. So we even tried input with size 150X150, which again gave similar accuracy. SGD is the chosen optimizer with categorical_crossentropy.

The data set is split into 10 folds. The model is trained with 10 fold cross validation. We have build 10 models one for each validation fold. Below is the summary of the model.

3.2 Model Summary



```
Layer (type)                 Output Shape              Param #
=================================================================
conv2d_37 (Conv2D)           (None, 98, 98, 32)        896
_________________________________________________________________
conv2d_38 (Conv2D)           (None, 96, 96, 32)        9248
_________________________________________________________________
max_pooling2d_19 (MaxPooling (None, 48, 48, 32)        0
_________________________________________________________________
dropout_28 (Dropout)         (None, 48, 48, 32)        0
_________________________________________________________________
conv2d_39 (Conv2D)           (None, 46, 46, 64)        18496
_________________________________________________________________
conv2d_40 (Conv2D)           (None, 44, 44, 64)        36928
_________________________________________________________________
max_pooling2d_20 (MaxPooling (None, 22, 22, 64)        0
_________________________________________________________________
dropout_29 (Dropout)         (None, 22, 22, 64)        0
_________________________________________________________________
flatten_10 (Flatten)         (None, 30976)             0
_________________________________________________________________
dense_19 (Dense)             (None, 256)               7930112
_________________________________________________________________
dropout_30 (Dropout)         (None, 256)               0
_________________________________________________________________
dense_20 (Dense)             (None, 101)               25957
=================================================================
Total params: 8,021,637
Trainable params: 8,021,637
Non-trainable params: 0
_________________________________________________________________
```

Figure 2: Model Summary

3.3 Data Augmentation

- rescale=1./255; The dataset is made up of different size images, and re-scaling can help in matching the images.
- shear range=0.2; Shear mapping is to try and remove some image stretches.
- zoom range=0.2; Since most of the images have same view point, zoom range need not be high.
- horizontal flip=True; Even though flipping is not the best option in some scenarios like Medical imaging, in case of Caltech101 we didn't find any such class where there would be an inverse impact.
- brightness = 0.2; No significant changes were seen by changes the intensity of the image.

4 Results

In this section the accuracy and loss during the training are described along with a few visualizations of layers followed by the confusion matrix and recall table.

4.1 Accuracy and Loss

The figues below show the accuracy and loss during training and validation. Accuracy and loss tend to get stable after approximately 15 epochs. We can also see the loss being the least for validation curve, which might be due to over-fitting. 10-fold cross validation has been implemented.

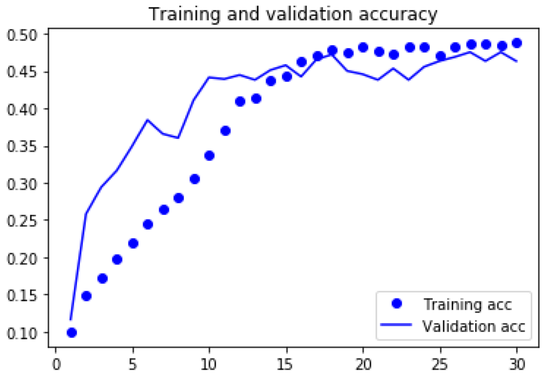


Figure 3

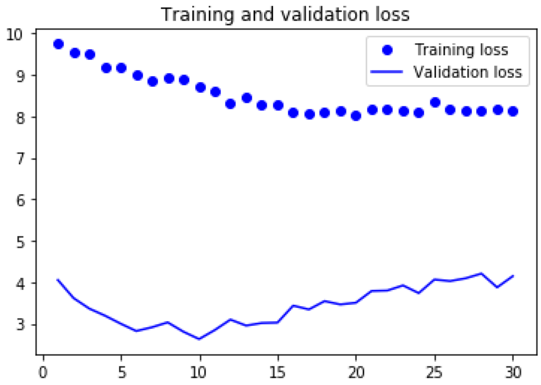


Figure 4

4.2 Visualize CNN Layers

Since class 'Face' was one of our top classes to show high recall, we visualized the output after each layer. The figure visualization was taken after the 8th layer in our CNN. From the figure below, many of the characteristics of the face are detected in different activation, and thus we get higher recall.

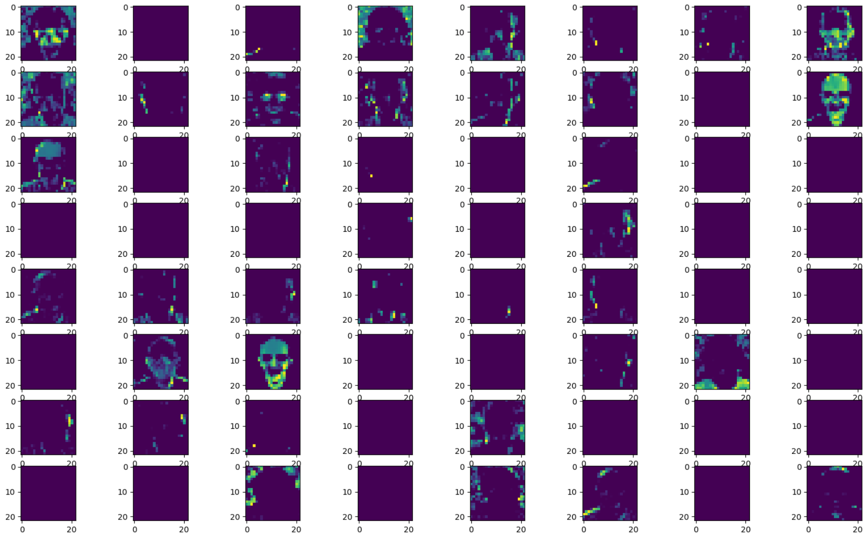


Figure 5: Visualisation

4.3 Confusion Matrix Plot

Accuracy [(True Positives + True Negatives)/total] gives the measure of classifier as a whole. To understand how the model performed per class, Recall [(True positive/actual yes)] is used as evaluation measure.

- Classes with more than 300 images in the dataset produced very high recall. Airplanes, car_side, Motorbikes, Faces gave an recall above 90.
- Classes having fewer than 50 images gave less recall. Cannon, ant, crocodile_head, platypus, mayfly,nautilus were the difficult ones to predict and gave very low recall.

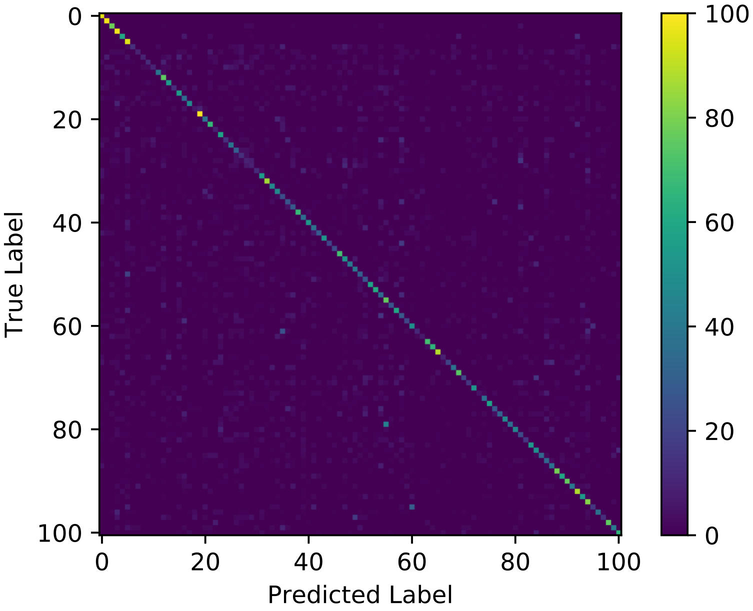


Figure 6: Confusion Matrix Plot

4.4 Standard Deviation of each label for 10 folds

To understand the consistency of each model, a plot is made with each point representing standard deviation for a particular class for 10 folds. For an ideal classifier, the values should be close to zero. Our model gives

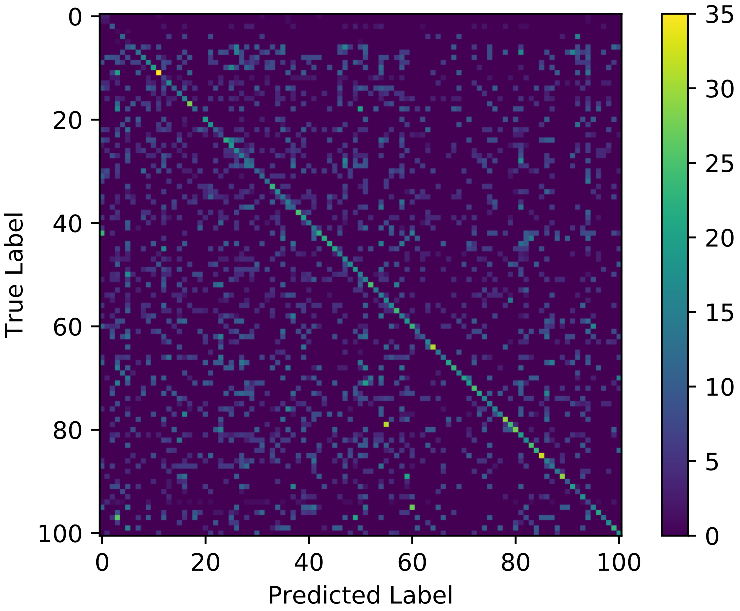


Figure 7: Standard Deviation of each label for 10 folds

Figure 8 shows the histogram of standard deviation of recall values for 10 folds. The mean value around 16.

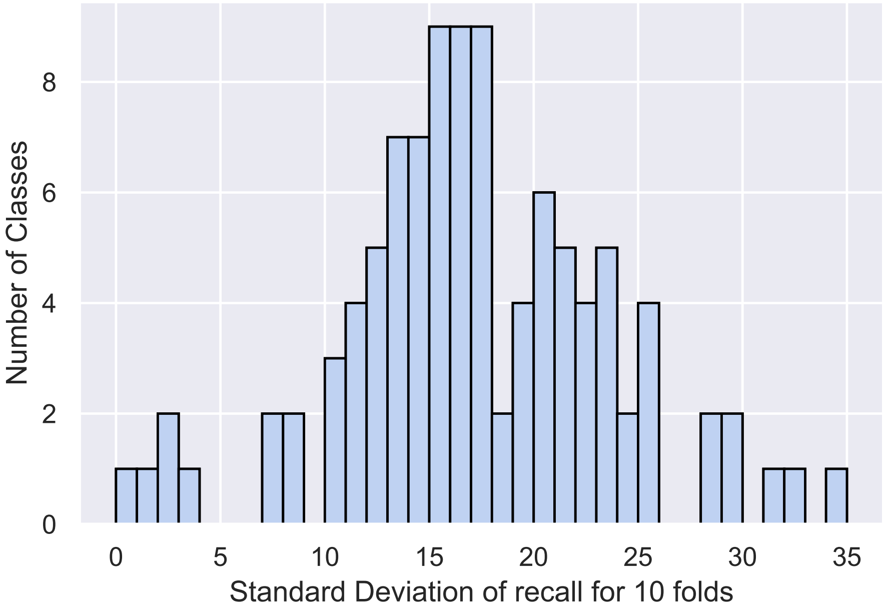
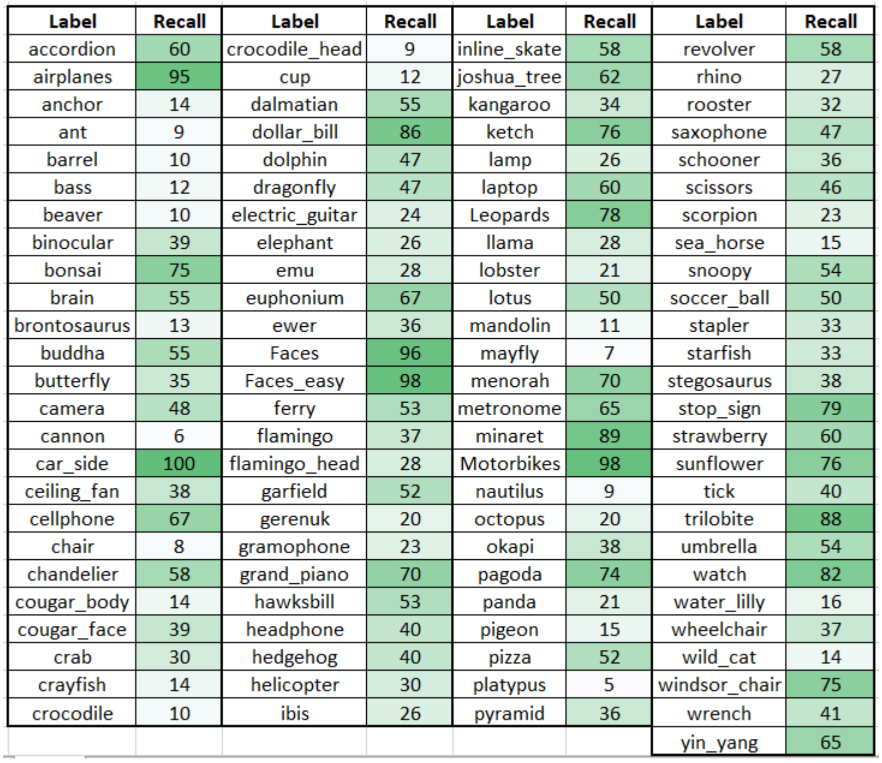


Figure 8: Standard Deviation of each label for 10 folds

4.5 Recall Table

The table shows average recall for each class over 10 folds



Label	Recall	Label	Recall	Label	Recall	Label	Recall
accordion	60	crocodile_head	9	inline_skate	58	revolver	58
airplanes	95	cup	12	joshua_tree	62	rhino	27
anchor	14	dalmatian	55	kangaroo	34	rooster	32
ant	9	dollar_bill	86	ketch	76	saxophone	47
barrel	10	dolphin	47	lamp	26	schooner	36
bass	12	dragonfly	47	laptop	60	scissors	46
beaver	10	electric_guitar	24	Leopards	78	scorpion	23
binocular	39	elephant	26	llama	28	sea_horse	15
bonsai	75	emu	28	lobster	21	snoopy	54
brain	55	euphonium	67	lotus	50	soccer_ball	50
brontosaurus	13	ewer	36	mandolin	11	stapler	33
buddha	55	Faces	96	mayfly	7	starfish	33
butterfly	35	Faces_easy	98	menorah	70	stegosaurus	38
camera	48	ferry	53	metronome	65	stop_sign	79
cannon	6	flamingo	37	minaret	89	strawberry	60
car_side	100	flamingo_head	28	Motorbikes	98	sunflower	76
ceiling_fan	38	garfield	52	nautilus	9	tick	40
cellphone	67	gerenuk	20	octopus	20	trilobite	88
chair	8	gramophone	23	okapi	38	umbrella	54
chandelier	58	grand_piano	70	pagoda	74	watch	82
cougar_body	14	hawksbill	53	panda	21	water_lilly	16
cougar_face	39	headphone	40	pigeon	15	wheelchair	37
crab	30	hedgehog	40	pizza	52	wild_cat	14
crayfish	14	helicopter	30	platypus	5	windsor_chair	75
crocodile	10	ibis	26	pyramid	36	wrench	41
						yin_yang	65

Figure 9: Recall per Class

5 Conclusion

For image classification on Caltech101 dataset, we are able to classify 63 classes with accuracy more than 30%. Overall recall for the model is 44.92%
The class 'Chair' has the least recall (4%) while the class 'Car_side' has maximum recall (100%) This could be because of extremes of instances available for each class. To further improve the model, for classes with very limited number of instances, one approach could be to generate more instances particular to the class using Generative Adversarial Network.