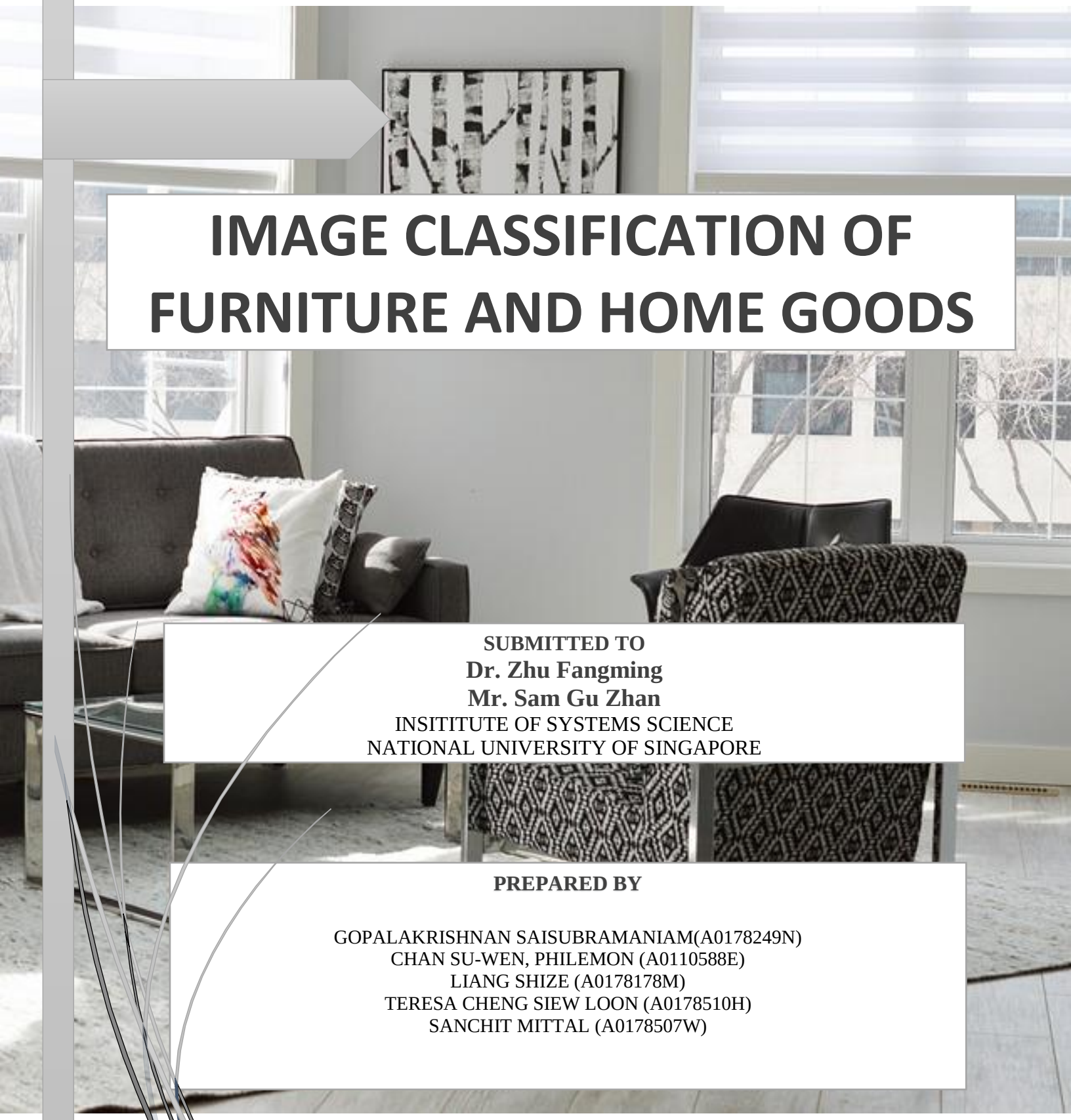


IMAGE CLASSIFICATION OF FURNITURE AND HOME GOODS

SUBMITTED TO

Dr. Zhu Fangming
Mr. Sam Gu Zhan
INSITITUTE OF SYSTEMS SCIENCE
NATIONAL UNIVERSITY OF SINGAPORE

PREPARED BY

GOPALAKRISHNAN SAISUBRAMANIAM(A0178249N)
CHAN SU-WEN, PHILEMON (A0110588E)
LIANG SHIZE (A0178178M)
TERESA CHENG SIEW LOON (A0178510H)
SANCHIT MITTAL (A0178507W)

Table of Contents

10. BUSINESS UNDERSTANDING

10.1. PROBLEM FORMULATION

In the past, machines have been unable to match human performance in specific tasks like image recognition and object detection. Due to recent availability of data and computation power, studies revolving deep neural networks have shown effective and promising results in image recognition and adoption rates have increased. With the advent of convolutional neural nets, machines are now able to exceed human-level performance in visual recognition[1].

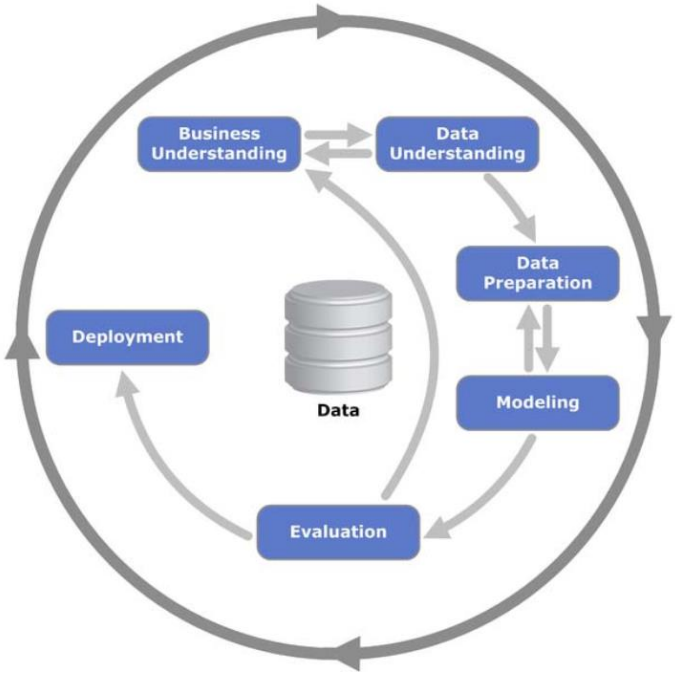
With the proliferation of e-commerce, furniture companies are turning to online platforms to advertise and sell their products. A system that automatically classifies a furniture product according to its type will be immensely helpful for buyers who seek furniture pieces that are suited to their needs.

10.2. DATA MINING GOALS

To implement an automatic image classification system through understanding, experimentation of different neural network architectures and evaluating their performance.

10.3. PROJECT PLAN

We follow the CRISP-DM Model for achieving the data mining and business goals for this project.



[Figure 1: CRISP-DM phases]

The following steps have been carried out to discover knowledge from the data:

- Business Understanding
- Data Understanding
- Data Preparation
- Modeling
- Evaluation

We discuss these phases in detail from our project point of view.

11. DATA UNDERSTANDING

11.1. DATA SOURCE

The dataset under consideration is Furniture Image classification provided by Kaggle [2]

11.2. DATA DESCRIPTION

The image dataset comprises 128 different categories of furniture and home goods with more than 214k samples.

The dataset format as shown below:

```
{
        "images" : [image],
        "annotations" : [annotation],
        image{
        "image_id" : int,
        "url": [string]
        },
        annotation{
        "image_id" : int,
        "label_id" : int
        }
}
```

For this assignment, we select the following 8 categories and overall 5339 sample numbers.

S.No.	Category	Image samples
1	Bed	508
2	Ceiling light	344
3	Chair	1577
4	Cupboard	629
5	Lamp	557
6	Rack	248
7	Sofa	529
8	Table	947

[Table 1: Dataset class distribution]

12. DATA PREPARATION

3.1 IMAGE RETRIEVAL

The original FGVC5 challenge provided a json file with the field description as shown in the previous section. The images were retrieved by sending request to the urls provided and downloading the images from response. We used *urllib.request* package from python for this purpose. We decided to restrict the samples and classes to focus more on the techniques to be applied rather than the challenge itself.

3.2 Image Resizing

3.2.1 Image Resizing for Capsule Network

To maintain consistency across all images, we resized them to 32 by 32 pixels. This was done as the neural network model was run on a laptop machine. When an image is considered as a matrix, each cell contains a color value from 0 to 255 per channel. The selected dataset dimensions – 5339 by 32 by 32 by 3 [5339 images, 32 by 32 dimensions per image with 3 (RGB) channels] The package used for resizing was *PIL*. (Python Imaging Library).

3.2.2 Image Resizing for Custom Convolutional Network (CNN)

Similarly, prior to passing images to a convolutional neural network, we resize the images to 128 by 128 pixels. (128 to make the image bigger but at lesser expense time) Thus the selected dataset dimensions – 5339 by 128 by 128 by 3.

3.2.3 Image Resizing for Residual Network

Finally, prior to passing images to a residual network architecture, we resize the images to 224 by 224 pixels (as ResNet50 only accepts that size) and each cell contains a value from 0 to 255 per channel. The selected dataset dimensions – 5339 by 224 by 224 by 3.

3.3 Image Augmentation

For image augmentation of this dataset, we used the library from https://github.com/aleju/imgaug. Image augmentation creates similar images from the original image and stresses more importance on the model to learn unique characteristics of each class rather than simply memorize the data. It also induces noise and helps avoid overfitting during training.

The following augmentation techniques were used, implemented using *imgaug* package in python.

- Horizontal flips
- Random crops
- Gaussian blur
- Contrast normalization
- Additive Gaussian Noise
- Brightness modification by a multiplier filter
- Affine transformations such as scaling, translation and rotation.

After augmentation, the overall dataset size increased by six times (1 original + 5 images from random combination of above augmentation techniques)

3.4 Image Vectors

After augmentation, three data files are generated by converting each image into a vector of N by N by 3 dimensions, where N = 32, 128, 224 for the three chosen model architectures. Two fields are created per record – class label (0 to 7) and the image input vector of N by N by 3 dimensions. For an image of 224 by 224 in the case of ResNet-50, the dimension size is 150528. For the custom CNN, the dimensions are 49152. For capsule network, the image vector has dimensions of 3072.

The class labels were transformed from numbers to one-hot encoded vectors.

13. MODELLING

4.1 MODELLING TECHNIQUES

The following model architectures are considered for the experiments –

3. Custom Convolutional Neural Net Architecture
4. Residual Network via Transfer Learning [4]
5. Capsule Network [5]

4.2 GENERATE TEST DESIGN

We split the dataset into train and testing – 70% train and 30% test. From the training dataset, we further split it into train and validation in the ratio 80%-20% The dataset split was performed by the stratified sampling approach to maintain the distribution.

Owing to the high runtime complexity of neural network training, we consider using the hold-out validation set and do not perform k-fold cross validation.

The distribution after the split is as follows:

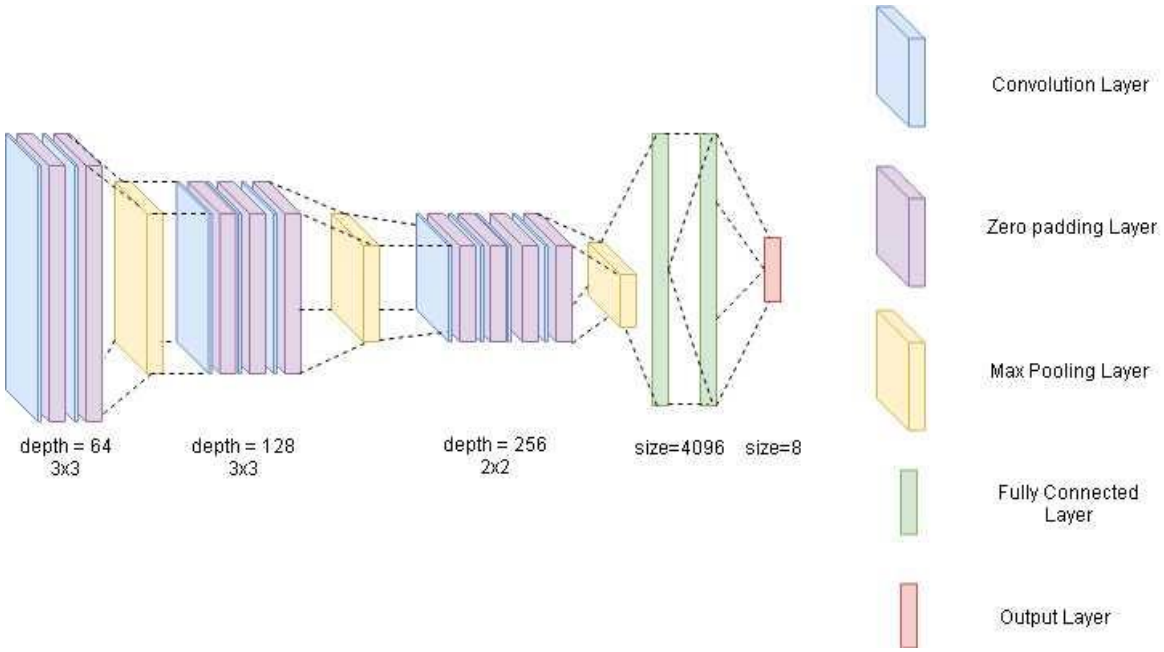
S.No.	Category	Image samples (after augmentation)	Training	Validation	Testing
1	Bed	3048	1707	427	914
2	Ceiling light	2064	1156	289	619
3	Chair	9462	5298	1325	2839
4	Cupboard	3774	2113	528	1132
5	Lamp	3342	1871	468	1003
6	Rack	1488	834	208	446
7	Sofa	3174	1777	444	952
8	Table	5682	3182	795	1705

[Table 2: Image Sample Sizes]

4.3 BUILD MODEL

4.3.1 Custom Convolutional Neural Network

We used Keras neural network Application Programming Interface (API) for the custom Convolutional neural network. Our initial custom CNN architecture was very simple and did not give satisfactory results, so we built a new 'deeper' network taking inspiration from VGGNet 19[3]. The schematic of the above-mentioned neural network architecture is as follows.



[Figure 2: Custom convolutional neural network architecture]

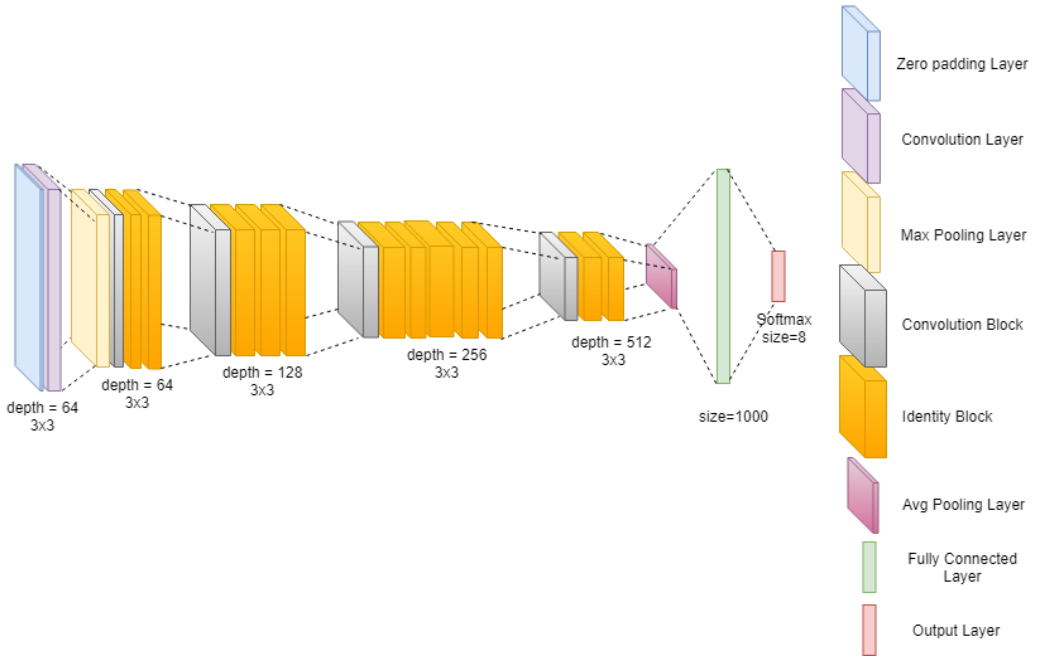
For the custom convolutional neural network architecture, the design was based on stacking 2 convolution layers and a max pooling layer, stacking 3 convolution layers followed by a max pooling layer, and finally 4 convolution layers followed by a max pooling layer. At the end of the neural network, we flatten and introduce two dense layers. The output layer has only 8 nodes (corresponds to number of classes) and softmax activation is applied to obtain the final scores between {0,1}.

4.3.1 Residual Neural Network

A quick way to train new images is to consider the architecture and weights of an existing network and change the last few layers to suit our dataset. This technique of storing knowledge while solving one problem and applying it to a different but similar problem is termed Transfer Learning. TensorFlow provides this functionality to retrain image classification models which were pretrained on ImageNet and Pascal-VOC datasets.

As part of this section, we train the furniture images on the Residual Network-50 model pretrained on ImageNet dataset. For the special case of ResNet, we scaled up the images from 128 by 128 to 224 by 224 due to the specification of the input layer.

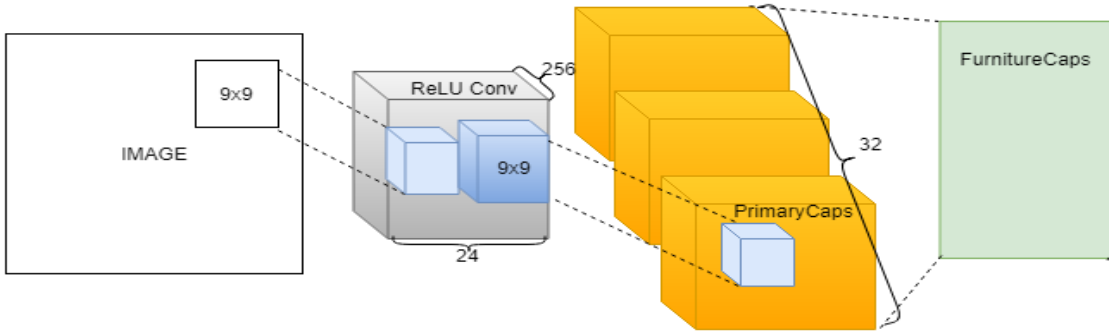
The ResNet weights was trained further on Keras with TensorFlow backend for the furniture images. The last two layers were removed from the original and re-trained by introducing a dense layer followed by an output layer (n=8) with softmax activation. The architecture is as shown in the figure below.



[Figure 3: Residual Neural Network Architecture]

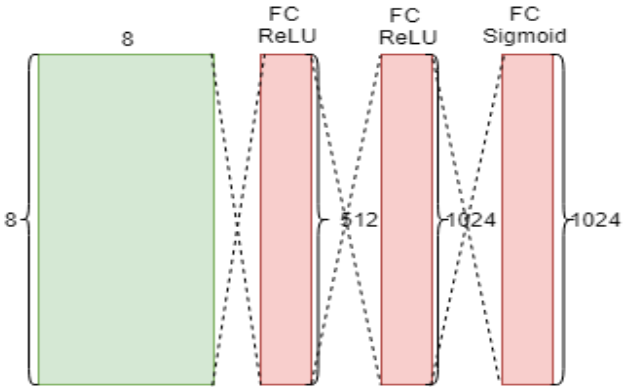
4.3.1 Capsule Network

The capsule network takes in images of 32 by 32-pixel dimensions and the network consists of encoder and decoder components. The encoder component comprises convolution, primary caps and furniture caps layers. The schematic for encoder architecture is as follows.



[Figure 4: Capsule Network Encoder]

The encoder part of the network takes in input using a 32 by 32 image and the convolutional layer has 256 kernels of shape 9 by 9 by 1 with a ReLU activation. In the primary caps, there are 32 primary capsules to process the features of the convolutional layer. The decoder components comprise 3 fully connected layers consisting of 2 rectified linear units (ReLU) and 1 Sigmoid layer respectively.



[Figure 5: Capsule Network Decoder]

4.4 Handling Class Imbalance

We found three ways of handling class imbalance –

- Upsamping the smaller samples per class by duplication or SMOTE
- Downsampling the higher samples per class
- Using class weights during model training

We decided to go with class weights to maintain the same dataset distribution and get the feeling of working on the real world population. To explain further, considering class *Rack* has the least number of samples amounting to 1488, the ratio with respect to every other class was: 2.0:1.4:6.3:2.5:2.2:1:2.1:3.8. Then we compute the class weights by taking individual reciprocal. i.e. 1/x -> {0.5, 0.71, 0.15, 0.4, 0.45, 1, 0.47, 0.26}.

4.5 Ensembles

There are multiple ways of performing ensemble. Three of them are –

- Simple majority voting across classifiers
- Weighted Sum
- Stacking by means of a meta-classifier

4.6 Transfer Learning

A quick way to train new images is to consider the architecture and weights of an existing network and change the last few layers to suit our dataset. This technique of storing knowledge while solving one problem and applying it to a different but similar problem is termed *Transfer* Learning. Tensorflow provides this functionality to retrain image classification models which were pretrained on ImageNet and Pascal-VOC datasets.

As part of this section, we train the furniture images on the ResNet model pretrained on ImageNet dataset. For the special case of ResNet we scaled up the images from 128*128 to 224*224 due to the specification of the input layer.

4.6 ASSESS MODEL PERFORMANCE

The following metrics are considered for assessing the performance during training (and validation) –

- Accuracy @ TOP-1, TOP-3, TOP-5
- Loss
- Mean Average Error (MAE) and Mean Square Error (MSE)

The following metrics are considered for assessing the performance during testing –

- Accuracy @ TOP-1, TOP-2, TOP-3, TOP-4, TOP-5

14. EVALUATION

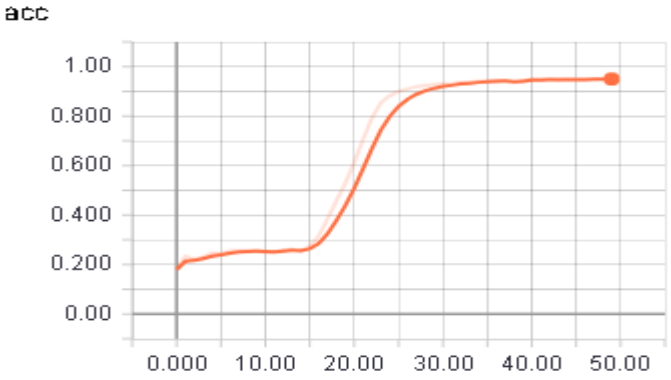
In this section, we present the results of custom convolutional network (CNN), residual network (ResNet-50) and Capsule Neural Network.

5.1 RESULTS OF EXPERIMENTS

5.1.1 Custom Convolutional Neural Network Results

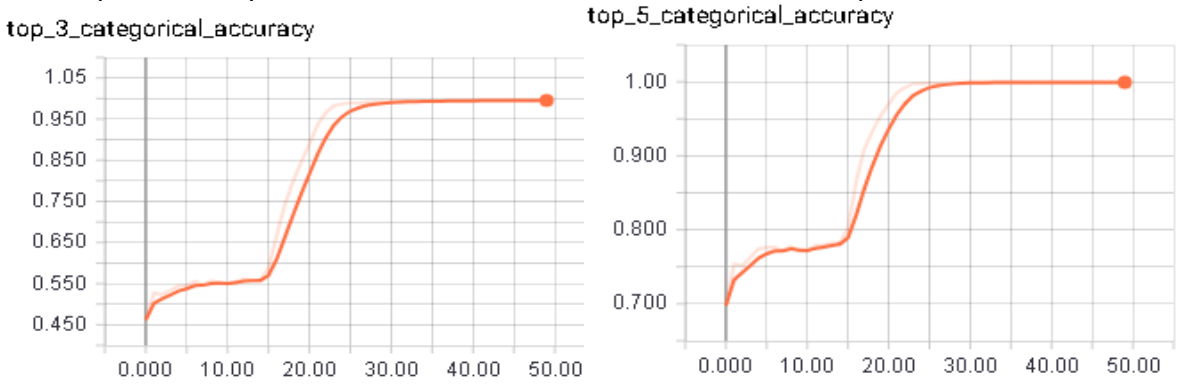
5.1.1.1 Custom CNN Training Accuracy

After training the model, we examined the accuracy values for the custom convolutional neural network. The Top-1, Top-3 and Top-5 accuracy graphs are as follows.



[Figure 6: Top-1 Model Accuracy for Custom CNN]

Initially, training the custom CNN results in an accuracy of 0.250 up to epoch 15. Subsequently, the Top-1 accuracy for the custom CNN stabilizes at 0.950 after epoch 40.

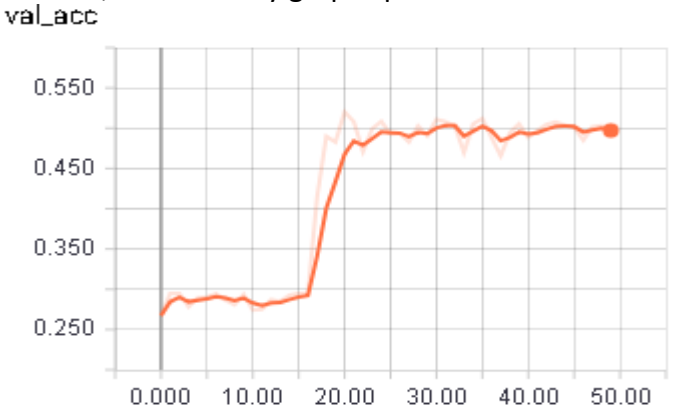


[Figure 7: LEFT. Top-3 Categories Accuracy. RIGHT. Top-5 Categories Train Accuracy for Custom Convolutional Neural Network]

The model accuracy graphs for Top-3 and Top-5 categories show a point of inflation at around 0.550 and 0.775 respectively. As the number of epochs increases beyond 20, it is likely that the custom convolutional neural network is overfitting the data as seen from the accuracy value of 1.00. By increasing the number of epochs, we observed an increase in model accuracy for all top-1, top-3 and top-5 categories. The overfitting could also be since we did not increase the dropout p value and fine tune the regularization parameter enough.

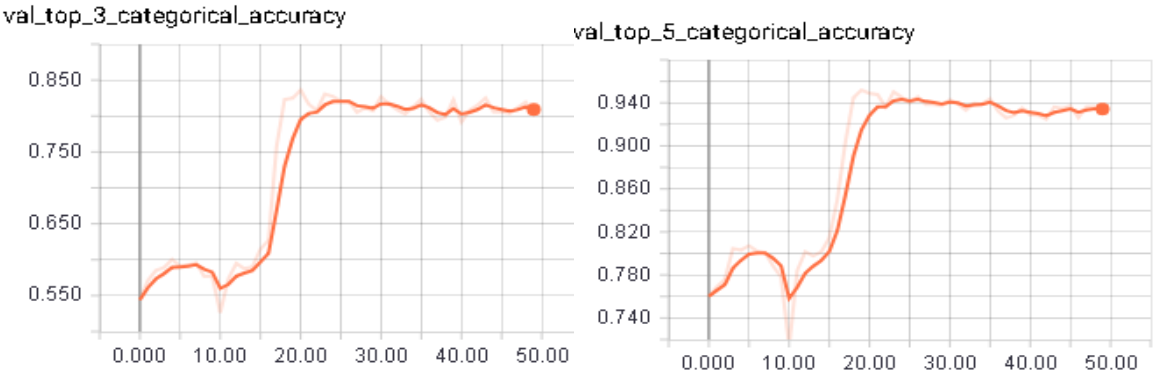
5.1.1.2 Custom CNN Validation Accuracy

For the validation dataset, the accuracy graphs plotted with TensorBoard are as follows.



[Figure 8: Top-1 Model Accuracy for Custom Convolutional Neural Network (CNN)]

By comparing with the accuracies obtained in the validation with the training dataset, the Top-1 accuracy for the validation set stabilizes at around 0.500.

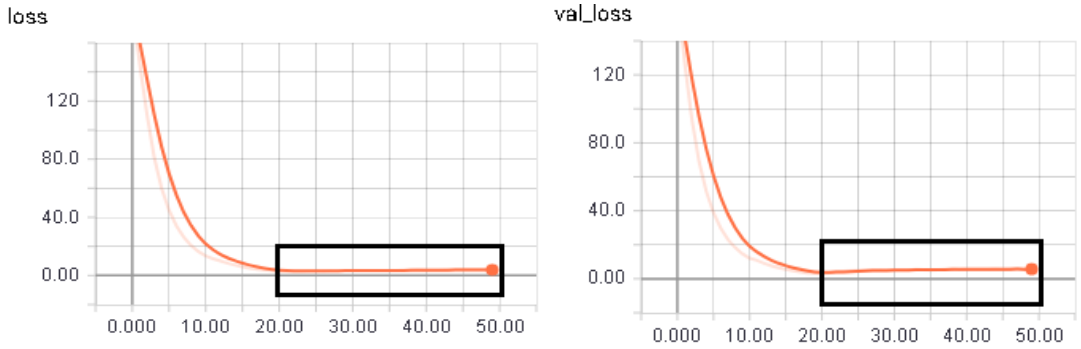


[Figure 9: LEFT. Top-3 Categories Accuracy. RIGHT. Top-5 Categories Validation Accuracy for Custom Convolutional Neural Network (CNN)]

Aside from Top-1 accuracy being lower for the case of validation data, the top-3 and top-5 accuracy are also lower for the validation set when compared to the training set. Next, we examine the train loss and validation loss associated with the custom convolutional neural network (CNN) model.

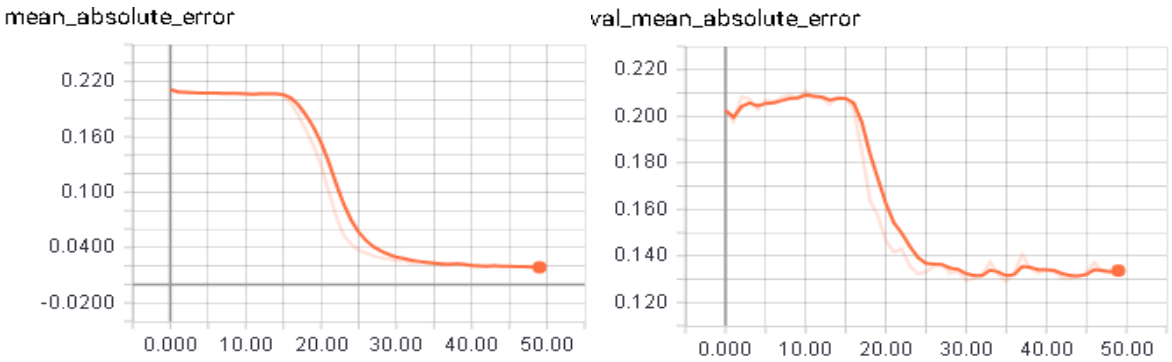
5.1.1.3 Custom CNN Train Loss versus Validation Loss

By examining the train and validation loss associated with the custom CNN, we realised that the validation loss is higher than the training loss. This can be seen by the region within the bounding box of the graph is higher for the validation loss as compared to the training loss. This supports the statement that the custom convolutional neural network overfits the furniture classification train dataset.



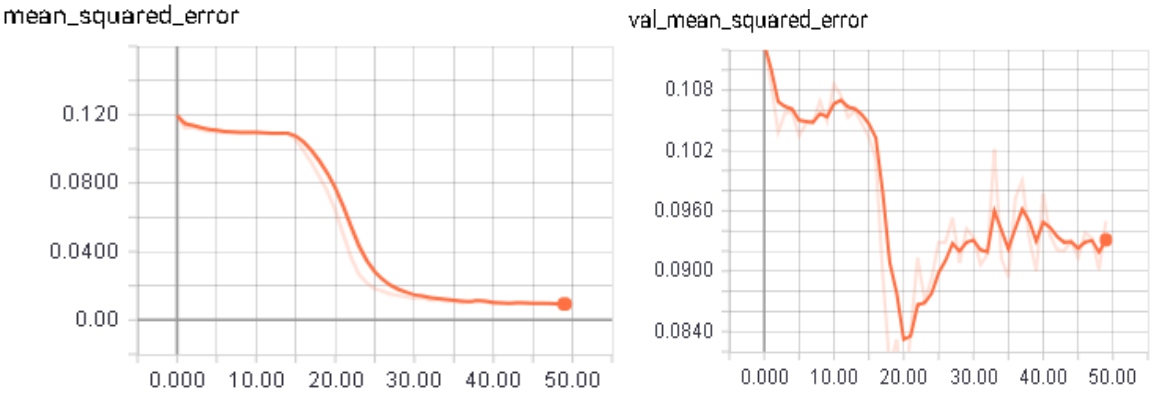
[Figure 10: LEFT. Training Loss. RIGHT. Validation loss for Custom Convolutional Neural Network (CNN)]

5.1.1.4 Custom CNN Mean Absolute Error (MAE) for Train and Validation



[Figure 11: LEFT. Training MAE. RIGHT. Validation MAE for Custom Convolutional Neural Network (CNN)]

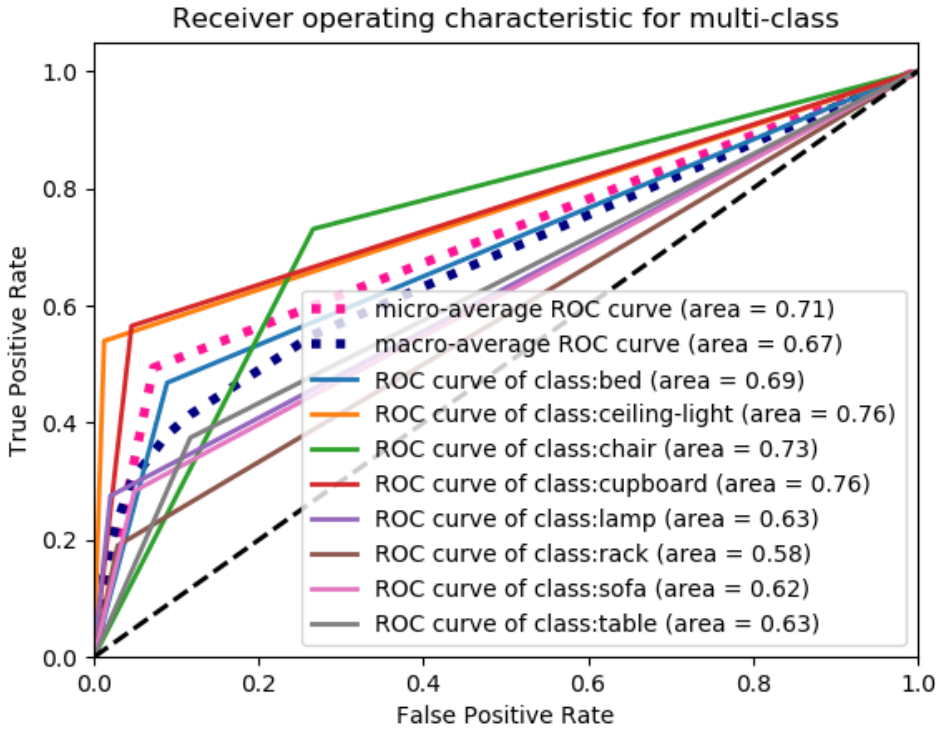
5.1.1.5 Custom CNN Mean Squared Error (MSE) for Train and Validation



[Figure 12: LEFT. Training MSE. RIGHT. Validation MSE for Custom Convolutional Neural Network (CNN)]

5.1.1.6 Area under Curve (AUC) of Receiver Operating Characteristic (ROC) Curve

Another measure of model performance is the receiver operating characteristic curve associated with the custom convolutional neural network. The best test accuracy occurs for the cupboard and ceiling light (both AUC of 0.76) as both have highest true positive rates. The lowest accuracy occurs for the rack (AUC is 0.58) as seen from the brown curve with a low true positive rate.

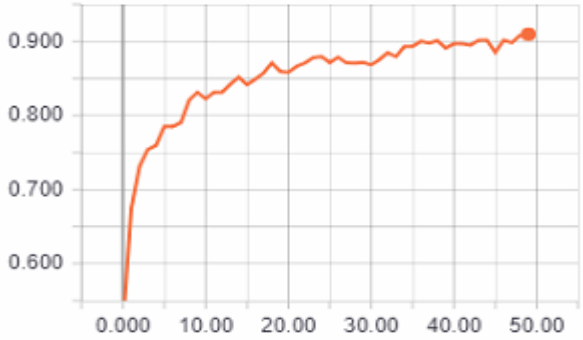


[Figure 13: Receiver Operating Characteristic (ROC) Curve associated with each furniture class for Custom Convolutional Neural Network Model]

5.1.2 Residual Network (ResNet-50) Results

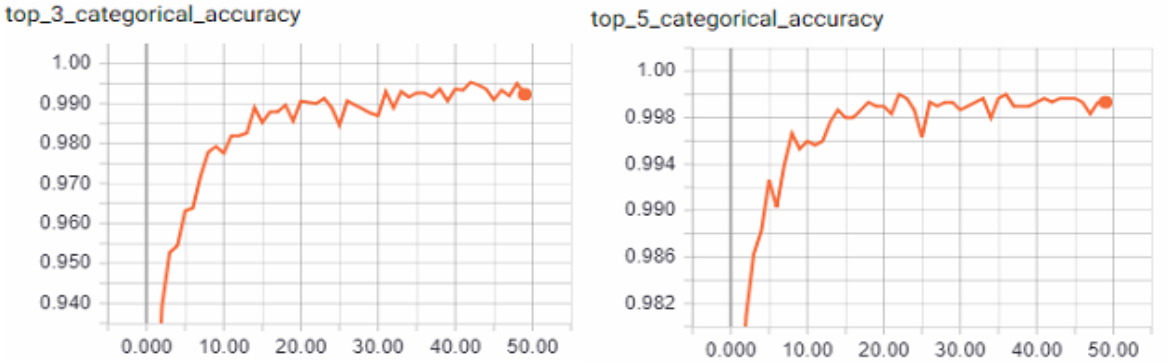
5.1.2.1 ResNet-50 Training Accuracy

The train and validation accuracy results for ResNet-50 were plotted on Tensor board and they are as follows. The Top-1 model accuracy over 50 epochs is shown below.



[Figure 14: Top-1 Model Accuracy for ResNet-50]

The Top-1 model accuracy exceeds 0.900 for the residual network. For top-3 and top-5 model accuracies, the values are above 0.990 and 0.998 respectively at 50 epochs (shown below).

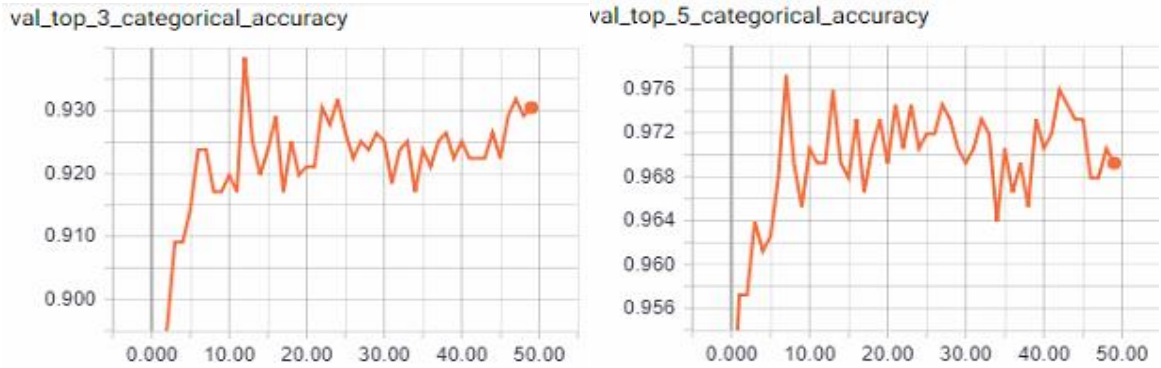


[Figure 15: LEFT. Top-3 Categories Accuracy. RIGHT. Top-5 Categories Train Accuracy for ResNet-50]

5.1.2.2 ResNet-50 Validation Accuracy

The validation accuracy graphs of the 50-layer residual network are as follows:

[Figure 16: Top-1 Validation Accuracy for ResNet-50]



[Figure 17: LEFT. Top-3 Categories Accuracy. RIGHT. Top-5 Categories Validation Accuracy for ResNet-50]

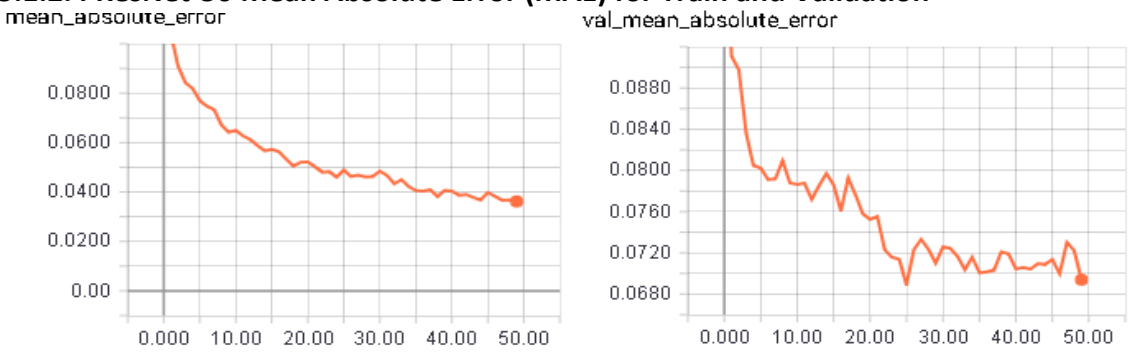
The validation accuracies of 0.760, 0.930 and 0.970 for top-1, top-3 and top-5 respectively are insufficient to measure to measure the model performance on the dataset. Hence, we look at the train and validation loss for the ResNet-50 model.

5.1.2.3 ResNet-50 Training Loss versus Validation Loss

From the results of Training and Validation loss for the ResNet-50 model, the validation loss graph is clearly higher than the train loss graph (note the different y-axis scales). This implies that the model performs better for the training set than the validation set. It means that overfitting of training data can occur such that the ResNet is less effective in classifying furniture in the validation set.

[Figure 18: Top-3 Training Loss. RIGHT. Top-5 Categories Validation Loss for ResNet-50]

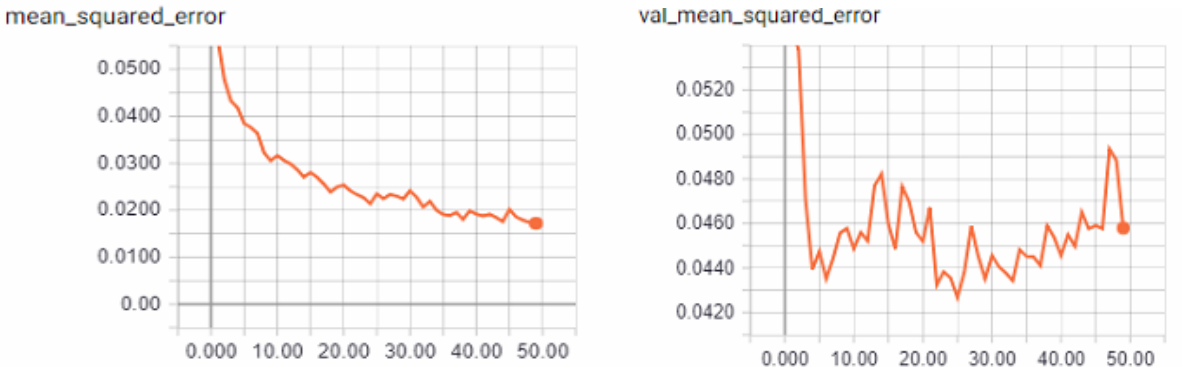
5.1.2.4 ResNet-50 Mean Absolute Error (MAE) for Train and Validation



[Figure 19: LEFT. Training MAE. RIGHT. Validation MAE for ResNet-50]

5.1.2.5 ResNet-50 Mean Squared Error (MSE) for Train and Validation

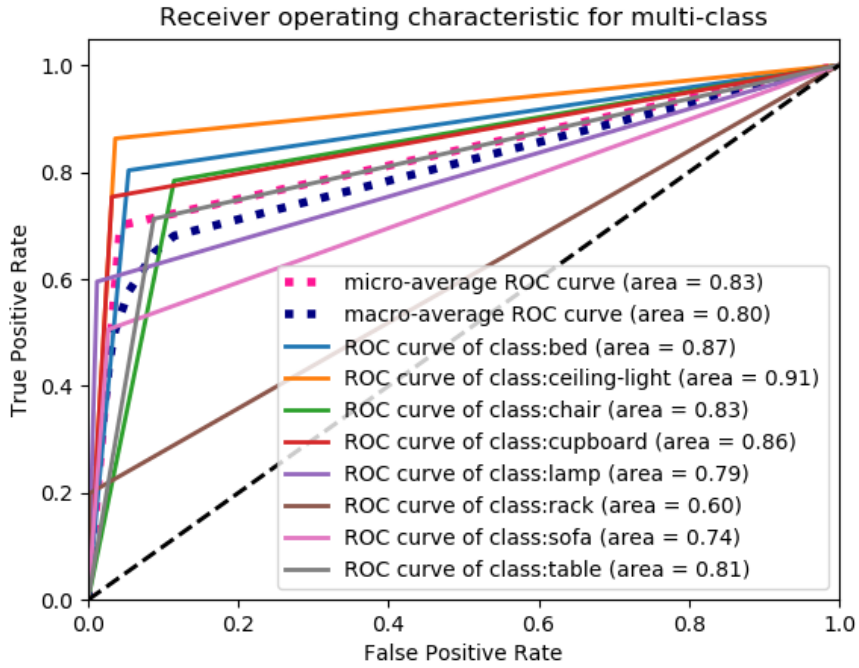
From the mean squared error graphs below, the validation MSE is higher than the train MSE. This is because the neural network has seen more data during training and the variance increases over the epochs. By applying the model learned from the training data to validation set, the validation MSE will be higher due to some overfit from training.



[Figure 20: LEFT. Training MSE. RIGHT. Validation MSE for ResNet-50]

5.1.2.6 Area under Curve (AUC) of Receiver Operating Characteristic (ROC) Curve

Finally, we examine the area under curve (AUC) of the ROC curve for the ResNet-50 model and realised that the class accuracies are higher as compared to the predictions by the Custom CNN model. The model best classifies ceiling-light with the highest true positive rate and an AUC of 0.91.



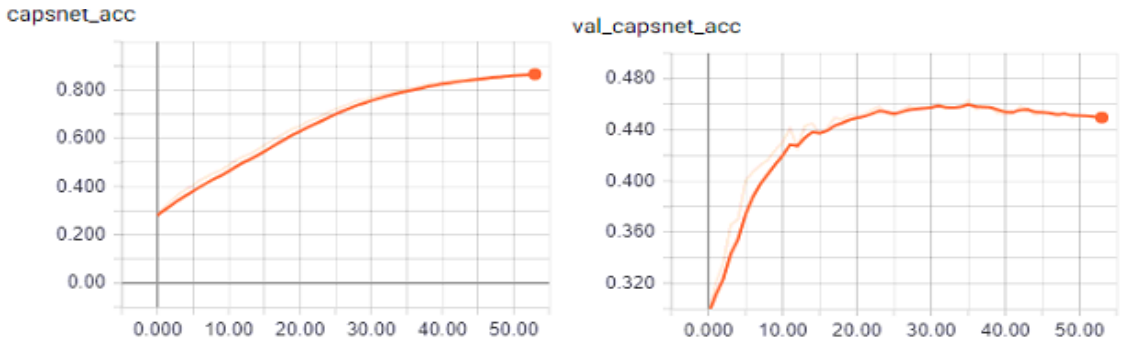
[Figure 21: Receiver Operating Characteristic (ROC) Curve associated with each furniture class for Residual Neural Network Model]

The precision, recall, f1-score and support associated with each furniture class for the Custom CNN as well as the residual network model during testing are presented in appendix.

5.1.3 Capsule Network Results

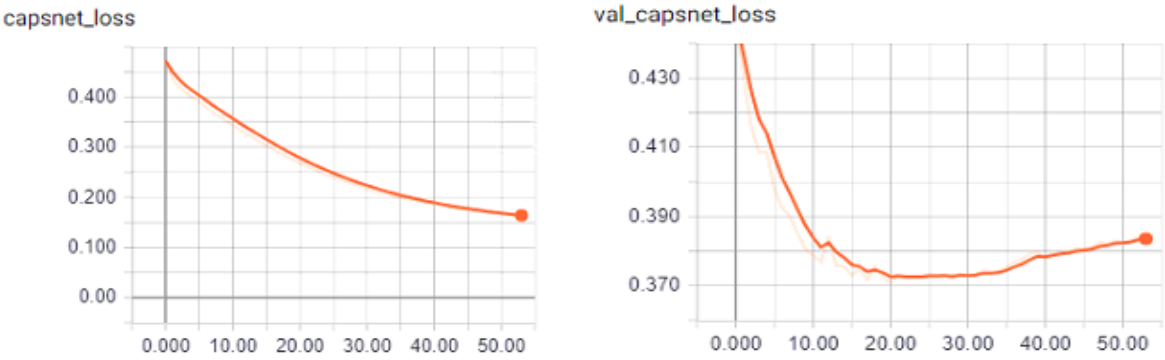
5.1.3.1 CapsNet Train and Validation Accuracy

The Capsule Network train accuracy increases from 0.300 to above 0.800 over 50 epochs. The validation accuracy plateaus at around 0.440 by 50 epochs.

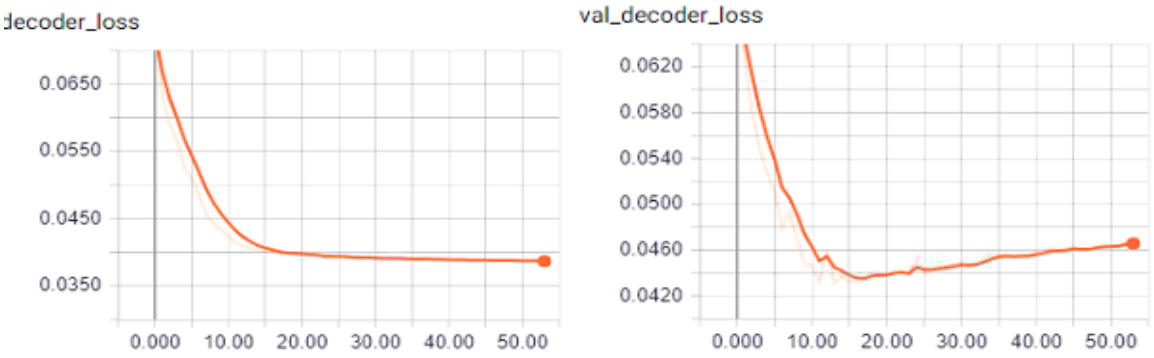


[Figure 22: LEFT. Train Accuracy. RIGHT. Validation Accuracy for CapsNet]

5.1.3.2 CapsNet Loss Graphs



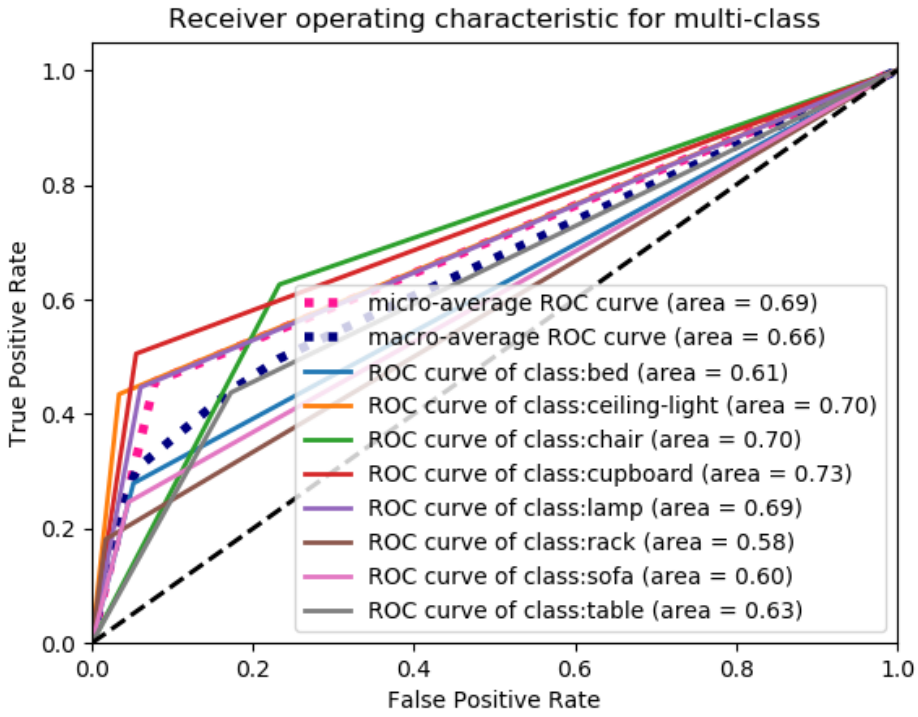
[Figure 23: LEFT. Train Loss. RIGHT. Validation Loss for CapsNet]



[Figure 24: LEFT. Train Decoder Loss. RIGHT. Validation Decoder Loss for CapsNet]

5.1.3.4 Area under Curve (AUC) of Receiver Operating Characteristic (ROC) Curve

The area under curve (AUC) of Receiver Operating Characteristic (ROC) for Capsule Network for each of the furniture class is shown below. The highest AUC is 0.73 which is indicative of the fact that Capsule Network predicts cupboards most accurately. On the other hand, rack has the lowest AUC of 0.58 indicating that the classifier lacks in accuracy in this area.



[Figure 25: Receiver Operating Characteristic (ROC) Curve associated with each furniture class for Capsule Network]

5.2 Model Comparison

5.2.1 Custom CNN Top-1 Classification Report for Test Set

Top-1 Accuracy: 0.494

S.No.	Category	Precision	Recall	F1-score	Support
1	Bed	0.35	0.47	0.40	914
2	Ceiling light	0.75	0.54	0.63	619
3	Chair	0.53	0.73	0.62	2840
4	Cupboard	0.62	0.57	0.59	1136
5	Lamp	0.61	0.28	0.38	1003
6	Rack	0.24	0.19	0.21	447
7	Sofa	0.39	0.28	0.33	963
8	Table	0.41	0.37	0.39	1710

[Table 3: Custom CNN Classification Report for Top-1]

5.2.2 ResNet-50 Top-1 Classification Report for Test Set

Top-1 Accuracy: 0.744

S.No.	Category	Precision	Recall	F1-score	Support
1	Bed	0.64	0.85	0.73	153
2	Ceiling light	0.80	0.79	0.79	104
3	Chair	0.75	0.80	0.77	474
4	Cupboard	0.85	0.80	0.83	189
5	Lamp	0.83	0.65	0.73	168
6	Rack	0.89	0.33	0.49	75
7	Sofa	0.62	0.70	0.65	159
8	Table	0.75	0.73	0.74	285

[Table 4: ResNet50 Classification Report for Top-1]

5.2.3 Capsule Net Top-1 Classification Report for Test Set

Top-1 Accuracy: 0.456

S.No.	Category	Precision	Recall	F1-score	Support
1	Bed	0.36	0.28	0.32	914
2	Ceiling light	0.47	0.43	0.45	619
3	Chair	0.53	0.63	0.57	2840
4	Cupboard	0.55	0.51	0.53	1136
5	Lamp	0.46	0.45	0.45	1003
6	Rack	0.34	0.18	0.24	447
7	Sofa	0.38	0.25	0.30	963
8	Table	0.35	0.44	0.39	1710

[Table 5: Capsule Net Classification Report Top-1]

5.3 Parameter Tuning on Simple Custom CNN

We used grid search with cross-validation to traverse the hyperparameter space and to also try different optimizers. The following table shows the parameters tested and the results obtained. Note that the following results were obtained for 1 convolution layer and 1 max pooling layer simple custom convolutional neural network (CNN) architecture before adding more layers to deepen the neural network. We did not utilise grid search cross-validation on the new CNN, ResNet and CapsNet due to the process being computationally expensive to run.

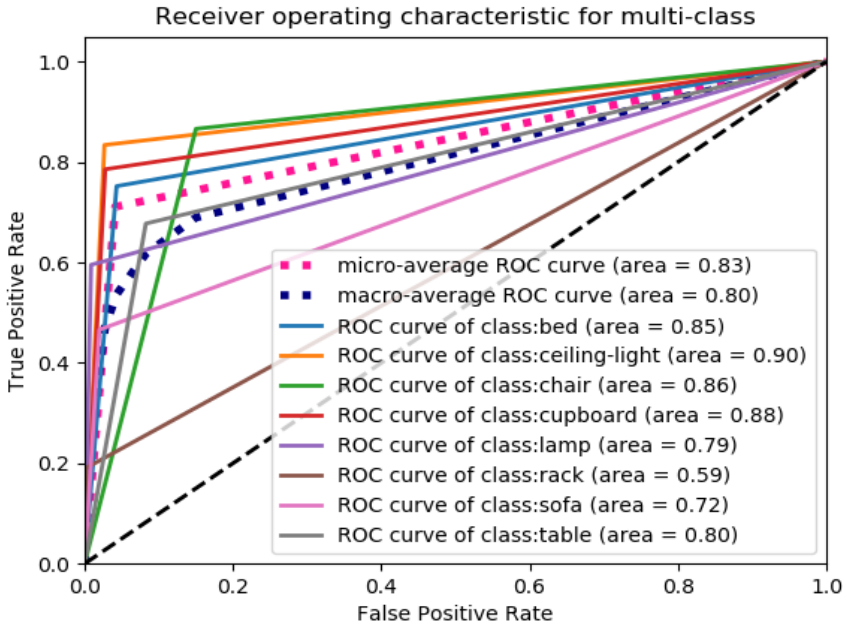
Activation	Batch Size	Dropout Rate	Number of Epochs	Optimizer	Mean Test Score	Stdev Test Score
ReLU	32	0	50	SGD	0.0553	0.0156
ReLU	32	0	50	Adam	0.0817	0.0254
ReLU	64	0	50	SGD	0.0448	0.00926
ReLU	64	0	50	Adam	0.0520	0.0113
ReLU	1024	0	50	SGD	0.0284	0.0238
ReLU	1024	0	50	Adam	0.00873	0.00285
ReLU	128	0.4	50	SGD	0.0577	0.00539
ReLU	128	0.4	50	RMSprop	0.0501	0.00882
ReLU	128	0.5	50	SGD	0.0473	0.00944
ReLU	128	0.5	50	RMSprop	0.0471	0.0219
Sigmoid	128	0.4	50	SGD	0.00521	0.00521
Sigmoid	128	0.4	50	RMSprop	0.132	0.0124
Sigmoid	128	0.5	50	SGD	0.00321	0.00321
Sigmoid	128	0.5	50	RMSprop	0	0

[Table 6: Parameters and Optimizers tuned for Simple CNN]

5.4 Ensemble Methods

5.4.1. Majority Vote

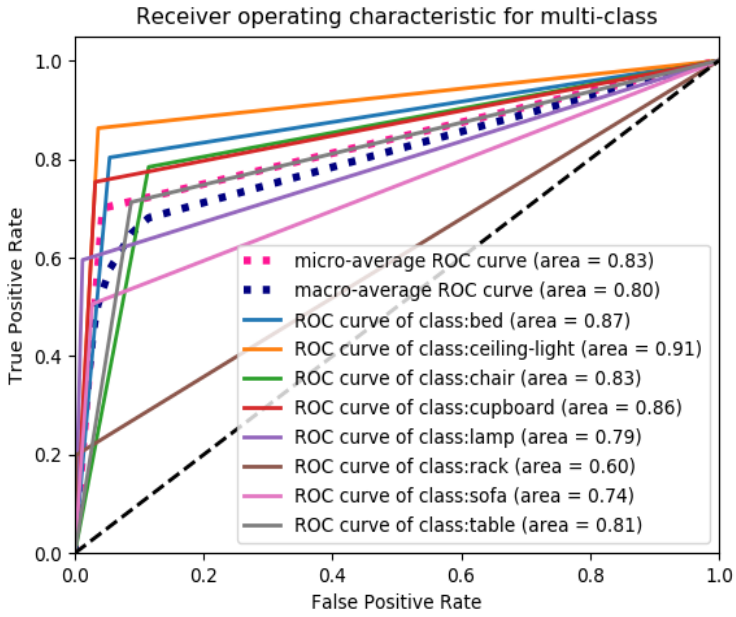
The three models participate in class voting to decide the winner for the image based on majority. In case of a tie or no majority, the class label associated with ResNet is chosen since it is the better performing model among the three.



[Figure 26: Receiver Operating Characteristic (ROC) Curve associated with each furniture class for Ensemble Majority Voting]

5.4.2. Weighted Sum

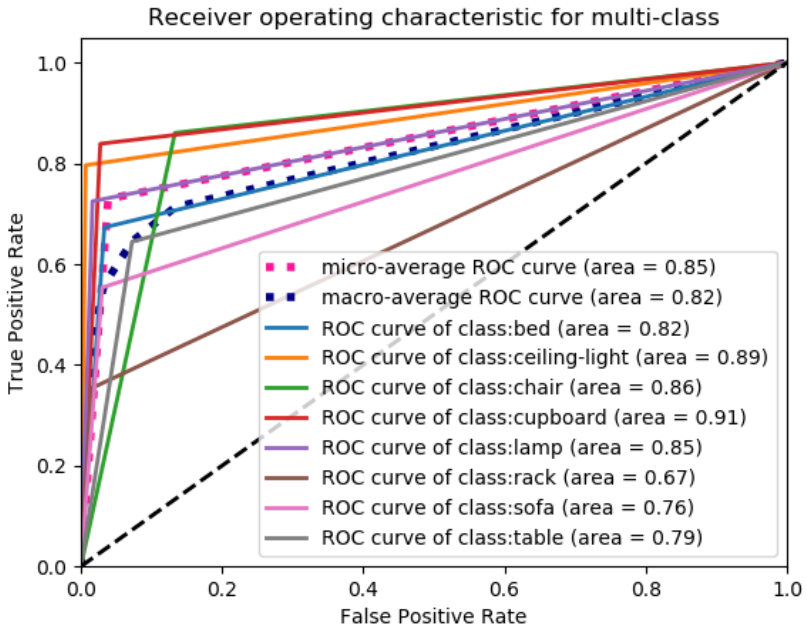
Instead of directly utilizing the predicted classes, we considered the individual class probability scores and added them together (sum per class across models) We also added a weight factor: 0.65 for custom CNN, 1.0 for ResNet and 0.7 for CapsNet to stress more on the better performing model - ResNet.



[Figure 27: Receiver Operating Characteristic (ROC) Curve associated with each furniture class for Ensemble Weighted Sum]

5.4.3. Stacking

Stacking involves building a meta classifier which takes as inputs the output probability scores of the individual classifiers and trains on them, to give an improved output score of the final classification. We employ Logistic Regression as the choice of meta-classifier here. Compared to the other approaches, Stacking seems to identify cupboard slightly better than ceiling lights.

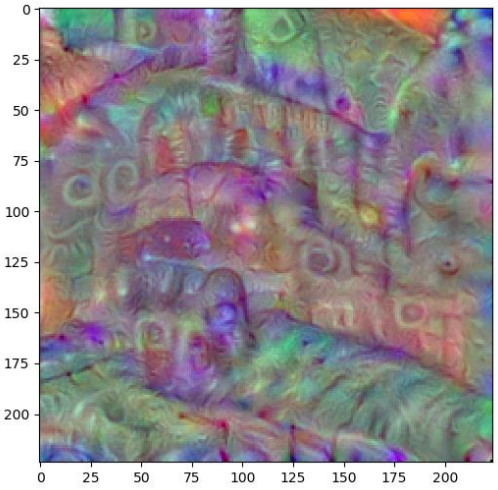


[Figure 28: Receiver Operating Characteristic (ROC) Curve associated with each furniture class for Ensemble Stacking]

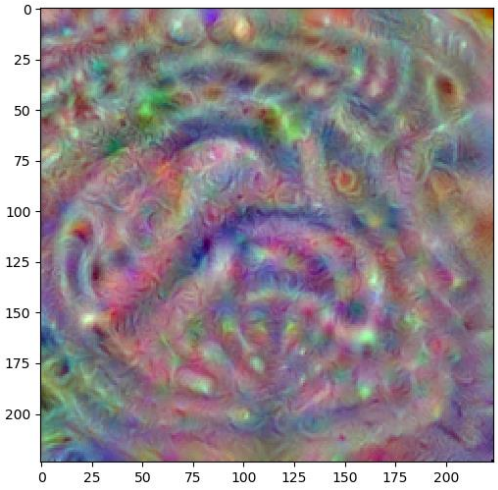
5.5 CLASS ACTIVATION MAPS

5.4.2 ResNet-50 Class Activations

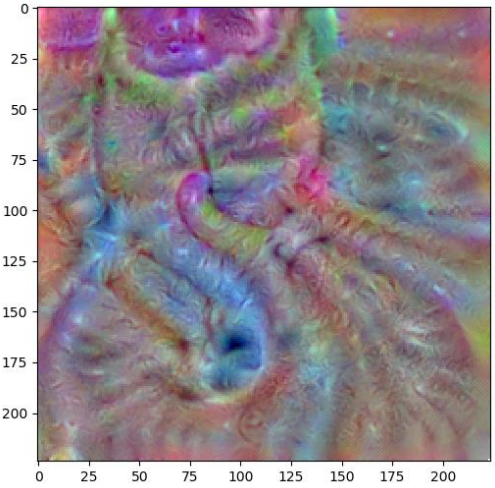
Using Keras visualisation toolkit [6], the following series of pictures show the class activation visualisation associated with each furniture class.



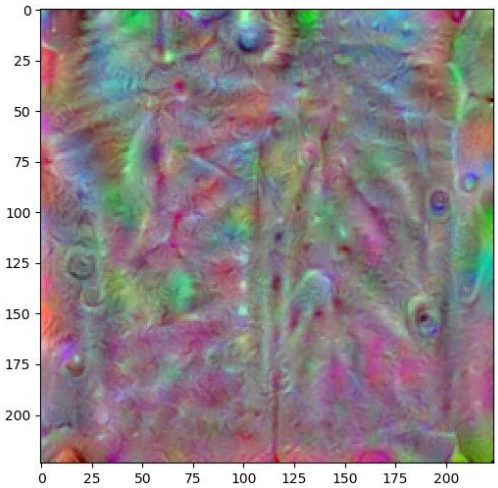
Bed (Class 0)



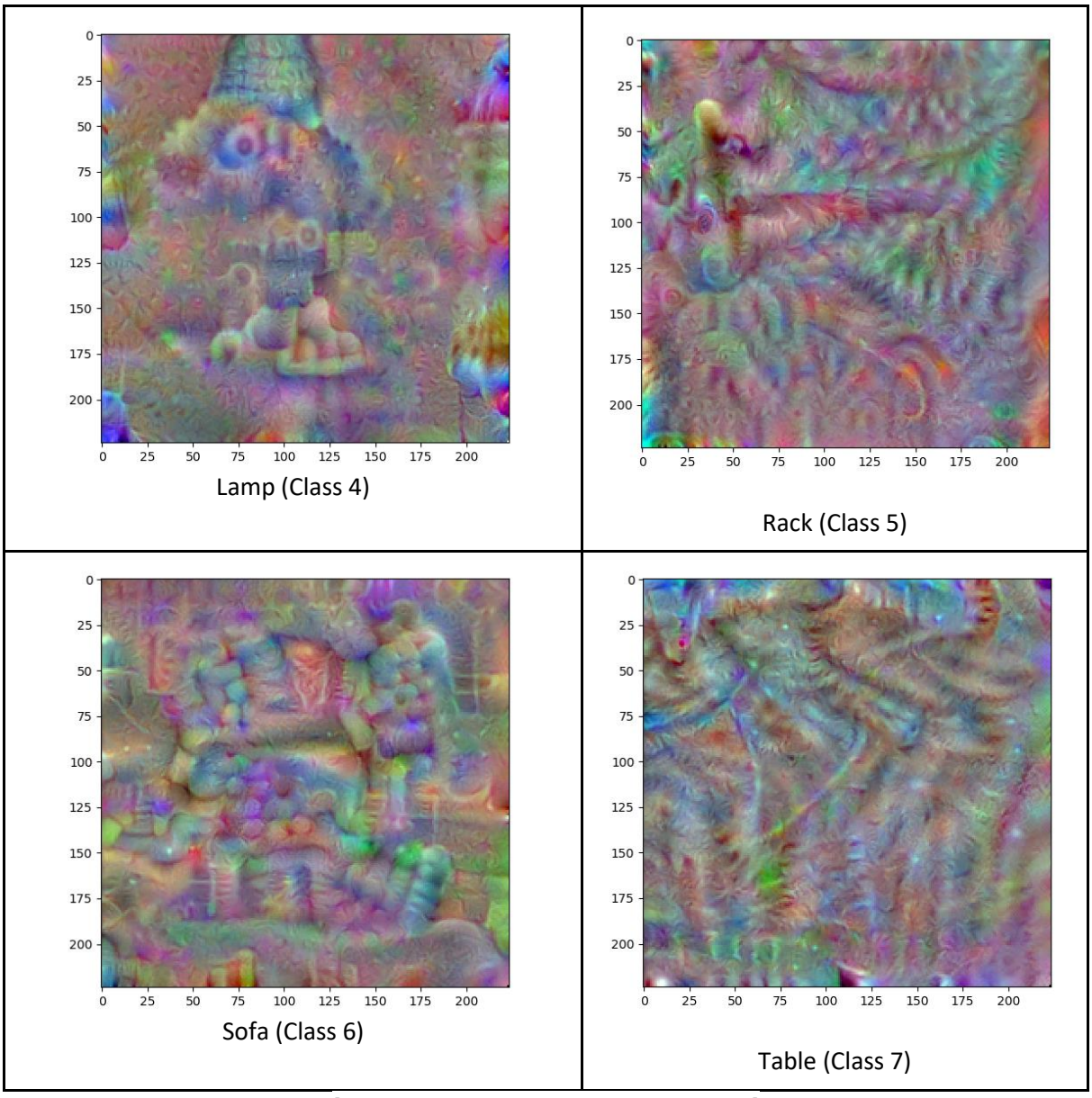
Ceiling-Light (Class 1)



Chair (Class 2)



Cupboard (Class 3)



[Table 7: Class Activation Visualisations]

15. FUTURE ENHANCEMENTS

We will vary the probability values such as dropout rate from a range of values from 20% to 80% to prevent overfitting during training on the dataset for Capsule Network and Residual network. We can also try to control the image augmentation properties even more to get more variety.

Due to the limited computing resources available, we were unable to run cross-validation for the deeper networks such as Capsule Network and Residual Network models. If better computing hardware emerges for us to use, we can tune the parameters using cross-validation.

16. ACKNOWLEDGEMENT

We thank Kaggle and CVPR [7] for publishing the data in their website.

17. CONCLUSION

In this report, we have used custom convolutional neural network, residual neural network and capsule network to classify furniture images into 8 classes from the iMaterialist Challenge at FGVC5 Image classification of furniture dataset. We conclude that ResNet-50 has the best model performance on the furniture image classification dataset. This is because the weights loaded into the ResNet are pre-trained from ImageNet dataset. ResNet with Transfer learning provides the highest class accuracy amongst all the neural network models tried individually. Among the ensembles, stacking based on logistic regression proves more effective than any of the previous runs.

18. REFERENCES

1. He, K., Zhang, X., Ren, S., & Sun, J. (2015). Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *2015 IEEE International Conference on Computer Vision (ICCV)*. doi:10.1109/iccv.2015.123.

2. Kaggle challenge https://www.kaggle.com/c/imaterialist-challenge-furniture-2018.

3. Liu, S., & Deng, W. (2015). Very deep convolutional neural network based image classification using small training sample size. *2015 3rd IAPR Asian Conference on Pattern Recognition (ACPR)*. doi:10.1109/acpr.2015.7486599.

4. He, K., Zhang, X., Ren, S., & Sun, J. (2016). Deep Residual Learning for Image Recognition. *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. doi:10.1109/cvpr.2016.90.

5. Hinton, G. E., Sabour, S., Frosst, N. (2017). Dynamic Routing between Capsules. arXiv:1710.09829.

6. Kotikalapudi, Raghavendra and Contributors. (2017). Keras-Vis. Github: https://github.com/raghakot/keras-vis.

7. Conference on Computer Vision and Pattern Recognition (CVPR).

10. Appendix

RESNET TOP-1,TOP-2, TOP-3, TOP-4, TOP-5

Top-1 Accuracy: 0.7442439327940261

S.No.	Category	Precision	Recall	F1-score	Support
1	Bed	0.64	0.85	0.73	153
2	Ceiling light	0.80	0.79	0.79	104
3	Chair	0.75	0.80	0.77	474
4	Cupboard	0.85	0.80	0.83	189
5	Lamp	0.83	0.65	0.73	168
6	Rack	0.89	0.33	0.49	75
7	Sofa	0.62	0.70	0.65	159
8	Table	0.75	0.73	0.74	285

Top-2 Accuracy: 0.8008711885500933

S.No.	Category	Precision	Recall	F1-score	Support
1	Bed	0.68	0.85	0.76	153
2	Ceiling light	0.81	0.79	0.80	104
3	Chair	0.81	0.80	0.80	474
4	Cupboard	0.90	0.80	0.85	189
5	Lamp	0.85	0.65	0.74	168
6	Rack	0.89	0.33	0.49	75
7	Sofa	0.69	0.79	0.74	159
8	Table	0.84	1.00	0.91	285

Top-3 Accuracy: 0.8294959551960174

S.No.	Category	Precision	Recall	F1-score	Support
1	Bed	0.79	0.85	0.82	153
2	Ceiling light	0.81	0.79	0.80	104
3	Chair	0.82	0.80	0.81	474
4	Cupboard	0.90	0.80	0.85	189
5	Lamp	0.85	0.65	0.74	168
6	Rack	0.95	0.51	0.66	75
7	Sofa	0.74	1.00	0.85	159
8	Table	0.87	1.00	0.93	285

Top-4 Accuracy: 0.8556316116988176

S.No.	Category	Precision	Recall	F1-score	Support
1	Bed	0.80	0.85	0.82	153
2	Ceiling light	0.84	0.79	0.81	104
3	Chair	0.85	0.80	0.82	474
4	Cupboard	0.96	0.80	0.88	189
5	Lamp	0.88	0.68	0.77	168
6	Rack	0.97	1.00	0.99	75
7	Sofa	0.75	1.00	0.86	159
8	Table	0.88	1.00	0.93	285

Top-5 Accuracy: 0.8985687616677038

S.No.	Category	Precision	Recall	F1-score	Support
1	Bed	0.82	0.85	0.84	153
2	Ceiling light	0.91	0.79	0.85	104
3	Chair	0.93	0.80	0.86	474
4	Cupboard	0.98	0.89	0.93	189
5	Lamp	0.92	1.00	0.96	168
6	Rack	0.97	1.00	0.99	75
7	Sofa	0.76	1.00	0.86	159
8	Table	0.91	1.00	0.95	285

CUSTOM CNN TOP-1,TOP-2, TOP-3, TOP-4, TOP-5

Top-1 Accuracy: 0.49356312292358806

S.No.	Category	Precision	Recall	F1-score	Support
1	Bed	0.35	0.47	0.40	914
2	Ceiling light	0.75	0.54	0.63	619
3	Chair	0.53	0.73	0.62	2840
4	Cupboard	0.62	0.57	0.59	1136
5	Lamp	0.61	0.28	0.38	1003
6	Rack	0.24	0.19	0.21	447
7	Sofa	0.39	0.28	0.33	963
8	Table	0.41	0.37	0.39	1710

Top-2 Accuracy: 0.6192898671096345

S.No.	Category	Precision	Recall	F1-score	Support
1	Bed	0.42	0.47	0.45	914
2	Ceiling light	0.80	0.54	0.64	619
3	Chair	0.62	0.73	0.67	2840
4	Cupboard	0.71	0.57	0.63	1136
5	Lamp	0.65	0.28	0.39	1003
6	Rack	0.29	0.19	0.23	447
7	Sofa	0.55	0.43	0.48	963
8	Table	0.69	1.00	0.81	1710

Top-3 Accuracy: 0.6887458471760798

S.No.	Category	Precision	Recall	F1-score	Support
1	Bed	0.55	0.47	0.51	914
2	Ceiling light	0.81	0.54	0.65	619
3	Chair	0.67	0.73	0.70	2840
4	Cupboard	0.73	0.57	0.64	1136
5	Lamp	0.67	0.28	0.39	1003
6	Rack	0.52	0.46	0.49	447
7	Sofa	0.75	1.00	0.86	963
8	Table	0.71	1.00	0.83	1710

Top-4 Accuracy: 0.7315199335548173

S.No.	Category	Precision	Recall	F1-score	Support
1	Bed	0.58	0.47	0.52	914
2	Ceiling light	0.83	0.54	0.65	619
3	Chair	0.70	0.73	0.72	2840
4	Cupboard	0.77	0.57	0.65	1136
5	Lamp	0.78	0.45	0.57	1003
6	Rack	0.75	1.00	0.86	447
7	Sofa	0.78	1.00	0.87	963
8	Table	0.74	1.00	0.85	1710

Top-5 Accuracy: 0.8127076411960132

S.No.	Category	Precision	Recall	F1-score	Support
1	Bed	0.61	0.47	0.53	914
2	Ceiling light	0.93	0.54	0.68	619
3	Chair	0.82	0.73	0.77	2840
4	Cupboard	0.87	0.76	0.81	1136
5	Lamp	0.91	1.00	0.95	1003
6	Rack	0.83	1.00	0.91	447
7	Sofa	0.80	1.00	0.89	963
8	Table	0.78	1.00	0.88	1710