Deep Forest

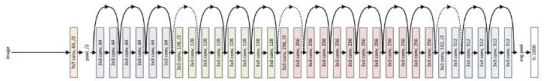
CNN Models to Analyse Forest Image and Deforestation

Goal

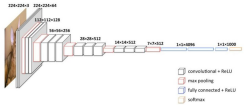
- Analyse Amazon Rainforest Images to predict ‘Deforestation’ and ‘Pollution’

How to achieve

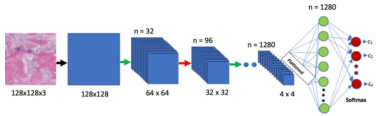
- Apply 4 different CNN models
  - 3 Pre-Trained Models
  - 1 Model from scratch



ResNet50



VGG

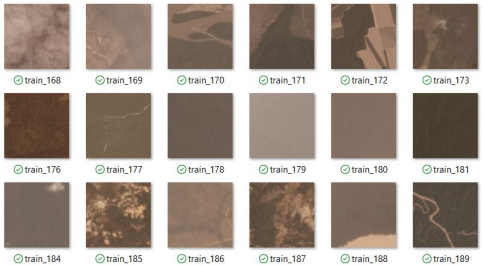


MobileNet

DeepForestNet

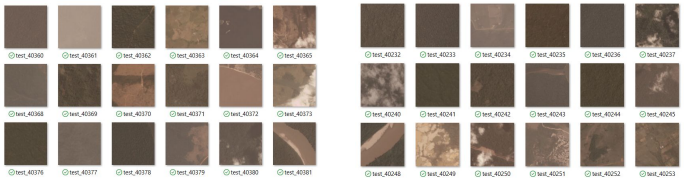
Our Dataset

- **Training Dataset**
  - ≈40,500 satellite images
  - Label embedded (tags) in CSV



	A	B
1	image_name	tags
2	train_0	haze primary
3	train_1	agriculture clear primary water
4	train_2	clear primary
5	train_3	clear primary
6	train_4	agriculture clear habitation primary road
7	train_5	haze primary water
8	train_6	agriculture clear cultivation primary water
9	train_7	haze primary
0	train_8	agriculture clear cultivation primary
1	train_9	agriculture clear cultivation primary road
2	train_10	agriculture clear primary slash_burn water
3	train_11	clear primary water
4	train_12	cloudy
5	train_13	clear primary
6	train_14	cloudy
7	train_15	clear primary

- **Testing Dataset**
  - ≈40,500 satellite images

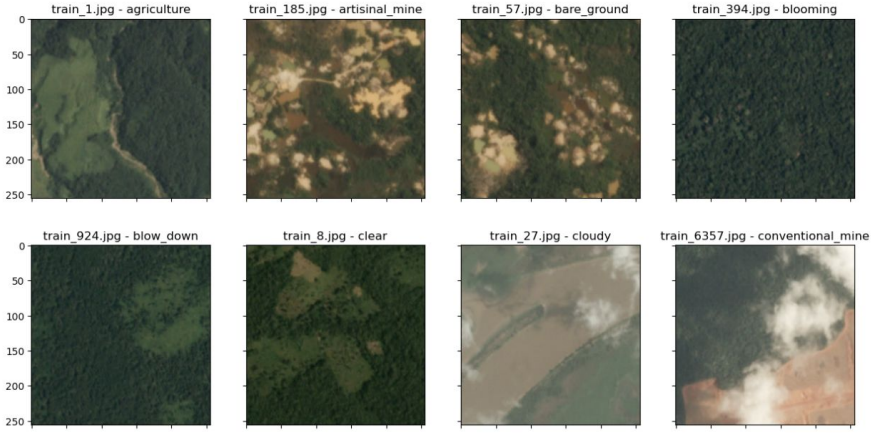


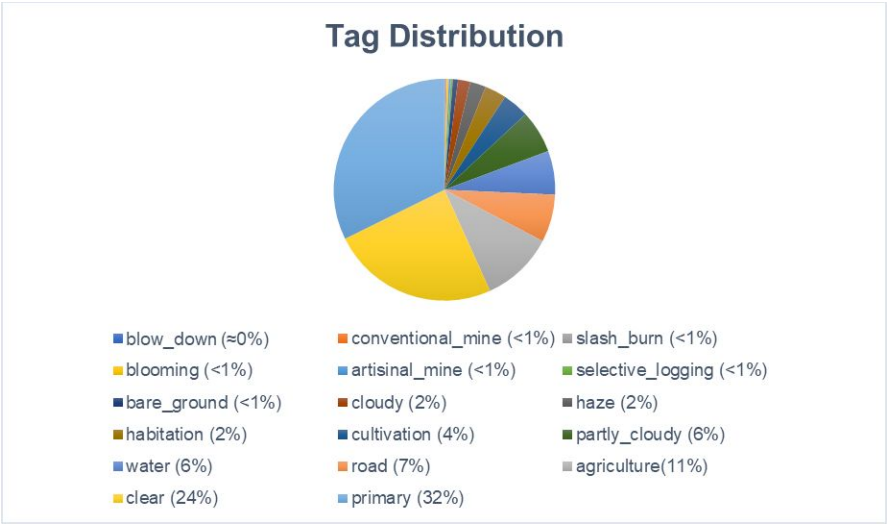
Explore Data

One-hot encode

Out[21]:

	agriculture	artisinal_mine	bare_ground	blooming	blow_down	clear	cloudy	conventional_mine	cultivation	habitation
0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	1	0	0	0	0
2	0	0	0	0	0	1	0	0	0	0
3	0	0	0	0	0	1	0	0	0	0
4	1	0	0	0	0	1	0	0	0	1
...	...	...	...	...	...	...	...	...	...	...
40474	0	0	0	0	0	1	0	0	0	0
40475	0	0	0	0	0	0	1	0	0	0
40476	1	0	0	0	0	1	0	0	0	0
40477	1	0	0	0	0	1	0	0	0	0
40478	1	0	0	0	0	0	0	0	1	0





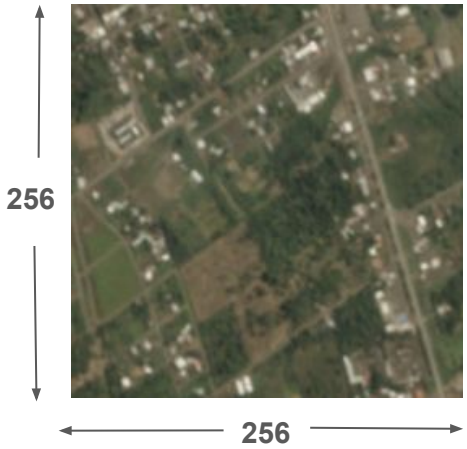
According to NASA Earth Observatory,

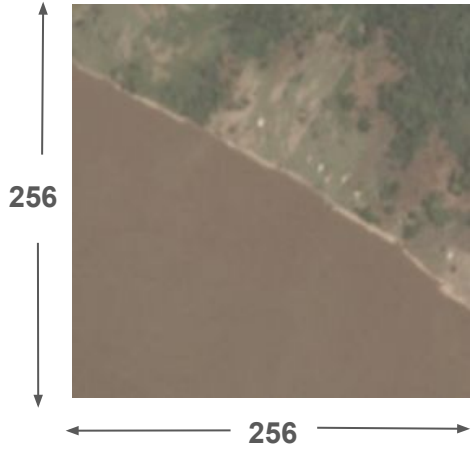
Deforestation ⊂ {the expansion of infrastructure and urbanization}
Pollution ⊂ {Farmers slash burn the remaining forest}

‘Habitation’ and ‘Slash_Burn’

Data Preprocessing

- Image Dimension





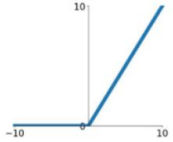
- Missing values NA in Label Encoding CSV

For all models: TensorFlow

- Activation Function

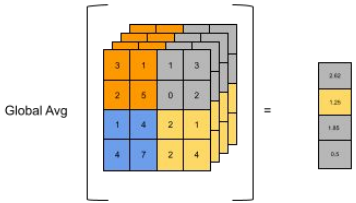
ReLU

$\max(0, x)$



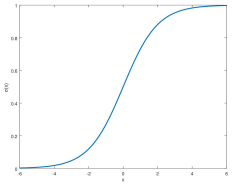
- Pooling

GlobalAveragePooling



- 3 Epochs

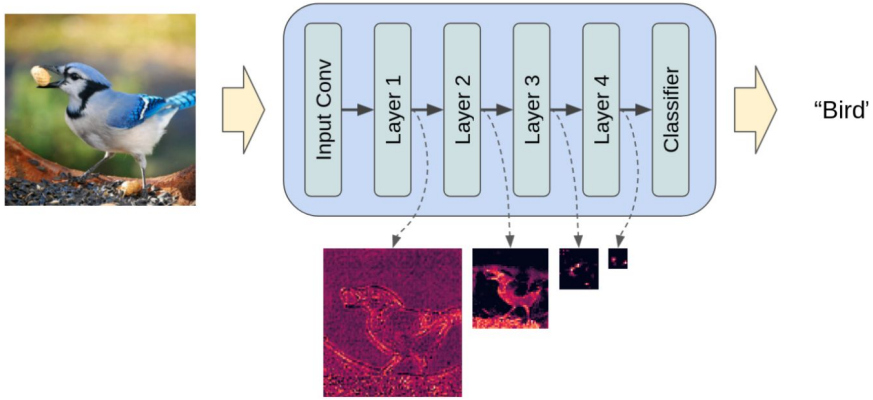
- Desired Outcome Sigmoid



‘Deforestation’
‘Pollution’

- Optimizer = Adam
- Binary Cross Entropy Loss
- AUC

Training Model



ResNet50

Principales Caractéristiques :

- Composé de 50 couches de convolution
- Connexions Skip
- Pré-entraînement sur ImageNet (d’image et classification)
- Adaptabilité

Resultats

```
'Epoch 1': {'val_auc': 0.8824, 'val_loss': 0.1599},
'Epoch 2': {'val_auc': 0.8821, 'val_loss': 0.1533},
'Epoch 3': {'val_auc': 0.8789, 'val_loss': 0.1642}
```

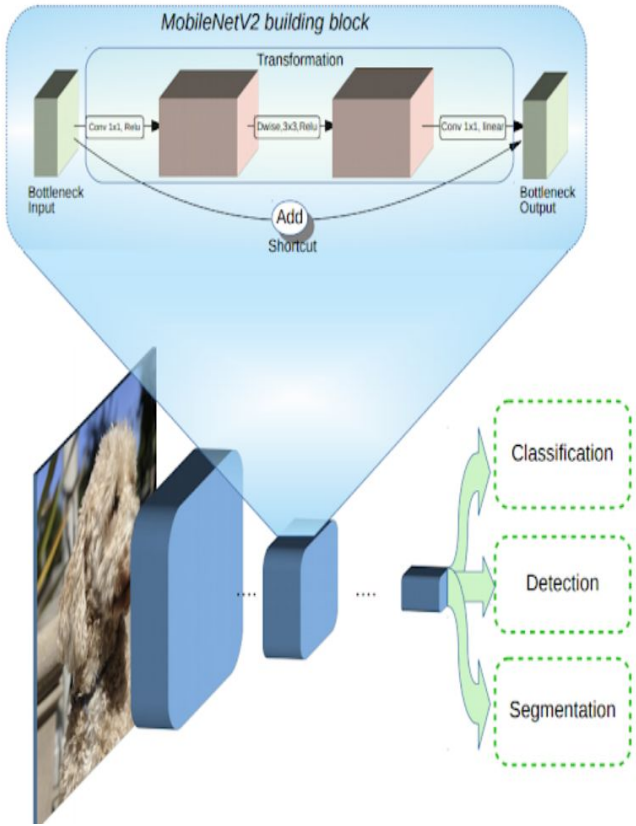
Training Model

MobilNetV2

- **Architecture Légère** : faible puissance de Calcul
- **Factorisation Profonde** : depthwise separable convolutior
- **Blocs Linéaires** : réduire le nombre de paramètres
- **Précision Équilibrée** : entre la précision et l'efficacité
- **Pré-Entraînement et Fine-Tuning** : ImageNet, tâches spécifiques

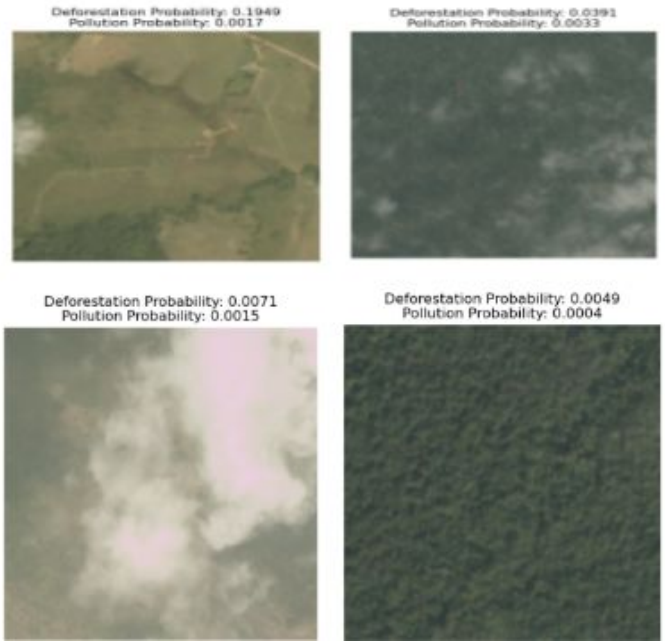
Resultats :

```
'Epoch 1': {'val_auc': 0.9377, 'val_loss': 0.1140},
'Epoch 2': {'val_auc': 0.9545, 'val_loss': 0.0910},
'Epoch 3': {'val_auc': 0.9531, 'val_loss': 0.0915}
```

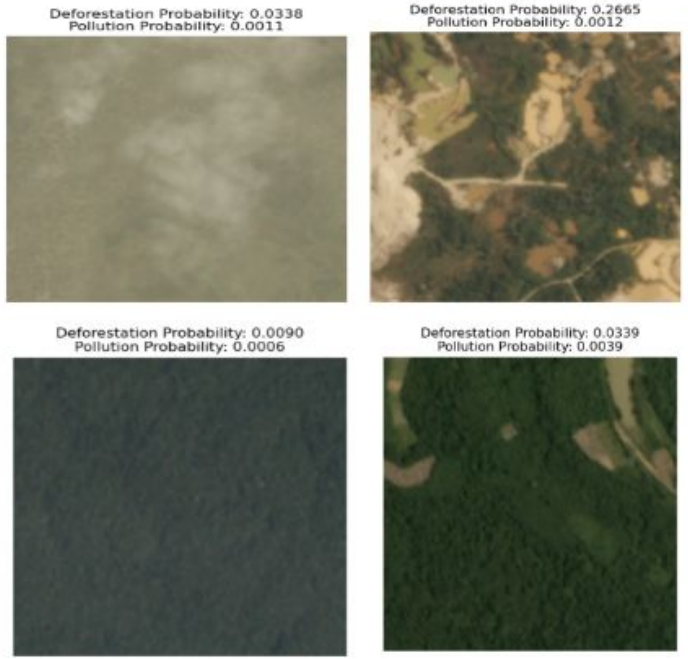


Training Model

MobilNetV2



ResNet50



VGG Model

•Choix du Modèle VGG16 : 92,7% de précision

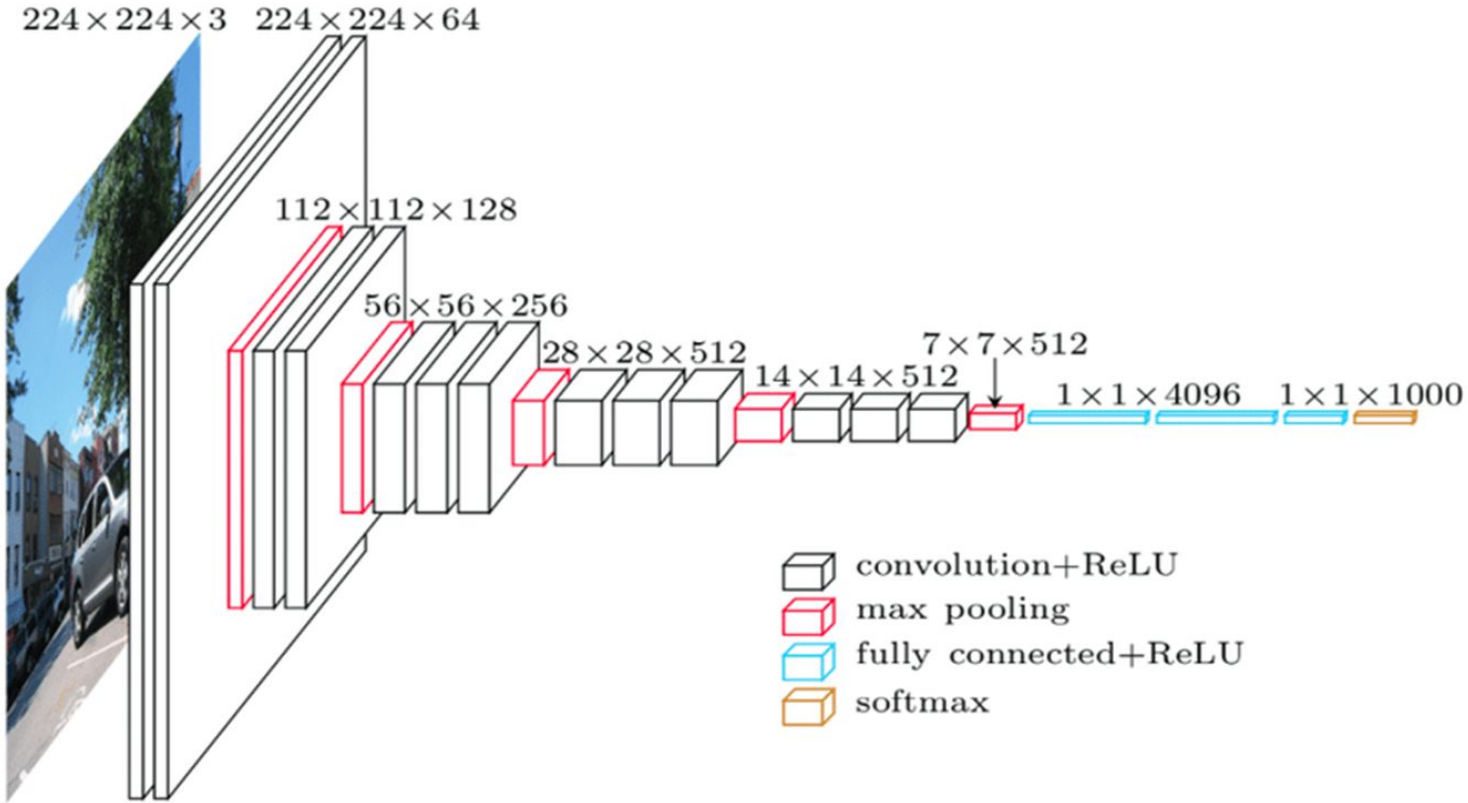
•Avantages VGG vs ResNet

- Simplicité de l'Architecture
- Transfert d'Apprentissage Plus Rapide
- Facilité d'Interprétation

•VGG vs MobileNet :

- Meilleure performance sur des jeux de données complexes.
- Transfert d'apprentissage plus puissant grâce à sa structure plus profonde.
- Facilité d'interprétation des caractéristiques extraites

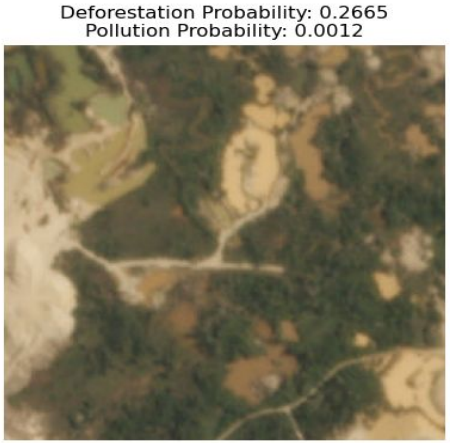
Architecture VGG16

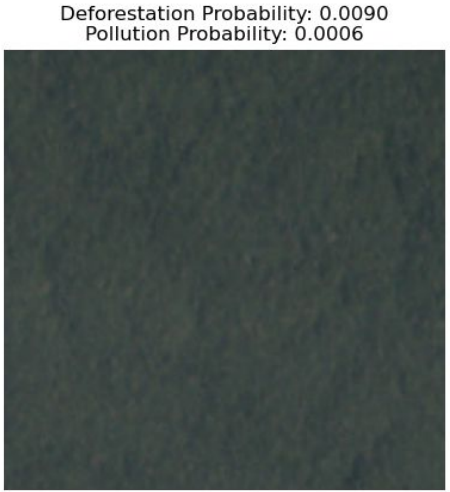


Résultats VGG16

Epoque	AUC (Training)	AUC (Validation)	Loss (Training)	Loss (Validation)
1	0.9155	0.9309	0.1133	0.1020
2	0.9308	0.9407	0.1028	0.1035
3	0.9327	0.9335	0.1019	0.1009

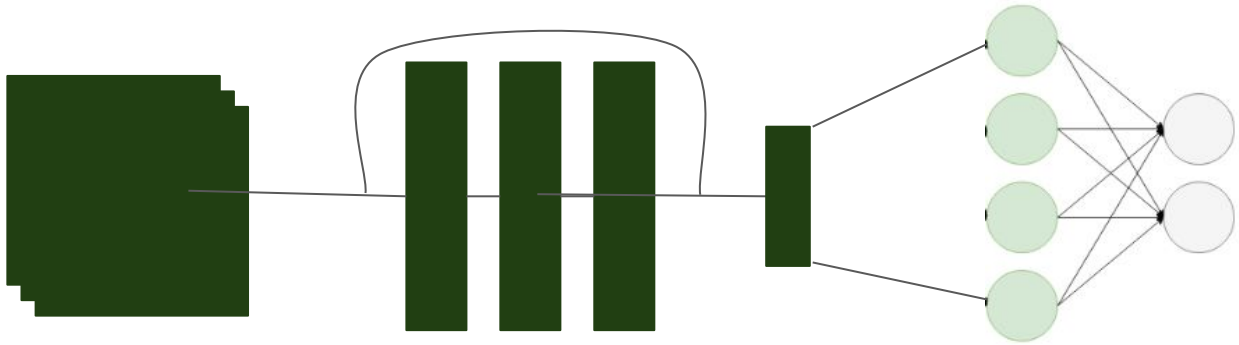
- Augmentation de AUC au fil des époques
- Diminution de Loss au fil des époques





DeepForestNet

- Uses CNN Residual Network Architecture



Convolutional

Conv2D
Batch Normalization
Relu and Max Pooling

Residual

3 Conv Layers

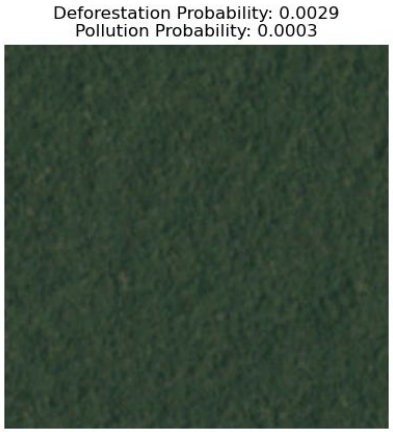
Global Average Pooling

Fully-Connected Layer

Use Dropout for regularization
(Trials and Errors)

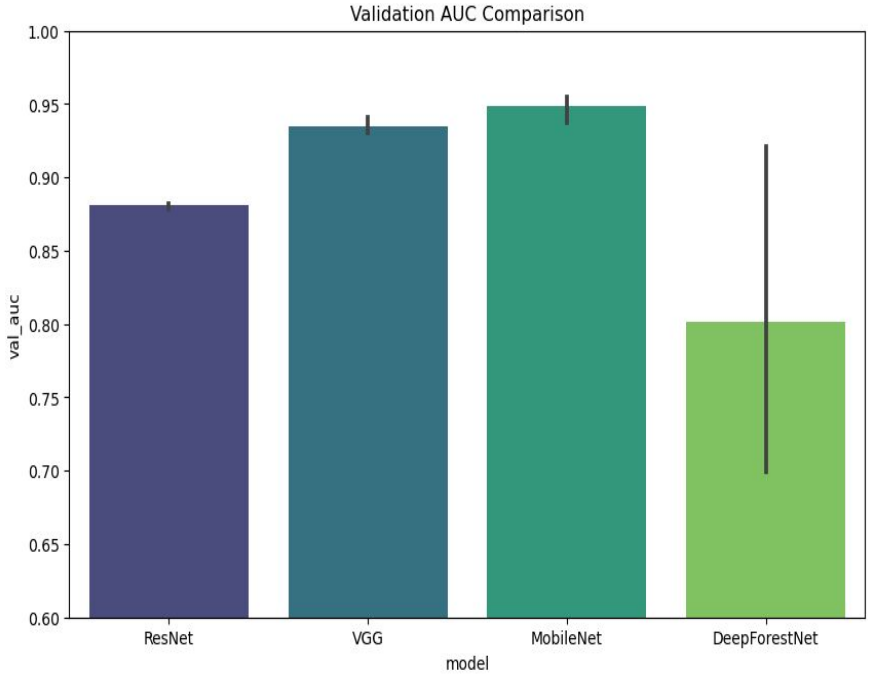
Compile Model

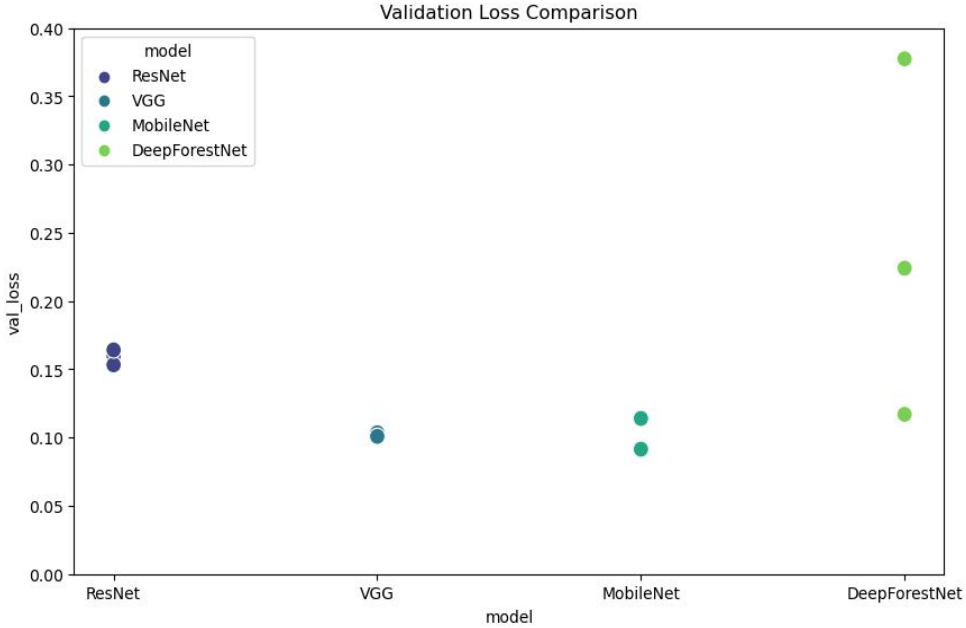
Learning rate α = 0.001



output

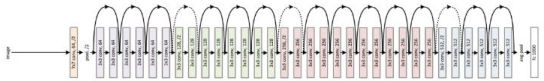
Conclusion - Model Comparison



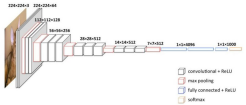


Conclusion

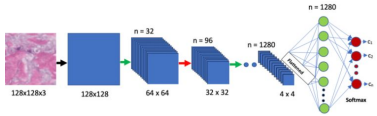
- **Training the Model with another Dataset (maybe)**



ResNet50



VGG



MobileNet

DeepForestNet