Real Estate Machine Learning for Dubizzle

Problem 1: Multiclass Image Classification catering to Area Types

Dataset

The REI (Real Estate Image) dataset has been used to train various models and perform a comparative analysis of which model scores the highest accuracy.

Data Information

6 folders [classes]
Backyard [745 jpeg images]
Bathroom [793 jpeg images]
Bedroom [1593 jpeg images]
Frontyard [884 jpeg images]
Kitchen [992 jpeg images]
LivingRoom [852 jpeg images]

Data Link

https://researchers.cdu.edu.au/en/publications/real-estate-image-classification

Shortlisted Models

As the problem is an image classification problem with a large dataset (images from Zameen etc), deep learning is most suitable for it. Deep learning is able to cater to large datasets in a short amount of time and extract features automatically using various model layers instead of manual feature extraction. This allows the creation of more complex neural networks and therefore better results.

Potential Models

After research in the multiclass image classification domain models such as ResNet, AlexNet, VGG, Inception, EfficientNet, MobileNet were found to be suitable.

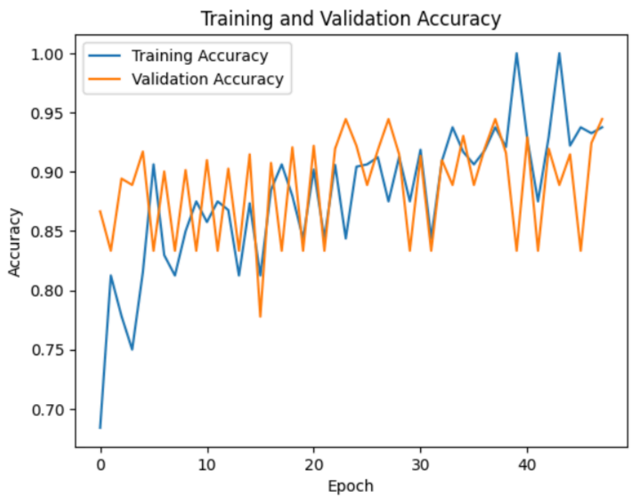
Tested Models

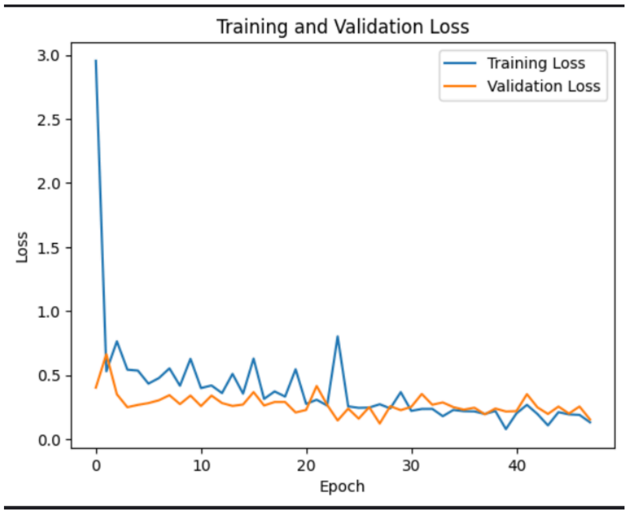
The models from the above list were further shortlisted for training the dataset (based on various results obtained by other works) are the following:

- ResNet152 (Larger dataset) or ResNet50V2 (Smaller dataset)
- EfficientNet
- InceptionV3
- MobileNet

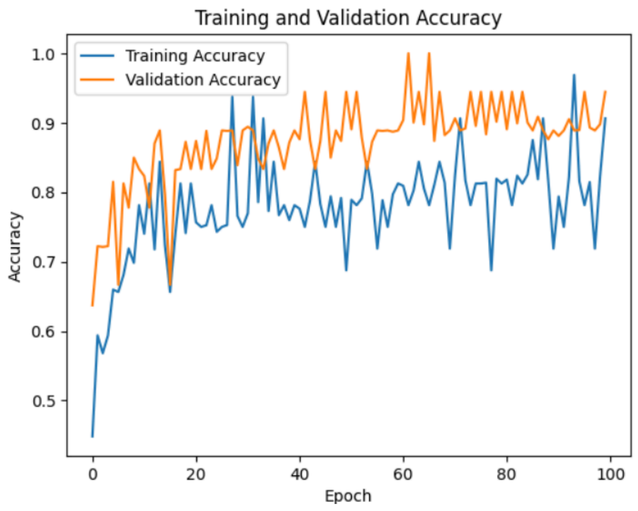
Results

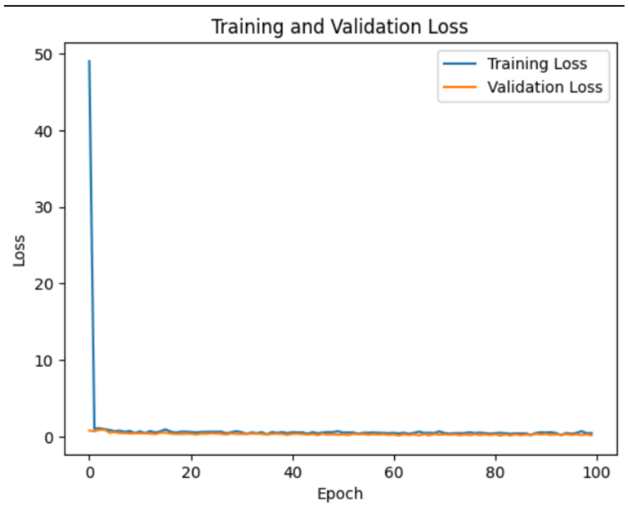
ResNetV50 (Use 152 for a larger dataset)

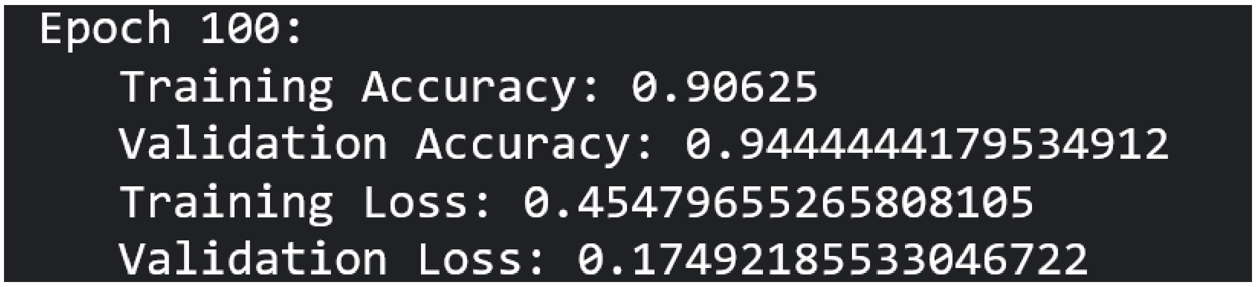




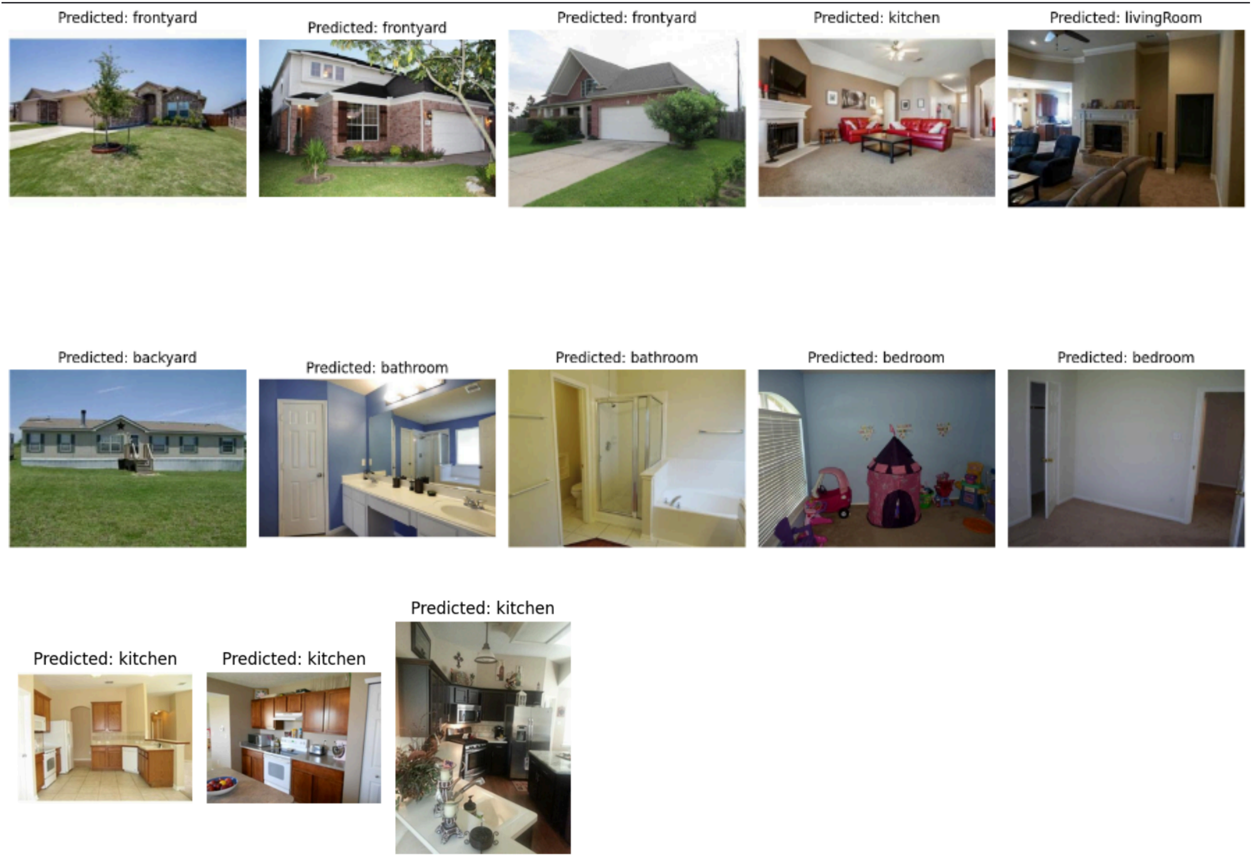
InceptionV3







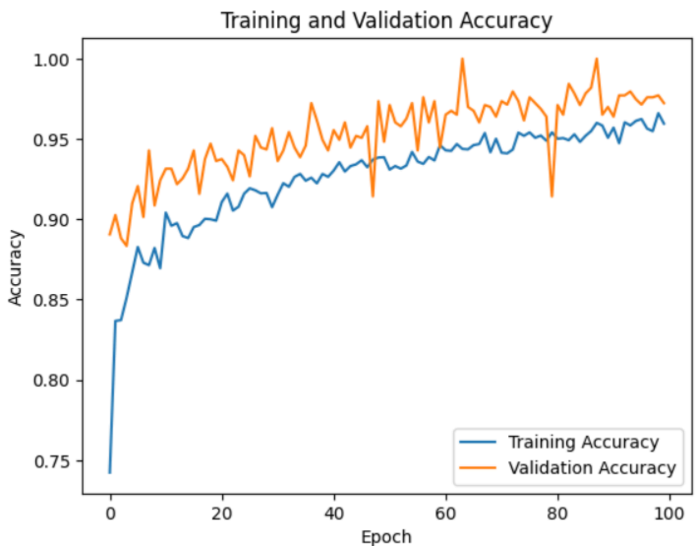
```
Epoch 100:
    Training Accuracy: 0.90625
    Validation Accuracy: 0.9444444179534912
    Training Loss: 0.45479655265808105
    Validation Loss: 0.17492185533046722
```

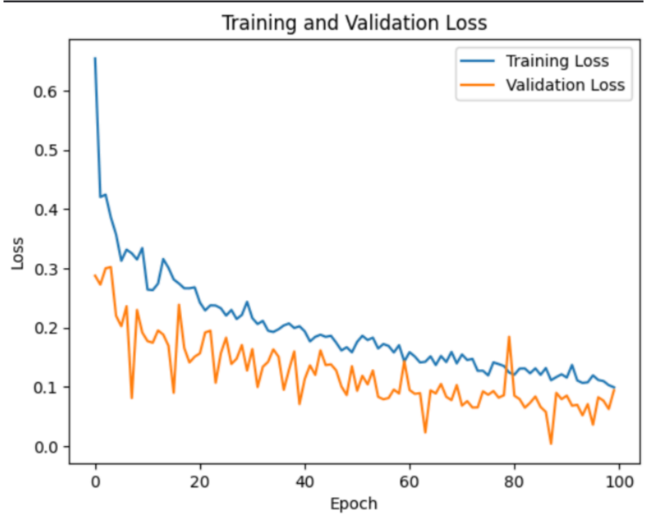


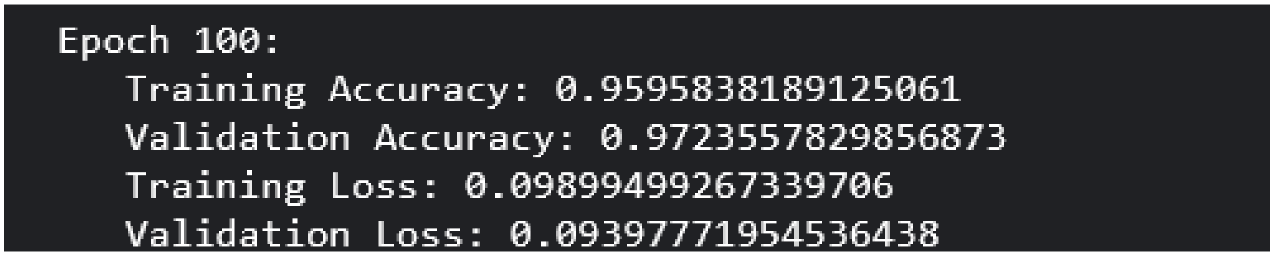
<u>Actual: FrontYard, FrontYard, FrontYard, LivingRoom, LivingRoom, BackYard, Bathroom, Bathroom, Bedroom, Bedroom, Kitchen, Kitchen, Kitchen</u>

12/13 correctly identified

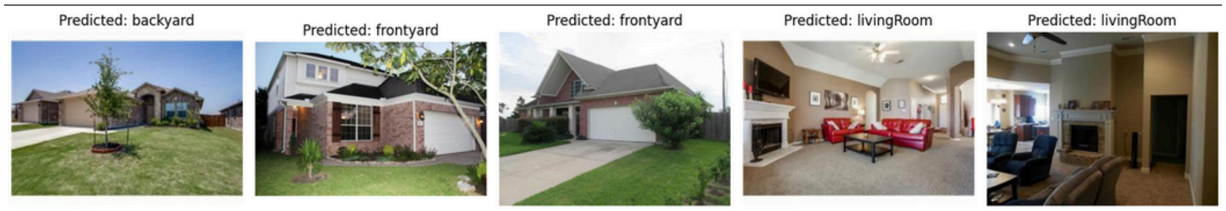
MobileNetV2

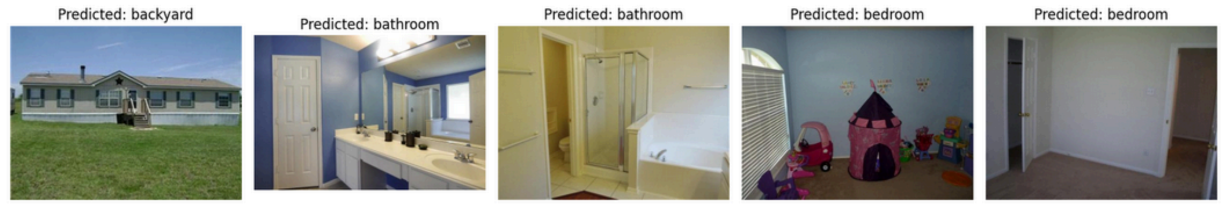


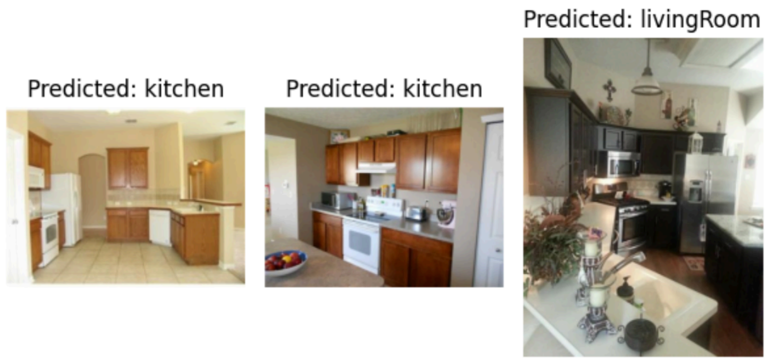




```
Epoch 100:
   Training Accuracy: 0.9595838189125061
   Validation Accuracy: 0.9723557829856873
   Training Loss: 0.09899499267339706
   Validation Loss: 0.09397771954536438
```



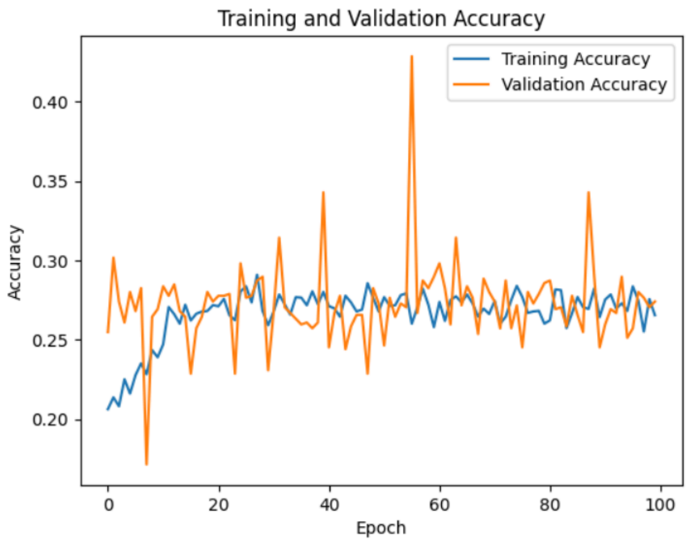




Actual: FrontYard, FrontYard, FrontYard, LivingRoom, LivingRoom, BackYard, Bathroom, Bathroom, Bedroom, Bedroom, Kitchen, Kitchen, Kitchen

11/13 correctly identified

EfficientNet (Did not perform well)



Analysis of the current dataset: **MobileNet touched an accuracy of 97% and InceptionNet touched an accuracy of 95%**, therefore both are top candidates for training the new dataset. **MobileNet gave 11/13 correctly identified samples** while **InceptionV3 gave 12/13 (with lesser accuracy).** This validation accuracy was achieved on a smaller (around 4000 images) REI dataset without much data augmentation. There was some overfitting observed along with quick accuracy jumps which can be made stable by further increasing data + augmentations along with experimenting with dropout / batch normalization and dense layers. K fold validation can also be performed for Dubizzle's dataset to ensure there are little to no wrong predictions.

Final Approach

Data augmentation

Add data augmentation techniques to add variability to the images. It can be done in various ways:
1. If the dataset is large, only applying augmentations on the existing images is enough.
2. If the dataset is small, perform various augmentations on a single image and save each altered image along with the original one to increase the effective size of the dataset.

Augmentation techniques like rotation, width/height shift, shear, zoom, horizontal flip, brightness adjustment.

Model construction

Use a pre-trained CNN as the base model and apply transfer learning (final model chosen: MobileNet (if it doesn’t perform optimally on test data InceptionV3 is the next option).
Remove the fully connected layers at the top of the base model.
Add custom fully connected layers for classification.

Freeze the layers of base to prevent their weights from being updated in training.
Compile the model with Adam optimizer, categorical (as there are many classes) loss function, and evaluation metric.

Model training
Train the model on the augmented training data.
Utilize callbacks such as early stopping (if no increase after certain epochs) and learning rate scheduling (decrease learning rate along the training process) to improve training efficiency and also prevent overfitting.
Dropout / Batch Normalization can be experimented with.

Model evaluation
Evaluate the trained model on the validation set.
Predict the separated testing images.
Calculate performance metrics such as accuracy, confusion matrix and F1-score.

Visualization
Plot training/ validation accuracy over epochs.
Plot training / validation loss over epochs.
Visualize model predictions on sample images as well as their ground truth labels.
Visualize the confusion matrix to know correctly vs incorrectly deduced predictions.

Hyperparameter tuning (if needed)
Experiment with different hyperparameters like learning rate, dropout rate, batch size, etc., to optimize model performance. Add different learning rate schedulers / layers.

Problem 2: Detection of Real Estate vs Non Real Estate Images

There can be different ways to tackle this problem i.e. using an entirely different model with a varying dataset to differentiate between real estate and non real estate images.
However, using two different models can raise performance issues and it would be more feasible to use a single model for **both problem one and problem two.** This could potentially be done by passing the image through the model for problem one and then comparing the thresholds obtained for each of the classes. If the image doesn't belong to any of the classes (low probability of each class) then we can deduce that the image is not a real estate example. This will be useful only when the real estate images are bound to be within the specified classes.

Problem 3: Identifying the presence of inappropriate content

For such a problem, two potential solutions can be carried out.

1. Using existing APIs of well established models. For example the Cloud Vision API (Google).
2. Train a separate model for this purpose.

Cloud Vision API

The API flags inappropriate images automatically and minimizes false positives.
SafeSearch detection categorizes inappropriate content into spoof, medical, adult, and violence.
Likelihood values ranging from "VERY_UNLIKELY" to "VERY_LIKELY."
Violence classifier also identifies images depicting killing, shooting, or blood and gore, using neural network analysis.

Deep learning based Research Paper Finding

Transfer Learning and Feature Fusion used.
TL utilized for knowledge transfer.
FFP for removing hyper-parameter tuning, improving performance, and reducing computations.
Fusion of low-level and mid-level features for better classification.
Generation of well-labeled obscene image dataset (GGOI) via Pix-2-Pix GAN.
Model architecture modifications for stability.
Selection of outperforming models for integration with FFP.
TL-based obscene image detection method by retraining the last layer.
TL model with MobileNet V2 + DenseNet169 fusion performs as state-of-the-art.
Average classification accuracy, sensitivity, and F1 score of 98.50%, 98.46%, and 98.49% respectively.

Problem 4: Text and Competitor Logo detection

Text

Text can be detected within an image using OCR technology (optical character recognition). For example: Python has a PyTesseract library that can detect text within an image. Google AI also allows text recognition within an image. Such technology can be incorporated to validate an image before passing it through the classifier and reject it beforehand.

Logos

Logo detection lies within the object detection field and therefore requires an object detection model. YOLO can be fed with various logos to look out for and pinpoint that particular logo within an image along with the coordinates (if any). A simple classifier will not work well for such a problem.
We can either train YOLO (Latest: YOLOV7) on an existing logo dataset or train a model with the images on Dubizzle’s platform by manually adding a file that contains annotations of coordinates for images having logos (like a bounding box).