



Presented By:

Brandy, Nasrin, and Toyin

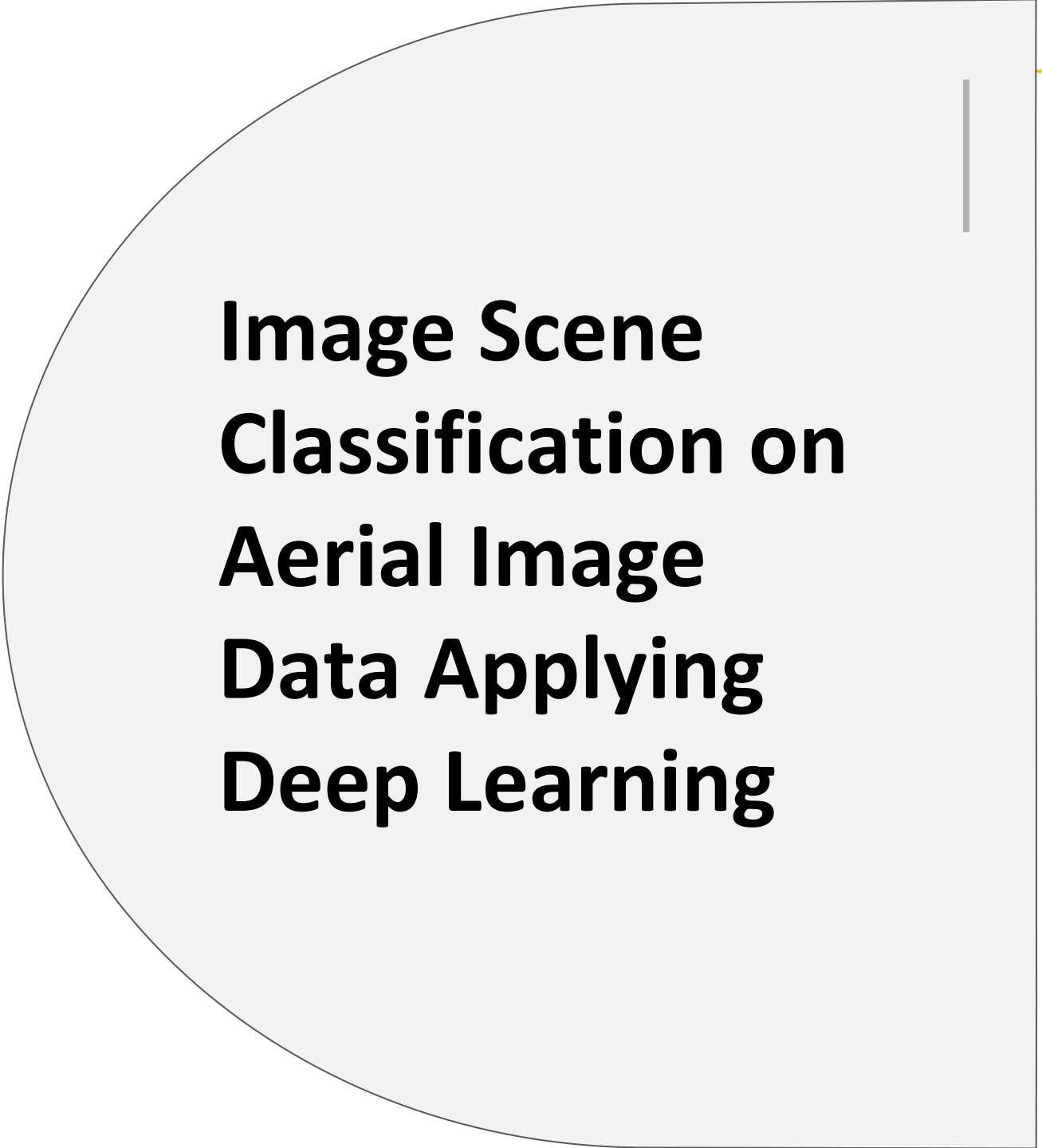
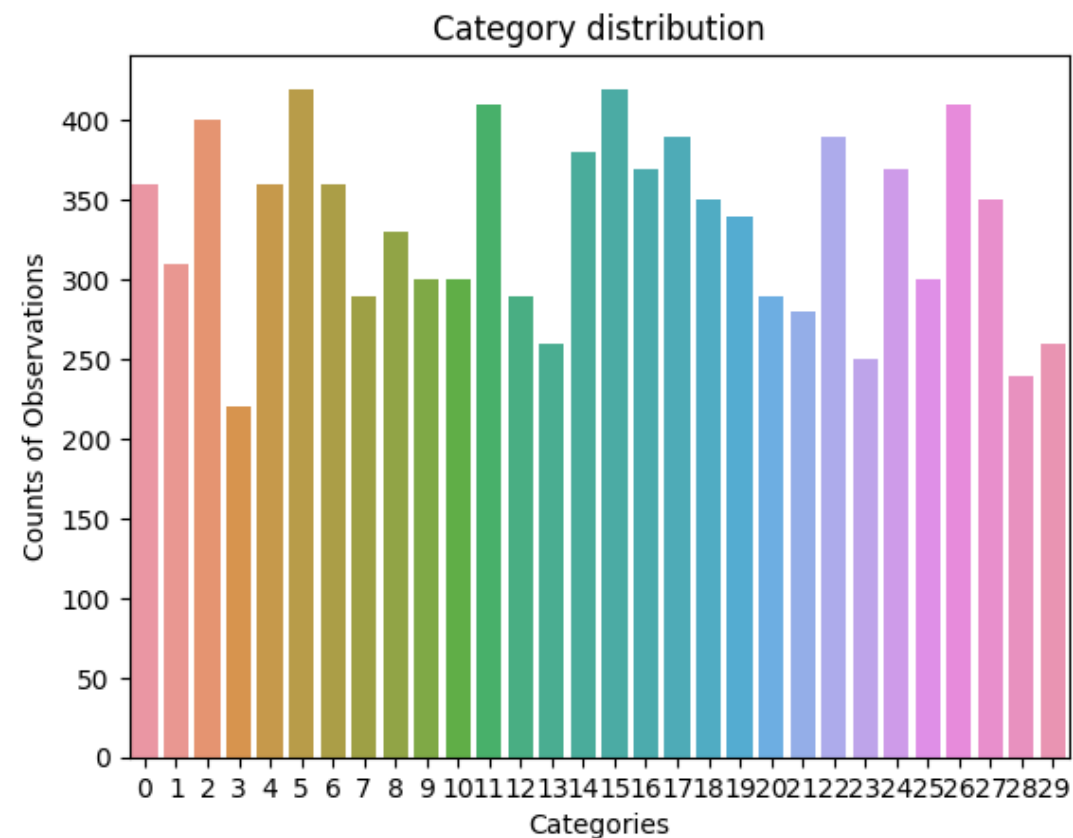


Image Scene Classification on Aerial Image Data Applying Deep Learning

Introduction

Image Scene Classification on Aerial Imagery Applying Deep Learning



Thirty Categories of AID's

- CNNs can be utilized to classify aerial imagery datasets (AID) to recognize urban land-use objects and patterns, but less research exists in scene classification.
- A scene is the composition of the objects and patterns.
- The aim of this study is to develop an automated framework for scene classification of an urban land-use AID through CNN.
- Various models are available, e.g., VGG16 and MobileNET-



Introduction: Importance of Research Topic

Identify
residential
and traffic
patterns

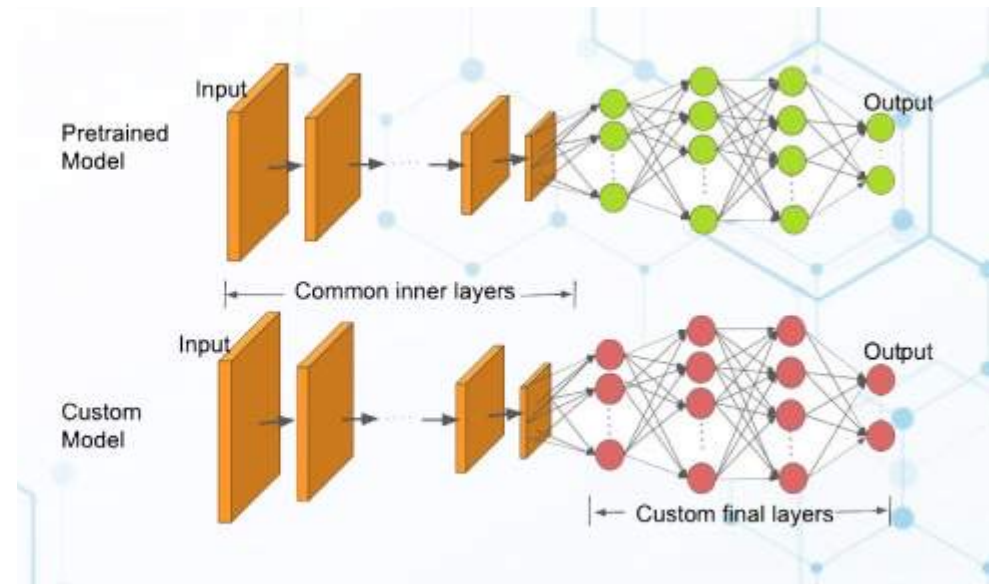
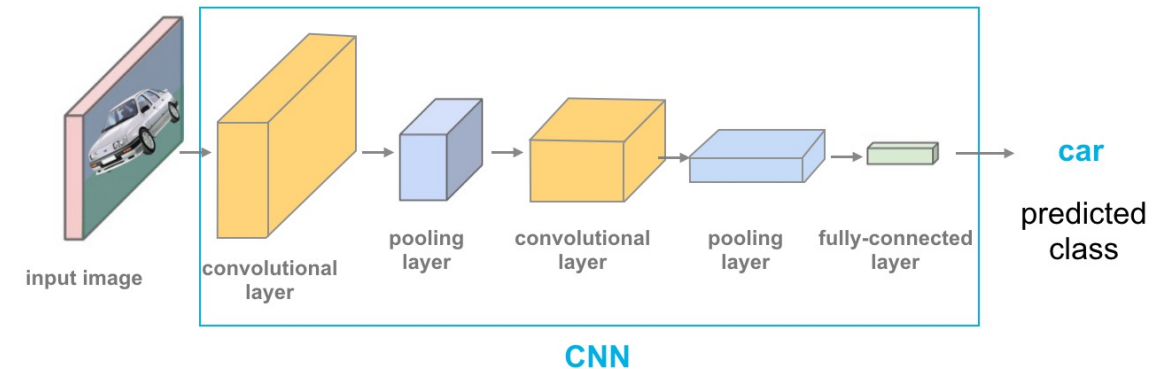
Population
estimates in
areas without
a census

Reduce
manual labor
and costs

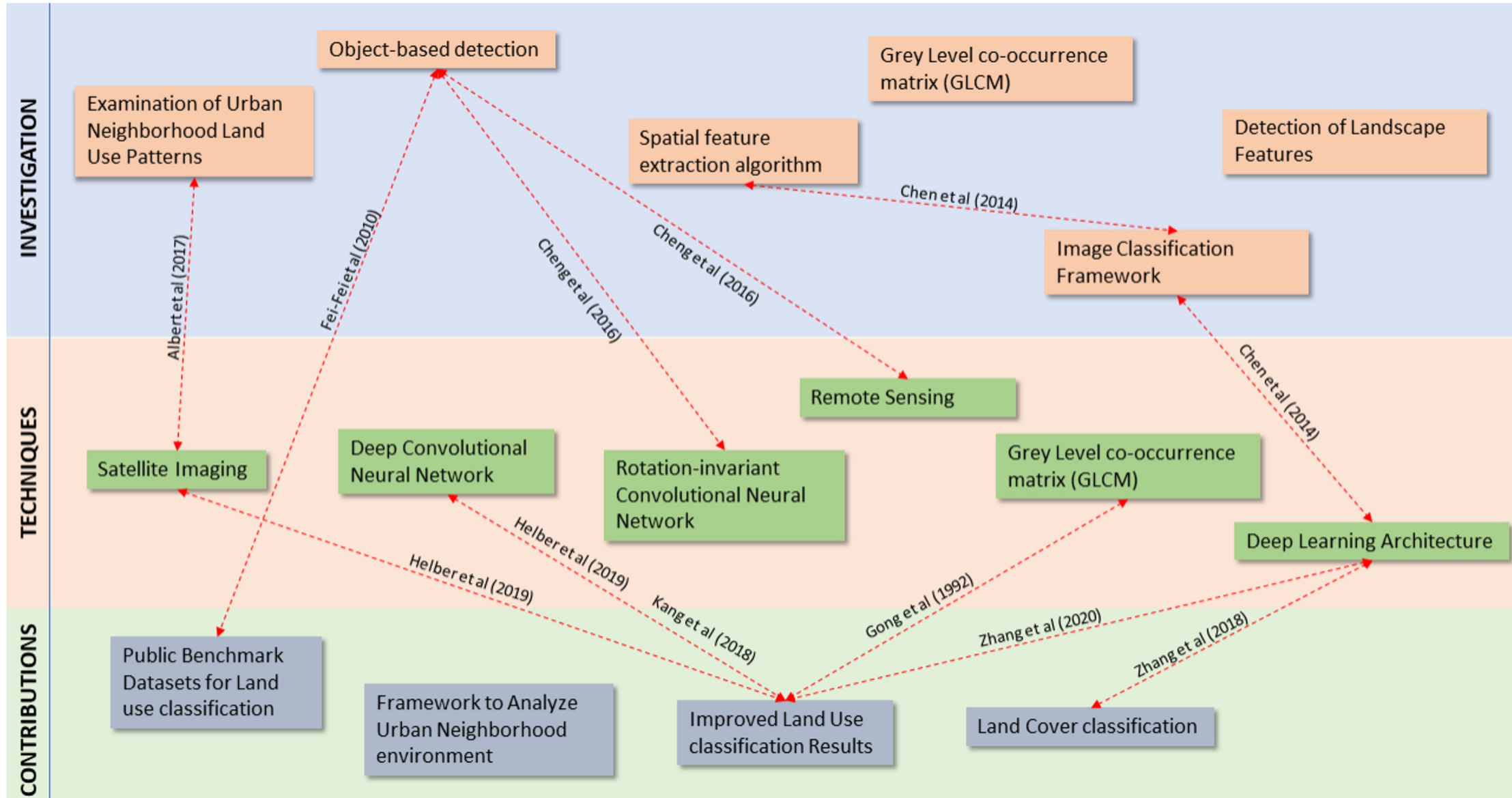
Identify
objects after
a disaster

Background: Basic Concepts

- **Neural network:** a type of machine learning model inspired by the structure and function of the human brain. It consists of interconnected processing units called neurons organized into layers which can be trained to recognize patterns in data and make predictions based on those patterns.
- **Deep Neural Network:** a type of artificial neural network (ANN) that has multiple hidden layers between the input and output layers and allow the network to learn more complex representations of the data
- **Convolutional Neural Network (CNN):** a type of neural network designed for image processing tasks and typically consists of convolutional layers, pooling layers, and fully connected layers.
- **Transfer Learning & Pre-Trained CNN :** Transfer learning is the process of taking a pre-trained model and fine-tuning it on a new, smaller dataset. Pre-trained CNN models are CNN models that have been trained on a large dataset, such as ImageNet, and can be used as a starting point for new image classification tasks.



Background: Literature Mapping



Background: Existing Research Gap

RESEARCH GAPS

1. How can an automated process using deep learning improve urban environment analysis.
2. How effective are CNN and transfer learning in improving the accuracy of image scene classification on the AID dataset?
3. Can the performance of CNN and transfer learning be further improved by combining multiple techniques or developing new ones?
4. How can the findings of this study be applied to real-world scenarios such as disaster management and urban planning?

INVESTIGATION

TECHNIQUES

CONTRIBUTIONS

Deep Convolutional
Neural Network

Rotation-invariant
Convolutional Neural
Network

Detection of Landscape
Features

Image Classification
Framework

Deep Learning Architecture

Framework to Analyze
Urban Neighborhood
environment

Improved Land Use
classification Results

Land Cover classification

FURTHER RESEARCH OPPORTUNITIES

- There are opportunities to apply DL techniques on a diverse and transferable database like AID, to develop a comprehensive and transferable framework with real life implications
- Developing an enriched urban land use classification model that can detect and classify a wide range of urban land use objects, have much practical implication in land use planning and modeling, resource management, emergency management planning, urban planning, historic preservation, construction management and landscape architecture.
- Evaluating the effectiveness of google map imagery compared to satellite images which requires less preprocessing, less time and cost.

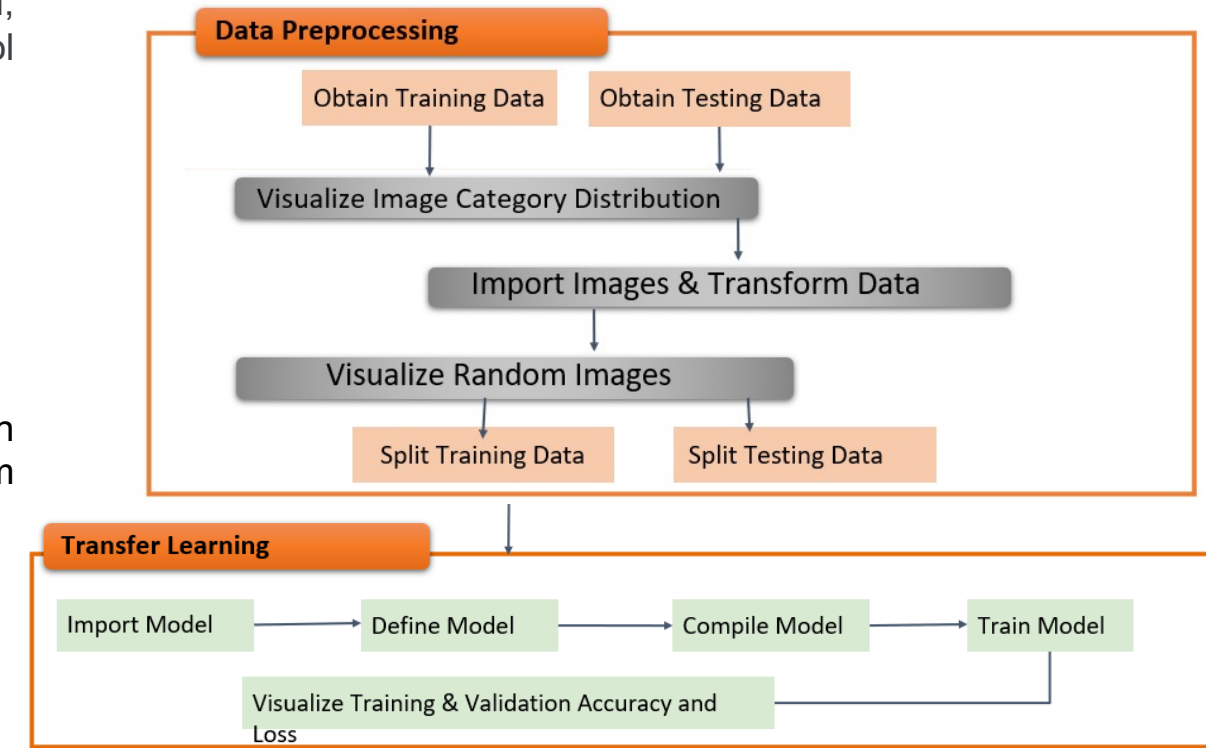
Approaches: Data & Method

- **Data :** A large-scale aerial image dataset (AID) labeled and collected from
 - ✓ Google earth imagery of different regions around the world
 - ✓ 30 urban land use scene types such as airport, bare land, beach, bridge, commercial, dense residential, desert, forest, industrial, meadow, medium residential, mountain, park, playground, river, school etc.
 - ✓ 200-400 images in each class

- **Method:**

- ✓ Data Preprocessing
- ✓ Developing a Custom CNN Model
- ✓ Apply pretrained CNN models- MobileNetV2, VGG 16 (Trained on ImageNet dataset initially, initialized model with the weights from training the model of ImageNet Dataset)
- ✓ Check model overfitting and underfitting
- ✓ Comparative Analysis of the model results
- ✓ Identifying the best fit model for image scene classification
- ✓ **Integral Development Environment (IDE)-** Google Colab
- ✓ **Hardware Accelerator-** GPU (NVIDIA V100)

Methodological Framework



Approaches: Data & Method

Data Preprocessing

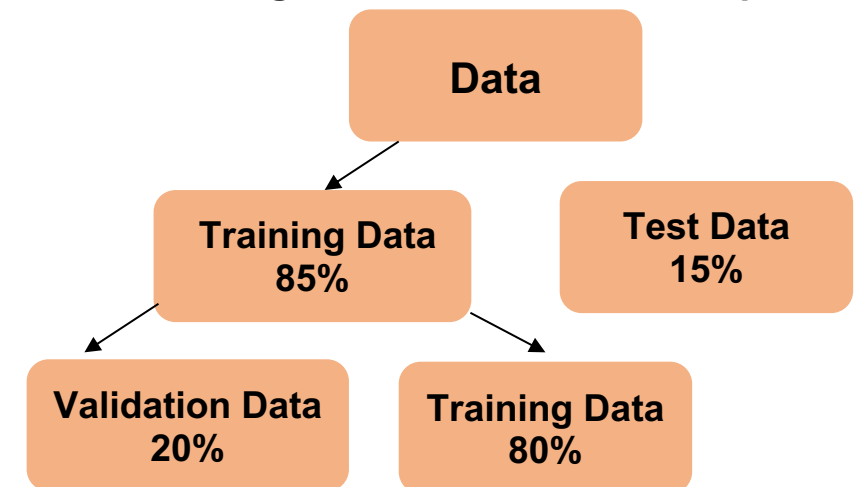
- **Total image dataset:** 10000
- **Size of Image:** 600 x 600
- **Resized Image size:** 224 x 224
- **Rescaled Image :** 1/255
- **Image Augmentation:** horizontal flip, vertical flip, 45-degree rotation

Model Training Hyper Parameters

- **Model Optimizer-** RMS Prop with momentum 0.9 (Used SGD and Adam Boost initially with no fixed momentum)
- **Learning Rate-** 0.001
- **Loss Function Minimized** – Categorical_CrossEntropy
- **Matric-** Accuracy
- **Training Batch Size** - 256
- **Validation Batch Size** – 128
- **Training Epochs** – 30
- **Callbacks Used** – Early stopping (monitored validation loss , patience= 5 epochs)



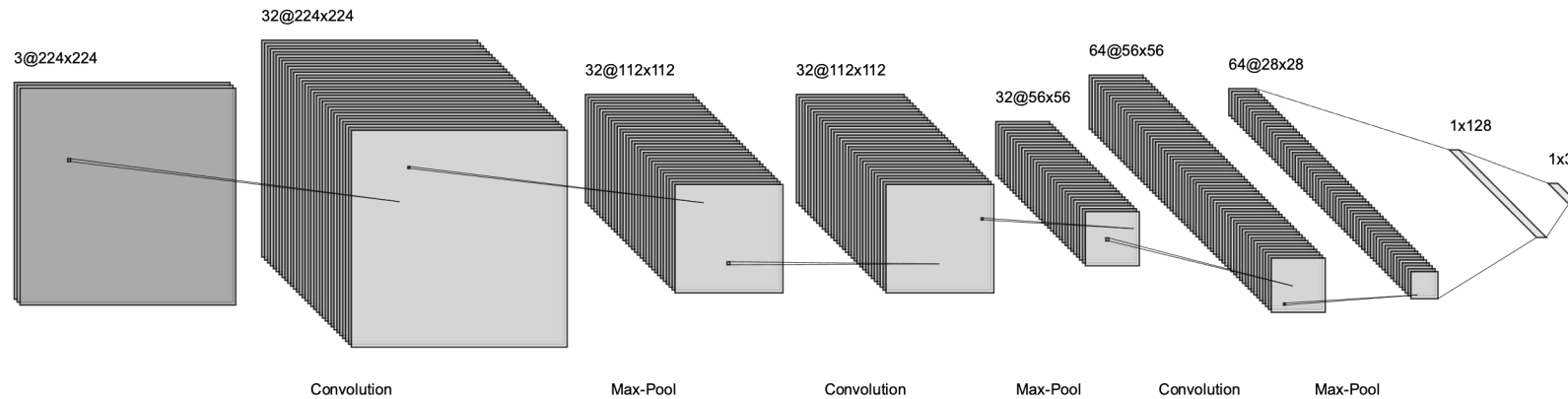
Training, Test & Validation Data Split



Approaches: Data & Method

Custom CNN Model Architecture

- **Applied standard architecture from literature**
- **Layers-** group of convolution layer and max pooling layer, then after flattening a dense layer is added with dropout regularization, finally an output layer is added with 30 nodes (one for each class)



| Layer (type) | Output Shape | Param # |
|--------------------------------|----------------------|---------|
| conv2d (Conv2D) | (None, 224, 224, 32) | 896 |
| max_pooling2d (MaxPooling2D) | (None, 112, 112, 32) | 0 |
| conv2d_1 (Conv2D) | (None, 112, 112, 32) | 9248 |
| max_pooling2d_1 (MaxPooling2D) | (None, 56, 56, 32) | 0 |
| conv2d_2 (Conv2D) | (None, 56, 56, 64) | 18496 |
| max_pooling2d_2 (MaxPooling2D) | (None, 28, 28, 64) | 0 |
| flatten (Flatten) | (None, 50176) | 0 |
| dense (Dense) | (None, 128) | 6422656 |
| dropout (Dropout) | (None, 128) | 0 |
| dense_1 (Dense) | (None, 30) | 3870 |

Figure : Custom CNN Model Architecture

Approaches: Data & Method

Transfer Learning Model Architecture (MobileNetV2 and VGG16)

- **1st step-** Base model from ImageNet dataset called through inference
- **2nd step-** Top layer of base model removed
- **3rd step -** each layer of base model defined to be non-trainable
- **4th step** – Final model is defined combining the base model followed by global average pooling, one dense layer, dropout regularization and final output layer.
- **5th Step-** Retrain the final model head for our classification task using the determined model training parameters

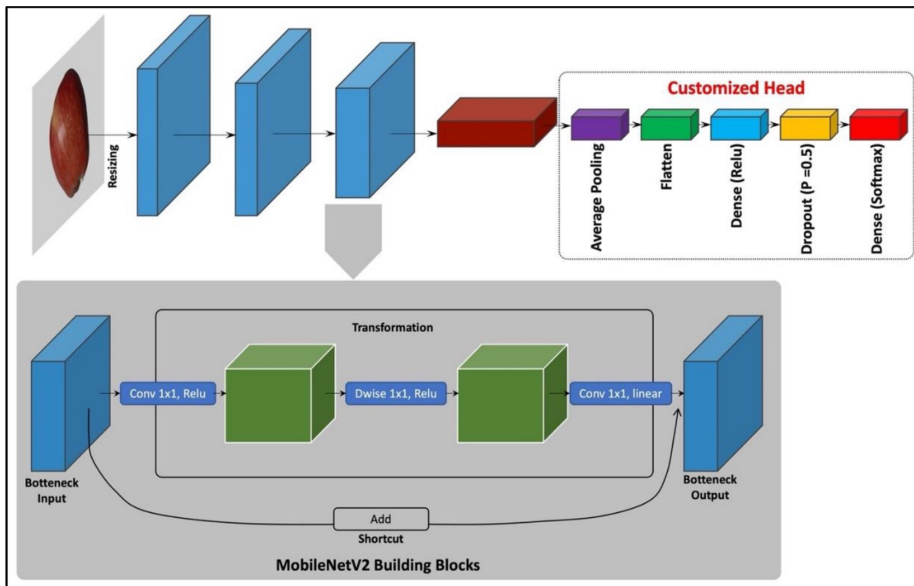


Figure : MobileNetV2 Model Architecture (Gulzar,2023)

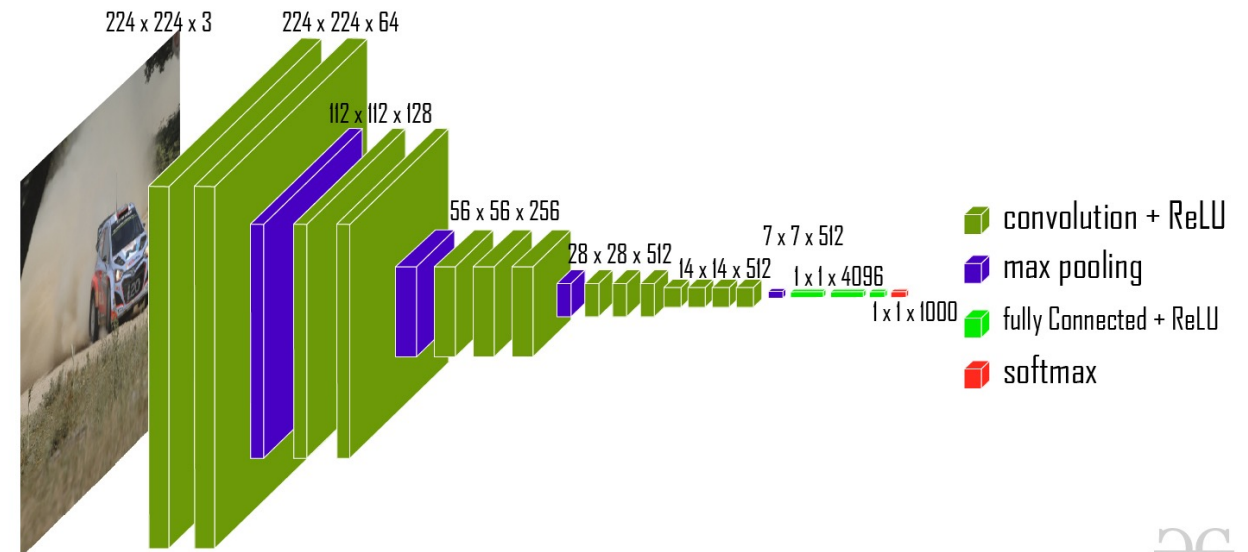
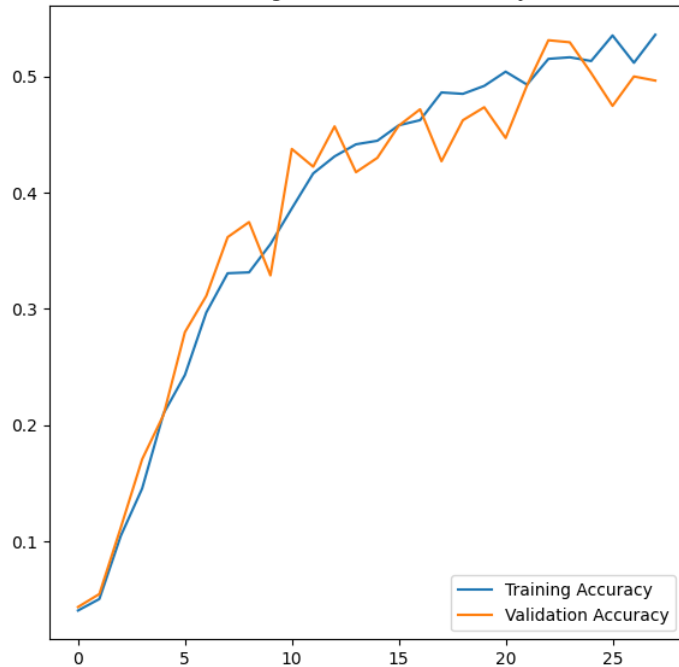


Figure : VGG16 Model Architecture (GeeksforGeeks,2023)

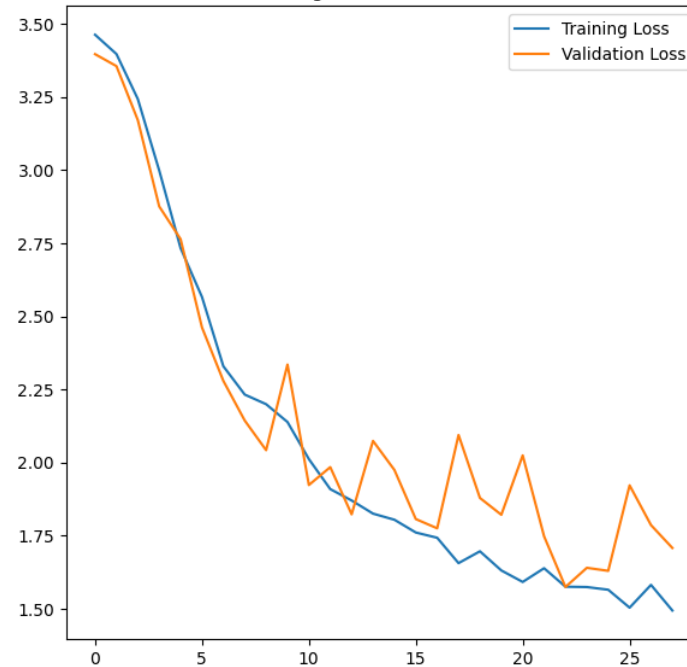
Results

Custom CNN Model Results

Training and Validation Accuracy



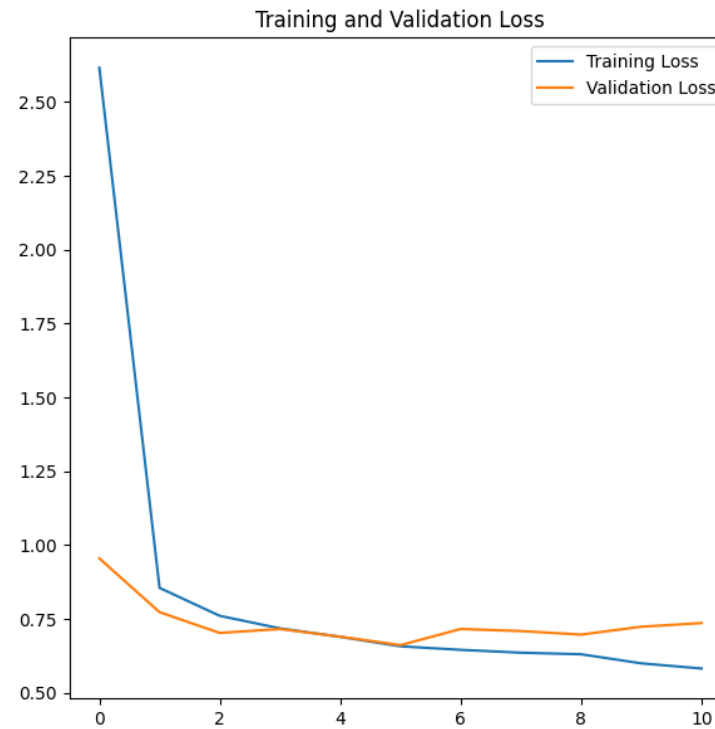
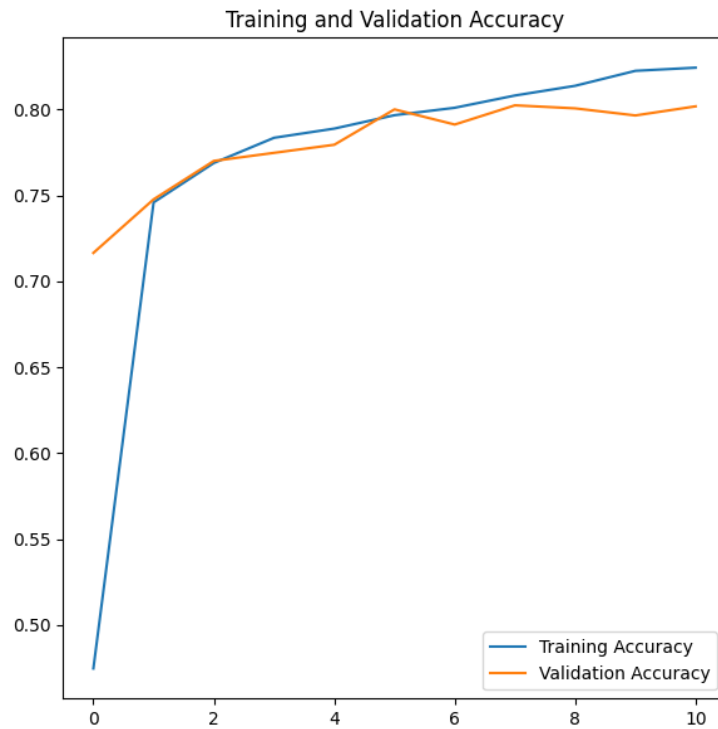
Training and Validation Loss



| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0 | 0.39 | 0.47 | 0.43 | 66 |
| 1 | 0.65 | 0.67 | 0.66 | 51 |
| 2 | 0.33 | 0.05 | 0.08 | 43 |
| 3 | 0.39 | 0.45 | 0.42 | 51 |
| 4 | 0.00 | 0.00 | 0.00 | 40 |
| 5 | 0.29 | 0.25 | 0.27 | 40 |
| 6 | 0.73 | 0.12 | 0.21 | 67 |
| 7 | 0.44 | 0.74 | 0.55 | 54 |
| 8 | 0.49 | 0.69 | 0.57 | 52 |
| 9 | 0.82 | 0.45 | 0.58 | 62 |
| 10 | 0.71 | 0.77 | 0.74 | 53 |
| 11 | 0.77 | 0.44 | 0.56 | 54 |
| 12 | 0.86 | 0.83 | 0.84 | 52 |
| 13 | 0.71 | 0.69 | 0.70 | 36 |
| 14 | 0.65 | 0.32 | 0.43 | 41 |
| 15 | 0.26 | 0.48 | 0.34 | 66 |
| 16 | 0.20 | 0.05 | 0.08 | 38 |
| 17 | 0.75 | 0.61 | 0.67 | 71 |
| 18 | 0.74 | 0.96 | 0.84 | 48 |
| 19 | 0.59 | 0.58 | 0.59 | 62 |
| 20 | 0.34 | 0.25 | 0.29 | 48 |
| 21 | 0.35 | 0.65 | 0.45 | 23 |
| 22 | 0.33 | 0.64 | 0.44 | 36 |
| 23 | 0.67 | 0.38 | 0.48 | 37 |
| 24 | 0.71 | 0.62 | 0.66 | 58 |
| 25 | 0.26 | 0.37 | 0.31 | 57 |
| 26 | 0.39 | 0.59 | 0.47 | 51 |
| 27 | 0.59 | 0.95 | 0.73 | 40 |
| 28 | 0.24 | 0.28 | 0.26 | 57 |
| 29 | 0.81 | 0.76 | 0.79 | 46 |
| accuracy | | | 0.50 | 1500 |
| macro avg | 0.52 | 0.50 | 0.48 | 1500 |
| weighted avg | 0.53 | 0.50 | 0.49 | 1500 |

Results

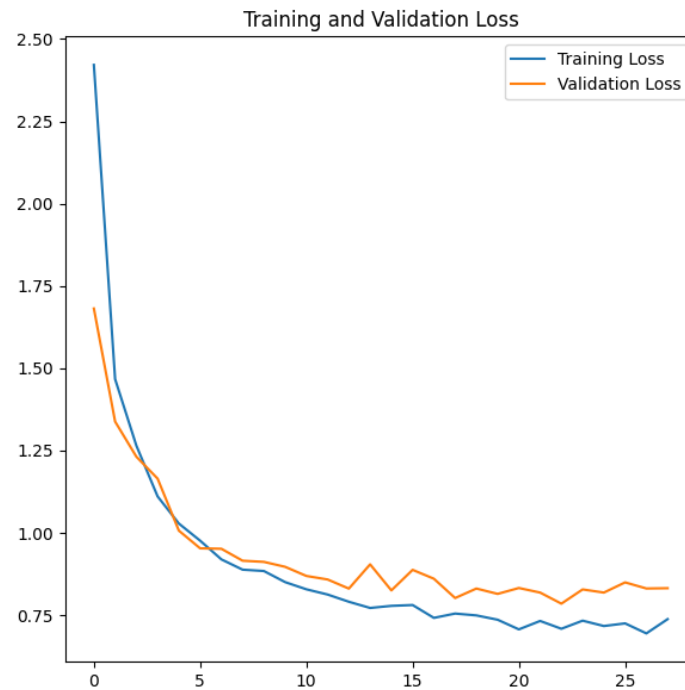
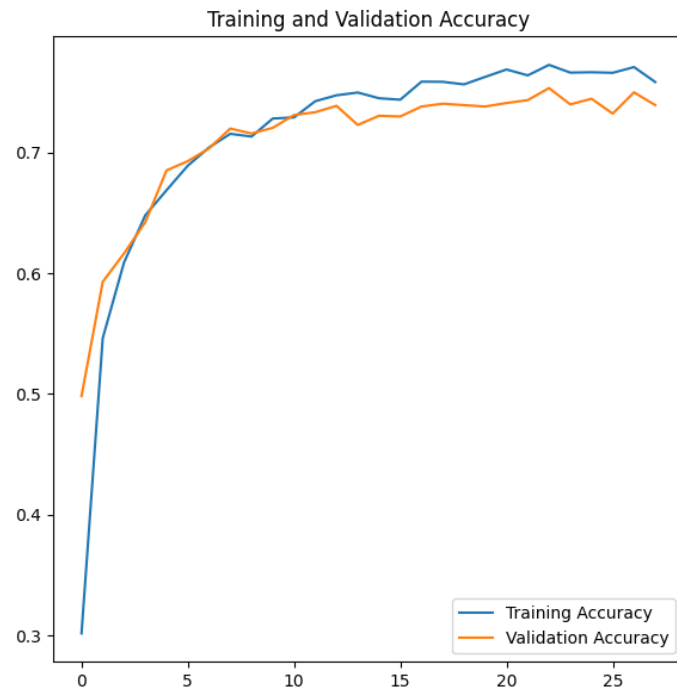
MobileNetV2 Model Results



| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0 | 0.88 | 0.85 | 0.86 | 66 |
| 1 | 0.86 | 0.75 | 0.80 | 51 |
| 2 | 0.58 | 0.74 | 0.65 | 43 |
| 3 | 0.87 | 0.92 | 0.90 | 51 |
| 4 | 0.62 | 0.50 | 0.56 | 40 |
| 5 | 0.86 | 0.75 | 0.80 | 40 |
| 6 | 0.74 | 0.60 | 0.66 | 67 |
| 7 | 0.85 | 0.83 | 0.84 | 54 |
| 8 | 0.81 | 0.67 | 0.74 | 52 |
| 9 | 0.80 | 0.82 | 0.81 | 62 |
| 10 | 0.81 | 0.87 | 0.84 | 53 |
| 11 | 0.98 | 1.00 | 0.99 | 54 |
| 12 | 0.98 | 0.85 | 0.91 | 52 |
| 13 | 0.94 | 0.83 | 0.88 | 36 |
| 14 | 0.91 | 0.78 | 0.84 | 41 |
| 15 | 0.85 | 0.79 | 0.82 | 66 |
| 16 | 0.63 | 0.84 | 0.72 | 38 |
| 17 | 0.89 | 0.76 | 0.82 | 71 |
| 18 | 0.77 | 0.98 | 0.86 | 48 |
| 19 | 0.95 | 0.89 | 0.92 | 62 |
| 20 | 0.72 | 0.58 | 0.64 | 48 |
| 21 | 0.73 | 0.48 | 0.58 | 23 |
| 22 | 0.40 | 0.89 | 0.55 | 36 |
| 23 | 0.89 | 0.92 | 0.91 | 37 |
| 24 | 0.96 | 0.93 | 0.95 | 58 |
| 25 | 0.93 | 1.00 | 0.97 | 57 |
| 26 | 0.91 | 0.98 | 0.94 | 51 |
| 27 | 0.76 | 0.97 | 0.86 | 40 |
| 28 | 0.80 | 0.61 | 0.69 | 57 |
| 29 | 0.96 | 0.98 | 0.97 | 46 |
| accuracy | | | 0.82 | 1500 |
| macro avg | 0.82 | 0.81 | 0.81 | 1500 |
| weighted avg | 0.83 | 0.82 | 0.82 | 1500 |

Results

VGG16 Model Results



| | precision | recall | f1-score | support |
|--------------|-----------|--------|----------|---------|
| 0 | 0.70 | 0.86 | 0.77 | 66 |
| 1 | 0.80 | 0.80 | 0.80 | 51 |
| 2 | 0.53 | 0.63 | 0.57 | 43 |
| 3 | 0.73 | 0.84 | 0.78 | 51 |
| 4 | 0.52 | 0.55 | 0.54 | 40 |
| 5 | 0.81 | 0.55 | 0.66 | 40 |
| 6 | 0.62 | 0.57 | 0.59 | 67 |
| 7 | 0.87 | 0.85 | 0.86 | 54 |
| 8 | 0.81 | 0.58 | 0.67 | 52 |
| 9 | 0.60 | 0.81 | 0.68 | 62 |
| 10 | 0.71 | 0.94 | 0.81 | 53 |
| 11 | 0.91 | 0.93 | 0.92 | 54 |
| 12 | 0.71 | 0.87 | 0.78 | 52 |
| 13 | 0.68 | 0.69 | 0.68 | 36 |
| 14 | 0.81 | 0.63 | 0.71 | 41 |
| 15 | 0.81 | 0.76 | 0.78 | 66 |
| 16 | 0.69 | 0.66 | 0.68 | 38 |
| 17 | 0.71 | 0.70 | 0.71 | 71 |
| 18 | 0.88 | 0.62 | 0.73 | 48 |
| 19 | 0.82 | 0.97 | 0.89 | 62 |
| 20 | 0.80 | 0.69 | 0.74 | 48 |
| 21 | 0.55 | 0.74 | 0.63 | 23 |
| 22 | 0.58 | 0.83 | 0.68 | 36 |
| 23 | 0.96 | 0.70 | 0.81 | 37 |
| 24 | 0.98 | 0.74 | 0.84 | 58 |
| 25 | 0.91 | 0.93 | 0.92 | 57 |
| 26 | 0.96 | 0.94 | 0.95 | 51 |
| 27 | 0.89 | 0.78 | 0.83 | 40 |
| 28 | 0.78 | 0.54 | 0.64 | 57 |
| 29 | 0.95 | 0.89 | 0.92 | 46 |
| accuracy | | | 0.76 | 1500 |
| macro avg | 0.77 | 0.75 | 0.75 | 1500 |
| weighted avg | 0.77 | 0.76 | 0.76 | 1500 |

Results

Model Results Comparison

- Among the three models, MobileNetV2 outperformed the custom CNN model and VGG16
- VGG16 provided good and comparable classification result
- The performance of the CNN model does not demonstrate significant improvement compared to the transfer learning model.
- Training and validation accuracy and loss increased and decreased with similar trend for all three models.
- The custom model indicates slight underfitting and the two transfer learning model shows slight overfitting (may be due to randomness)

| Models | Precision (Macro Avg) | Recall (Macro Avg) | F1-score (Macro Avg) | Accuracy |
|-------------|--------------------------|-----------------------|-------------------------|----------|
| Custom CNN | 0.52 | 0.50 | 0.48 | 0.50 |
| MobileNetV2 | 0.82 | 0.81 | 0.81 | 0.82 |
| VGG16 | 0.77 | 0.75 | 0.75 | 0.76 |

Conclusion & Discussion

- Aerial imagery has become increasingly popular in recent years due to the availability of satellite and drone technology. These images are useful in various fields such as agriculture, urban planning, and military surveillance.
- Convolutional Neural Networks (CNNs) have been shown to be effective in image classification tasks, and transfer learning has been demonstrated to improve classification accuracy on small datasets Albert et al., 2017; Cassidy et al., 2010; Liu et al., 2017; Patino & Duque, 2013; Zhang et al., 2018; Zhang et al., 2019).
- Transfer learning models are found to be performing better for image classification due to their better generalization capability
- Limitation of computational resource and time prevented further exploration of Custom CNN model improvement. Further improvement on the design architecture and model hyperparameter might improve the custom CNN model performance and can be direction for future research.
- This research will contribute to the development of image classification techniques for aerial imagery, which has numerous applications in various fields. The use of CNN and transfer learning will enhance the accuracy of image classification on the AID dataset, which will be useful in real-world scenarios. The outcomes of this research can be used to improve the classification of scenes in aerial imagery and to develop more accurate and efficient models for image classification tasks.

References

- Albert, A., Kaur, J., & Gonzalez, M. C. (2017, August). Using convolutional networks and satellite imagery to identify patterns in urban environments at a large scale. In *Proceedings of the 23rd ACM SIGKDD international conference on knowledge discovery and data mining* (pp. 1357-1366).
- AID: A scene classification dataset. (2022, May 31). Kaggle. https://www.kaggle.com/datasets/jiayuanhengala/aid-scene-classification-datasets?fbclid=IwAR2baoDJTC-kktTTidMuKOaxToE4IC_PJfoPsO1yJ_Do5Q4WAc5YB-xQt5s
- Cheng, G., Zhou, P., & Han, J. (2016). Learning rotation-invariant convolutional neural networks for object detection in VHR optical remote sensing images. *IEEE Transactions on Geoscience and Remote Sensing*, 54(12), 7405-7415.
- Chew, R. F., Amer, S., Jones, K., Unangst, J., Cajka, J., Allpress, J., & Bruhn, M. (2018). Residential scene classification for gridded population sampling in developing countries using deep convolutional neural networks on satellite imagery. *International Journal of Health Geographics*, 17(1), 1-17.
- GeeksforGeeks. (2023). VGG 16 CNN model. GeeksforGeeks. <https://www.geeksforgeeks.org/vgg-16-cnn-model/>
- Gong, P., Marceau, D. J., & Howarth, P. J. (1992). A comparison of spatial feature extraction algorithms for land-use classification with SPOT HRV data. *Remote sensing of environment*, 40(2), 137-151.
- Gulzar, Y. (2023). Fruit Image Classification Model Based on MobileNetV2 with Deep Transfer Learning Technique. *Sustainability*, 15(3), 1906.
- Helber, P., Bischke, B., Dengel, A., & Borth, D. (2019). Eurosat: A novel dataset and deep learning benchmark for land use and land cover classification. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 12(7), 2217-2226.
- Hu, Y., Li, W., Wright, D., Aydin, O., Wilson, D., Maher, O., & Raad, M. (2019). Artificial intelligence approaches. *arXiv preprint arXiv:1908.10345*.
- Fei-Fei, L., Deng, J., & Li, K. (2009). ImageNet: Constructing a large-scale image database. *Journal of vision*, 9(8), 1037-1037.
- Chen, Y., Lin, Z., Zhao, X., Wang, G., & Gu, Y. (2014). Deep learning-based classification of hyperspectral data. *IEEE Journal of Selected topics in applied earth observations and remote sensing*, 7(6), 2094-2107.
- Kang, Y., Cho, N., Yoon, J., Park, S., & Kim, J. (2021). Transfer learning of a deep learning model for exploring tourists' urban image using geotagged photos. *ISPRS International Journal of Geo-Information*, 10(3), 137.
- Li, X., & Zhang, C. (2016, September). Urban Land Use Information Retrieval Based on Scene Classification of Google Street View Images. In *SDW@ GIScience* (pp. 41-46).
- Martins, C. (2022, September 23). *Multiclass Image Classification — Hands-On with Keras and TensorFlow*. Medium. <https://pub.towardsai.net/multiclass-image-classification-hands-on-with-keras-and-tensorflow-e1cf434f3467>
- Zhang, C., Sargent, I., Pan, X., Li, H., Gardiner, A., Hare, J., & Atkinson, P. M. (2019). Joint Deep Learning for land cover and land use classification. *Remote sensing of environment*, 221, 173-187.
- Zhang, C., Harrison, P. A., Pan, X., Li, H., Sargent, I., & Atkinson, P. M. (2020). Scale Sequence Joint Deep Learning (SS-JDL) for land use and land cover classification. *Remote Sensing of Environment*, 237, 111593.