Plant Seedlings Classification

Outline

- Environment
- Data Introduction
- Preprocessing
- Method
- Neural Network
- Evaluation
- Result

Environment

- Cuda 11.6
- Numpy 1.21.6
- Pandas 1.1.5
- Tensorflow 2.10.0
- Matplotlib 3.5.3
- Scikit-learn 1.0.2

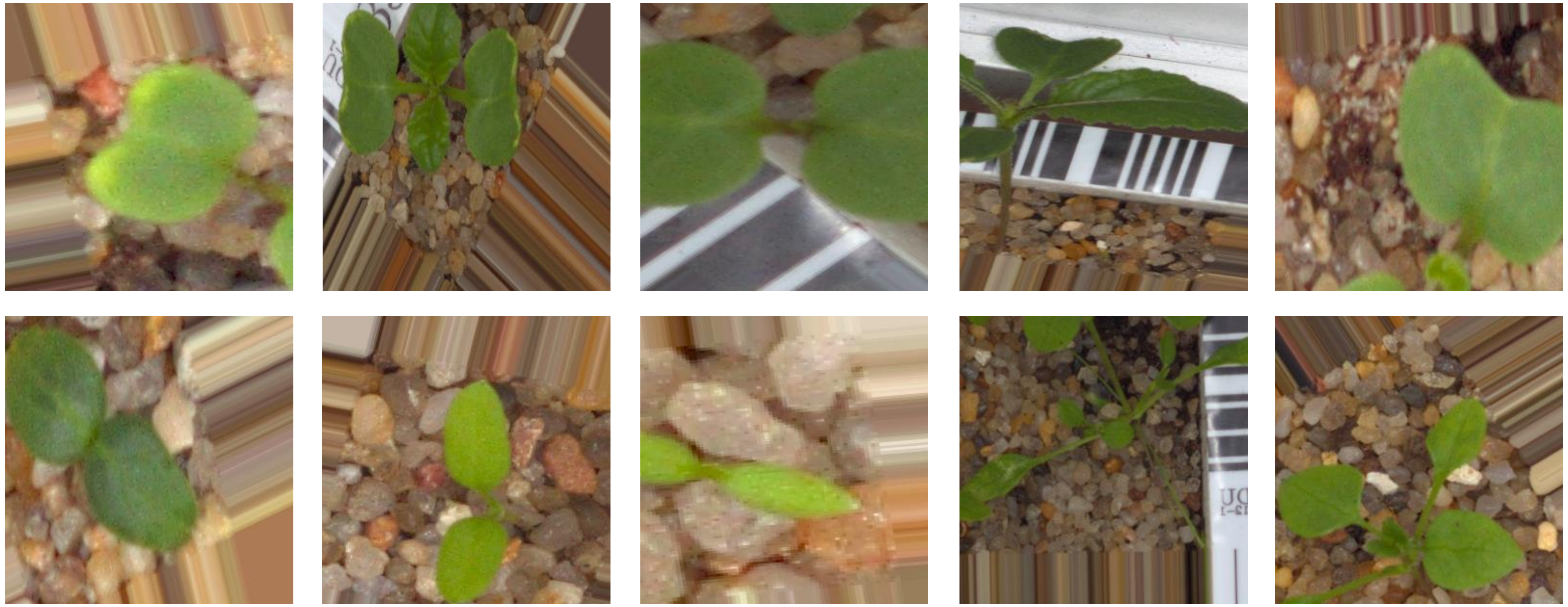
Data Introduction - 12 Classes

- Black-grass(263 picture)
- Charlock (390 picture)
- Cleavers (287 picture)
- Common Chickweed (611 picture)
- Common wheat (221 picture)
- Fat Hen (475 picture)
- Loose Silky-bent (654 picture)
- Maize (221 picture)
- Scentless Mayweed (516 picture)
- Shepherds Purse (231 picture)
- Small-flowered Cranesbill (496 picture)
- Sugar beet (385 picture)
- The total number of training data is **4750** images.

Preprocessing

- Resize data into same size (299,299)
- Shuffle
- One-hot encoding

- Data augmentation
  - rotation_range
  - width_shift_range
  - height_shift_range
  - shear_range
  - zoom_range
  - horizontal_flip
  - vertical_flip
  - brightness_range



Method

- Due to the limited dataset size, it may be challenging to effectively split the data into training and validation sets. In such cases, using all the data for training maximizes the utilization of the limited data resources.
- To increase data diversity and enhance model generalization, transformations such as rotation, translation, and scaling were applied to the training data. This allows the model to learn robust features that are invariant to different transformations, thereby addressing the issue of invariance.

Network Architecture

- DenseNet121
- DenseNet169
- DenseNet201
- Efficientnetb0
- Efficientnetb7
- MobileNetV3 Small
- MobileNetV3 Large
- Restnet50
- Restnet101
- InceptionResNetV2

Evaluation

- Submissions are evaluated on **MeanFScore**,
- which at Kaggle is actually a micro-averaged F1-score.

- Given positive/negative rates for each class k, the resulting score is computed this way:

$$Precision_{micro} = \frac{\sum_{k \in C} TP_k}{\sum_{k \in C} TP_k + FP_k}$$

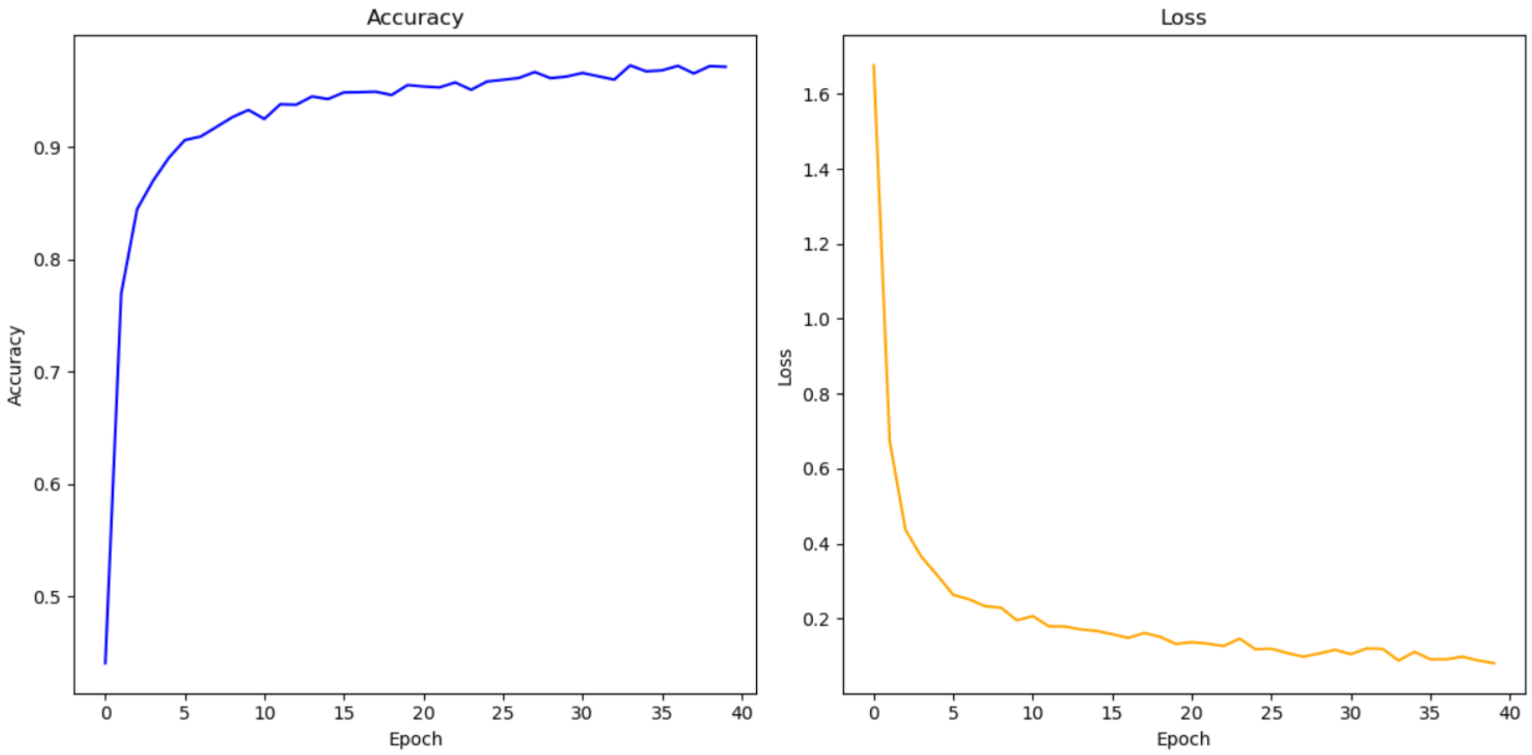
$$Recall_{micro} = \frac{\sum_{k \in C} TP_k}{\sum_{k \in C} TP_k + FN_k}$$

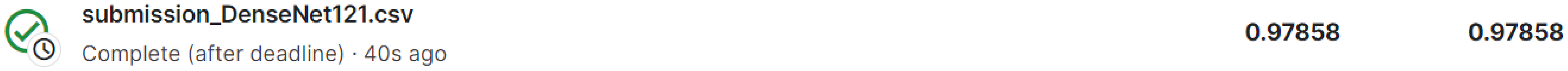
- F1-score is the harmonic mean of precision and recall.

$$MeanFScore = F1_{micro} = \frac{2 Precision_{micro} Recall_{micro}}{Precision_{micro} + Recall_{micro}}$$

Accuracy & Loss

- DenseNet121





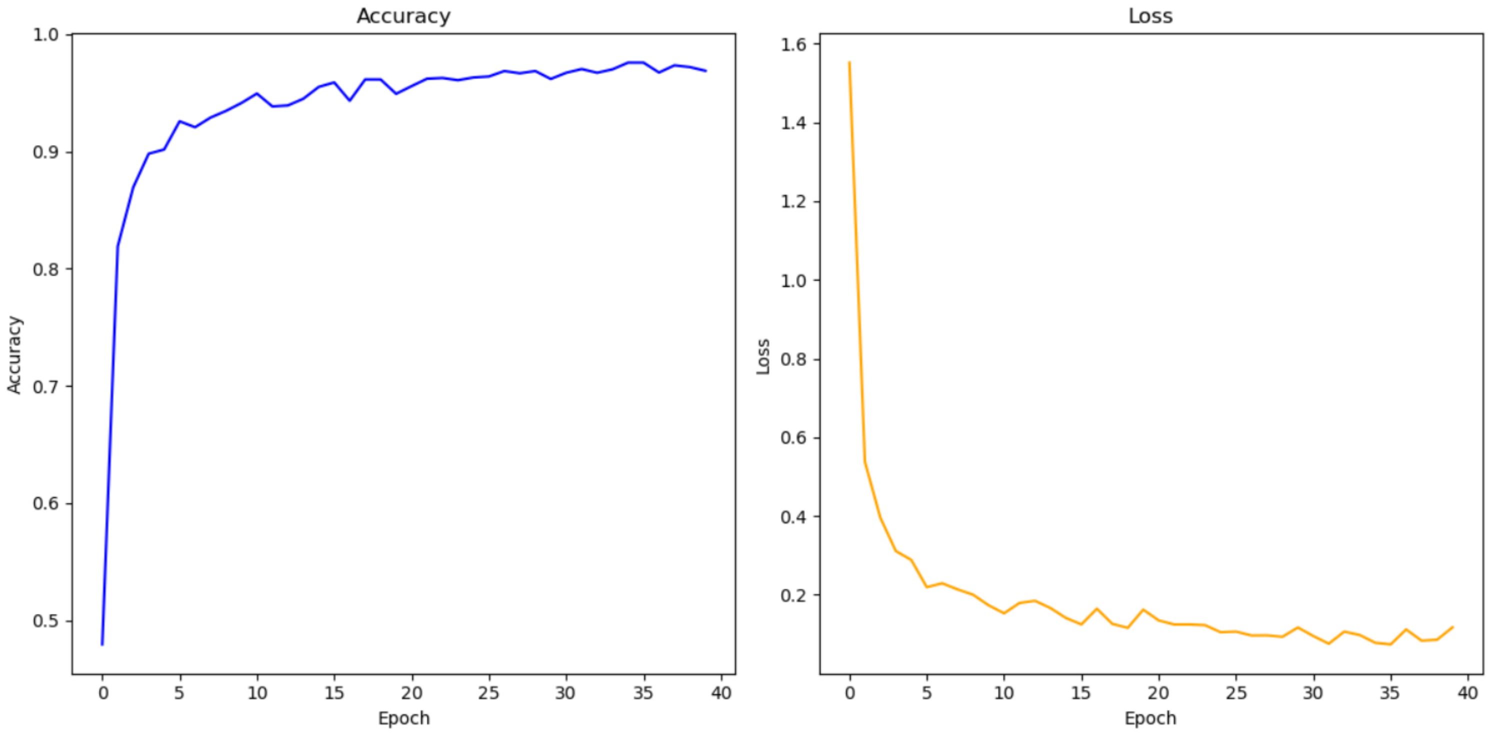
submission_DenseNet121.csv

Complete (after deadline) · 40s ago

0.97858 0.97858

Accuracy & Loss

- DenseNet169



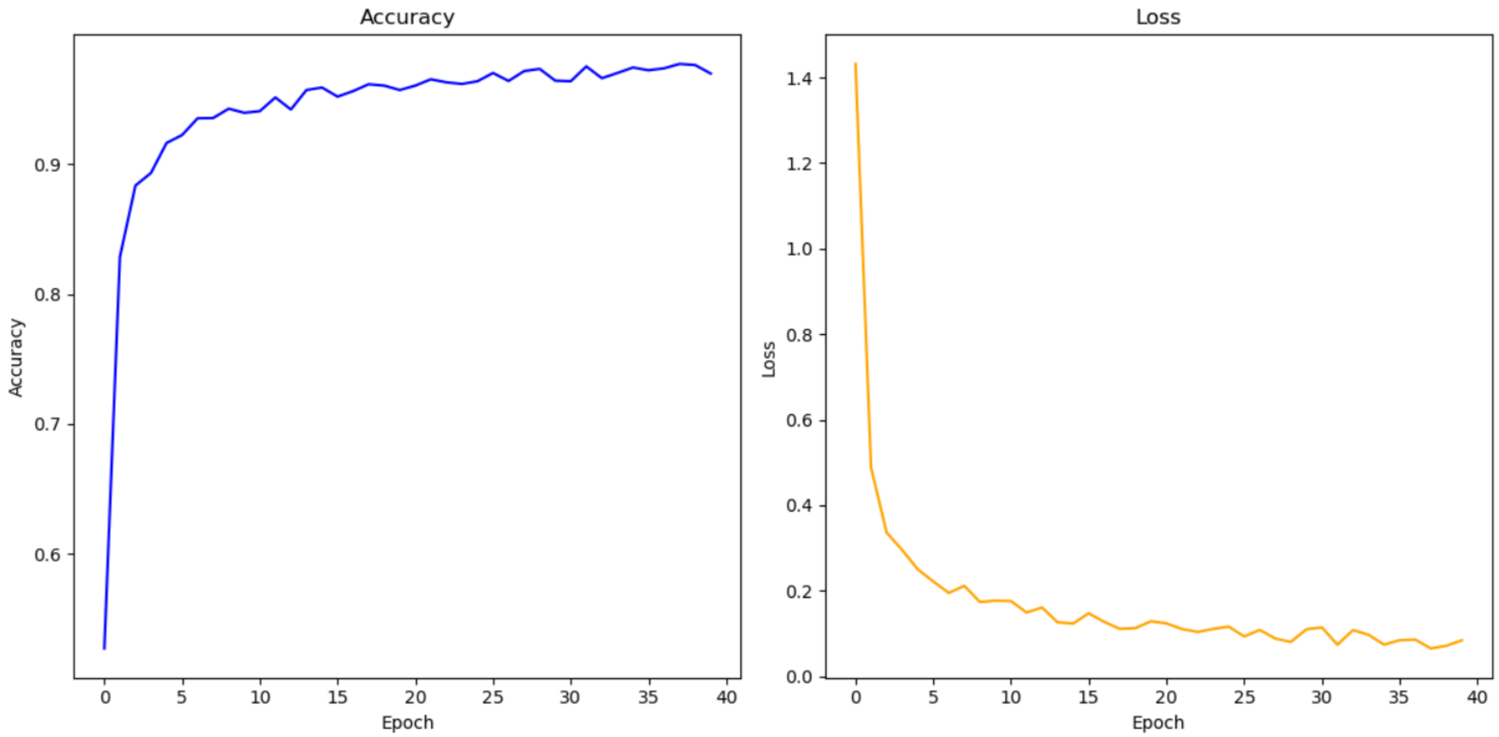
submission_DenseNet169.csv

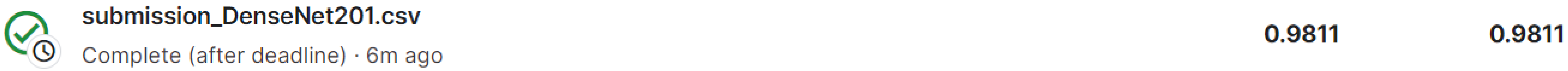
Complete (after deadline) · 1m ago

0.97984 **0.97984**

Accuracy & Loss

- DenseNet201



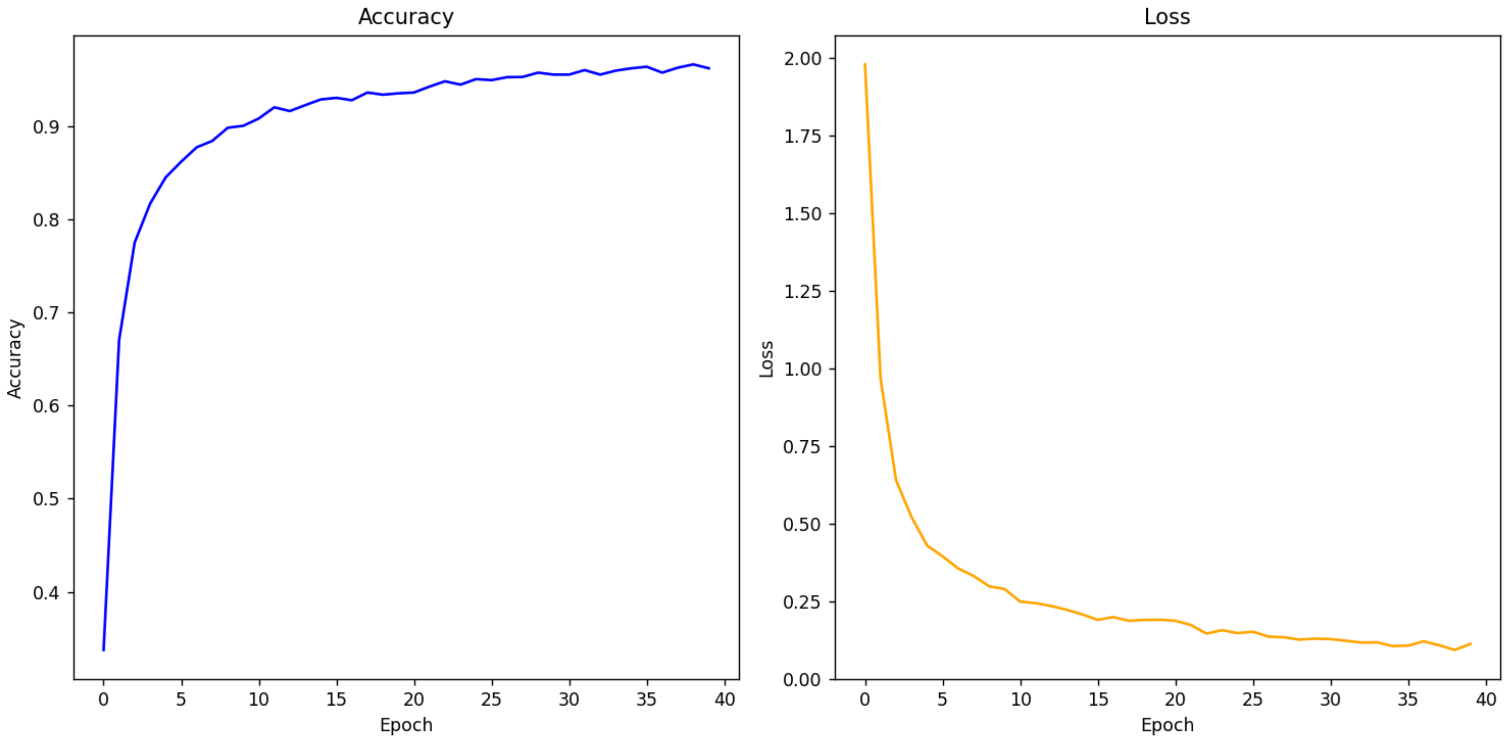

submission_DenseNet201.csv

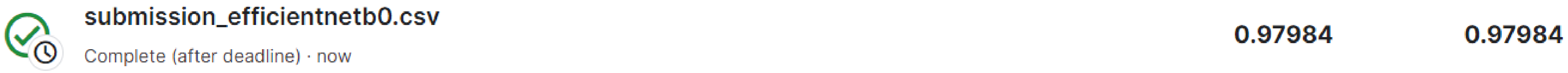
Complete (after deadline) · 6m ago

0.9811 0.9811

Accuracy & Loss

- Efficientnet b0





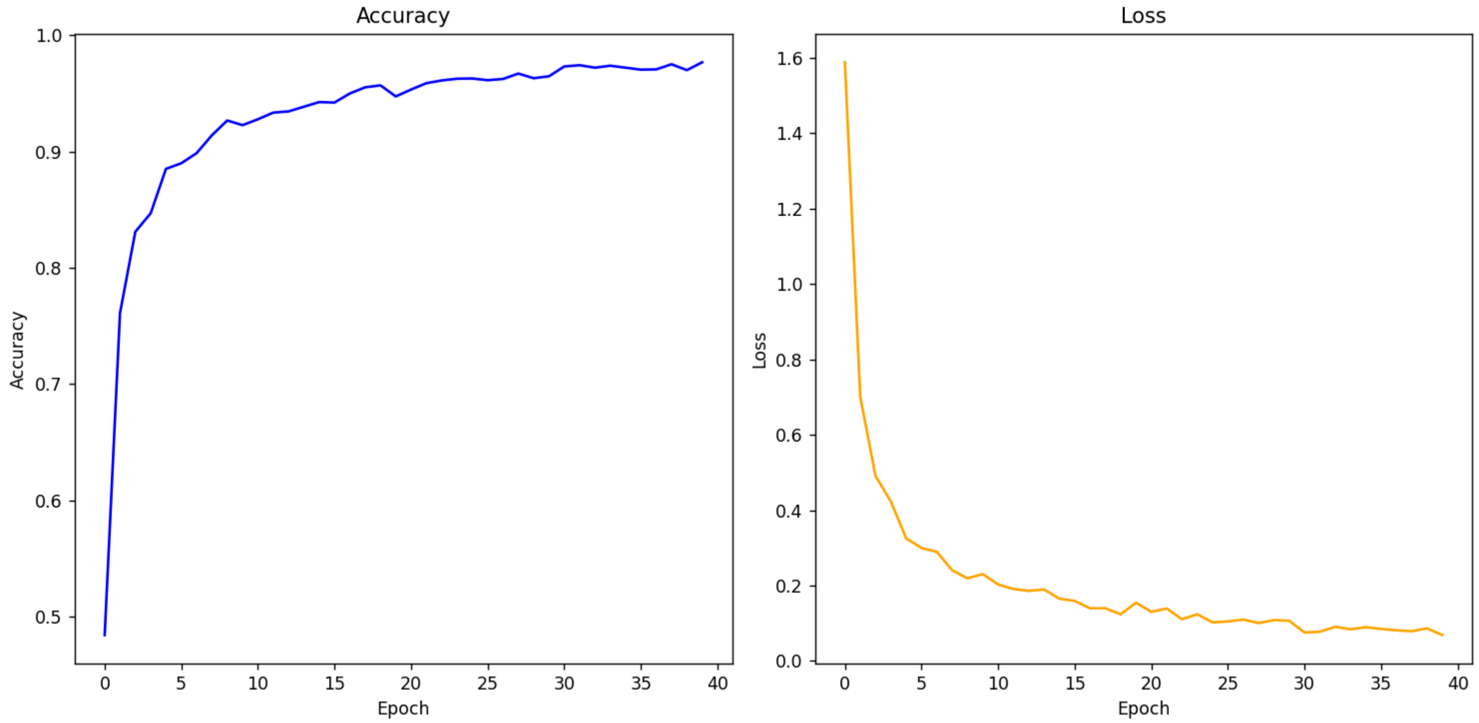
submission_efficientnetb0.csv

Complete (after deadline) · now

0.97984 0.97984

Accuracy & Loss

- Efficientnet b4



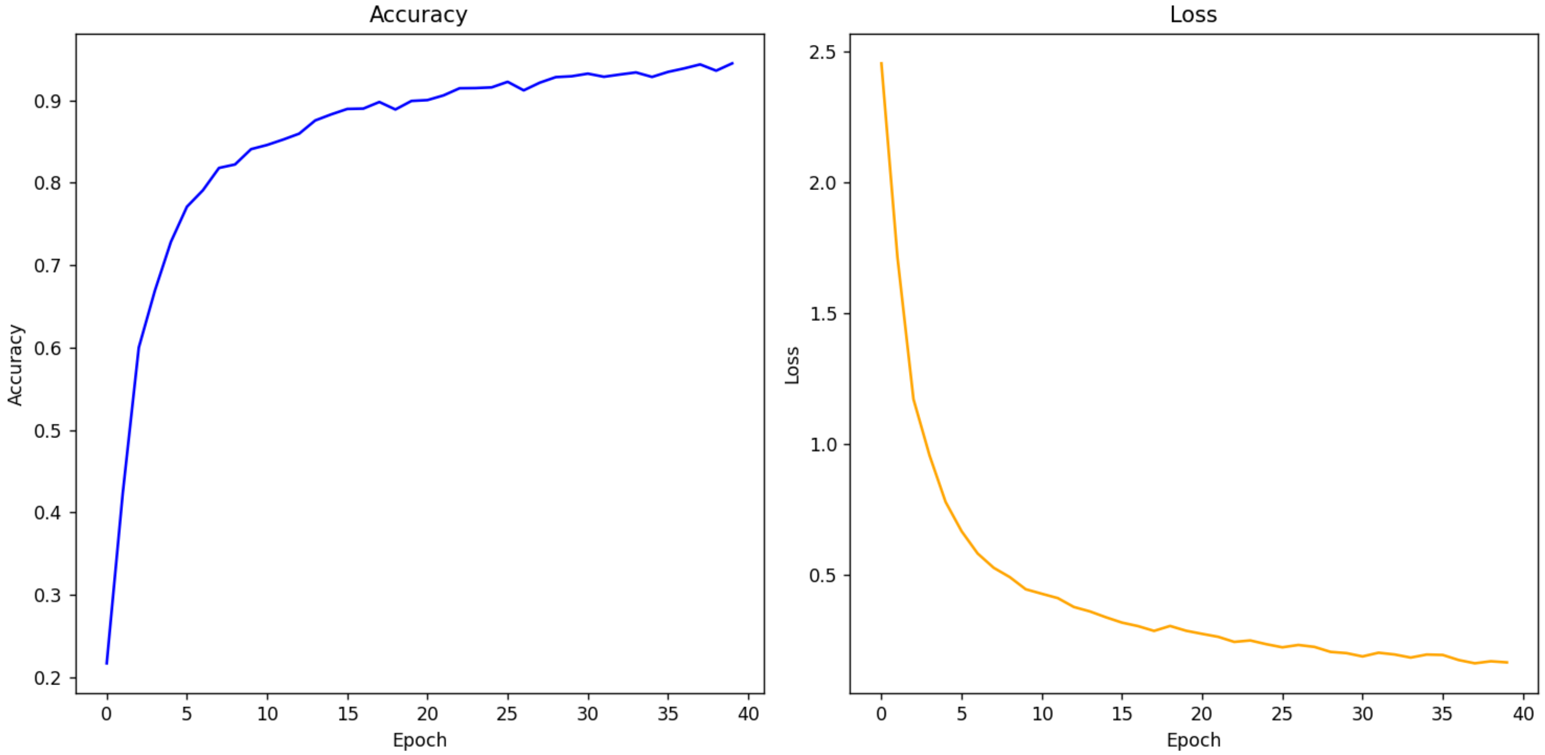
submission_efficientnetb4.csv

Complete (after deadline) · now

0.97858 **0.97858**

Accuracy & Loss

- MobileNetV3 Small



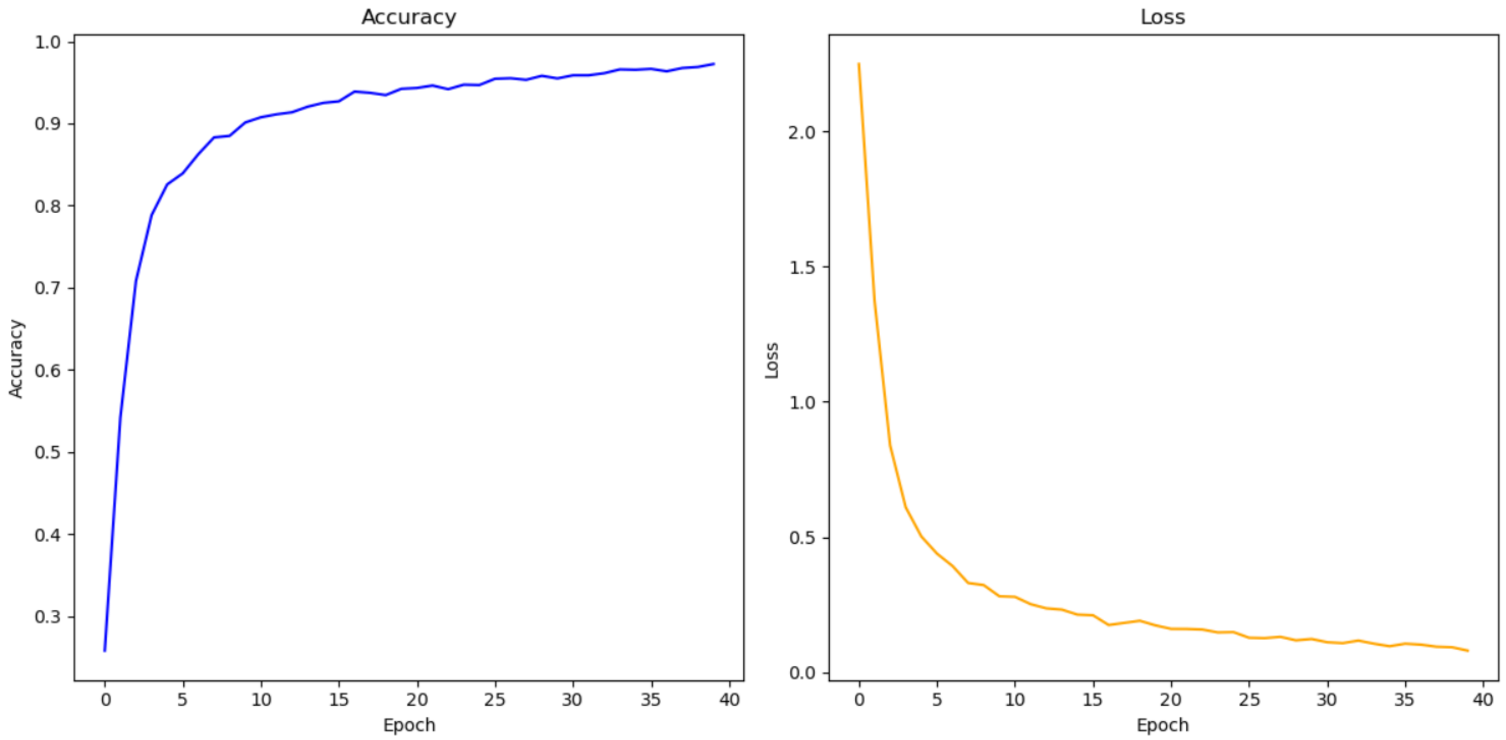
submission_MobileNetV3Small.csv

Complete (after deadline) · now

0.95214 0.95214

Accuracy & Loss

- MobileNetV3 Large



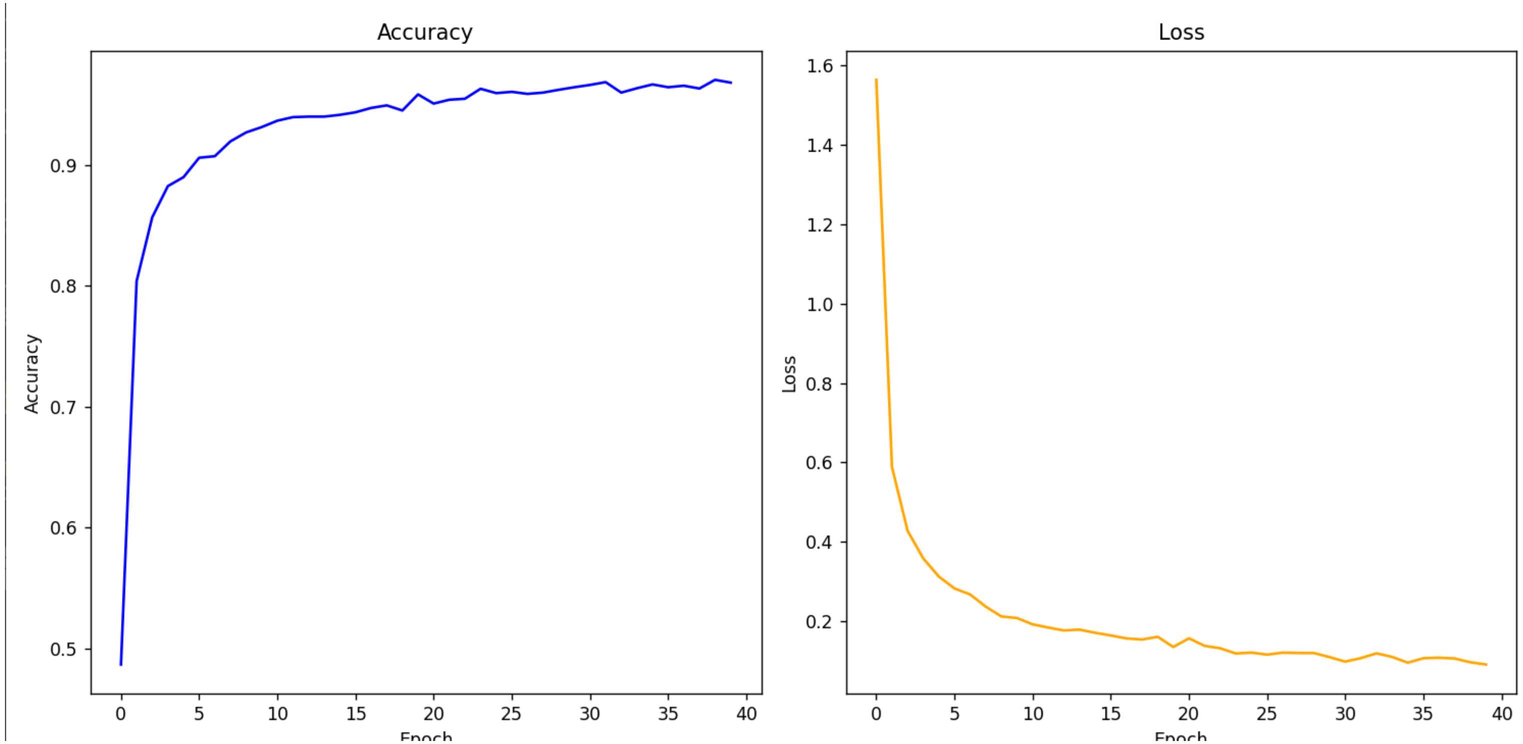
submission_MobileNetV3Large.csv

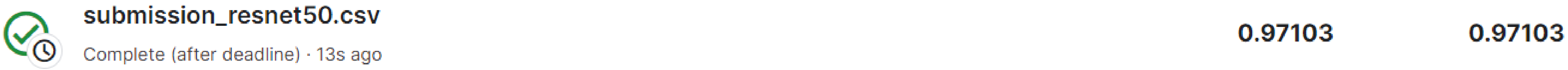
Complete (after deadline) · 6m ago

0.97607 0.97607

Accuracy & Loss

- Restnet50



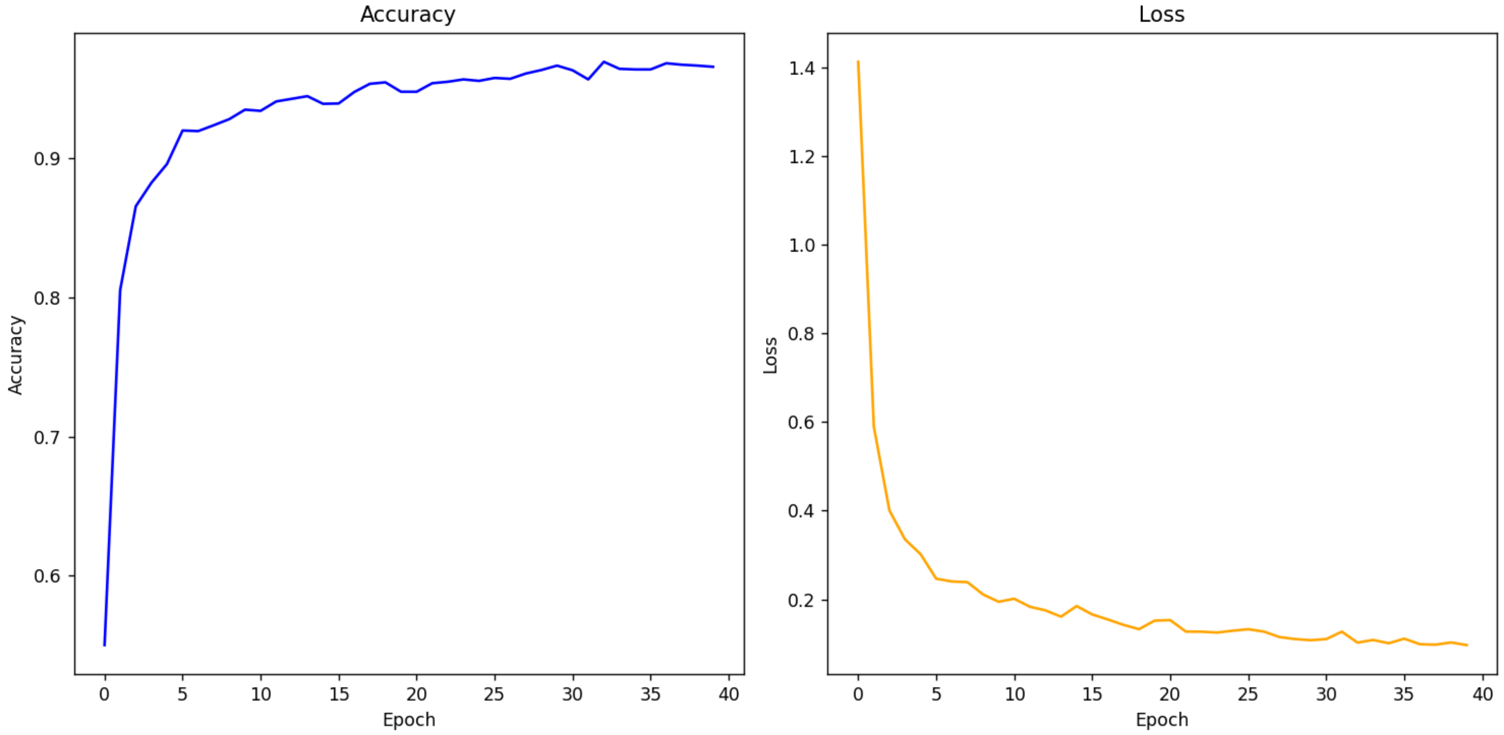

submission_resnet50.csv

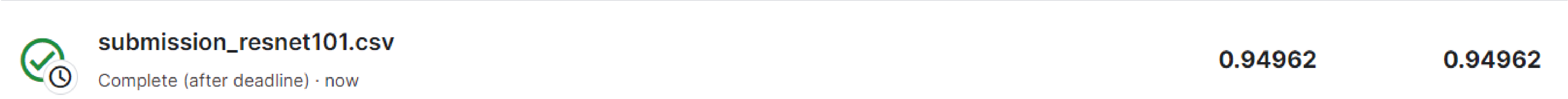
Complete (after deadline) · 13s ago

0.97103 **0.97103**

Accuracy & Loss

- Restnet101





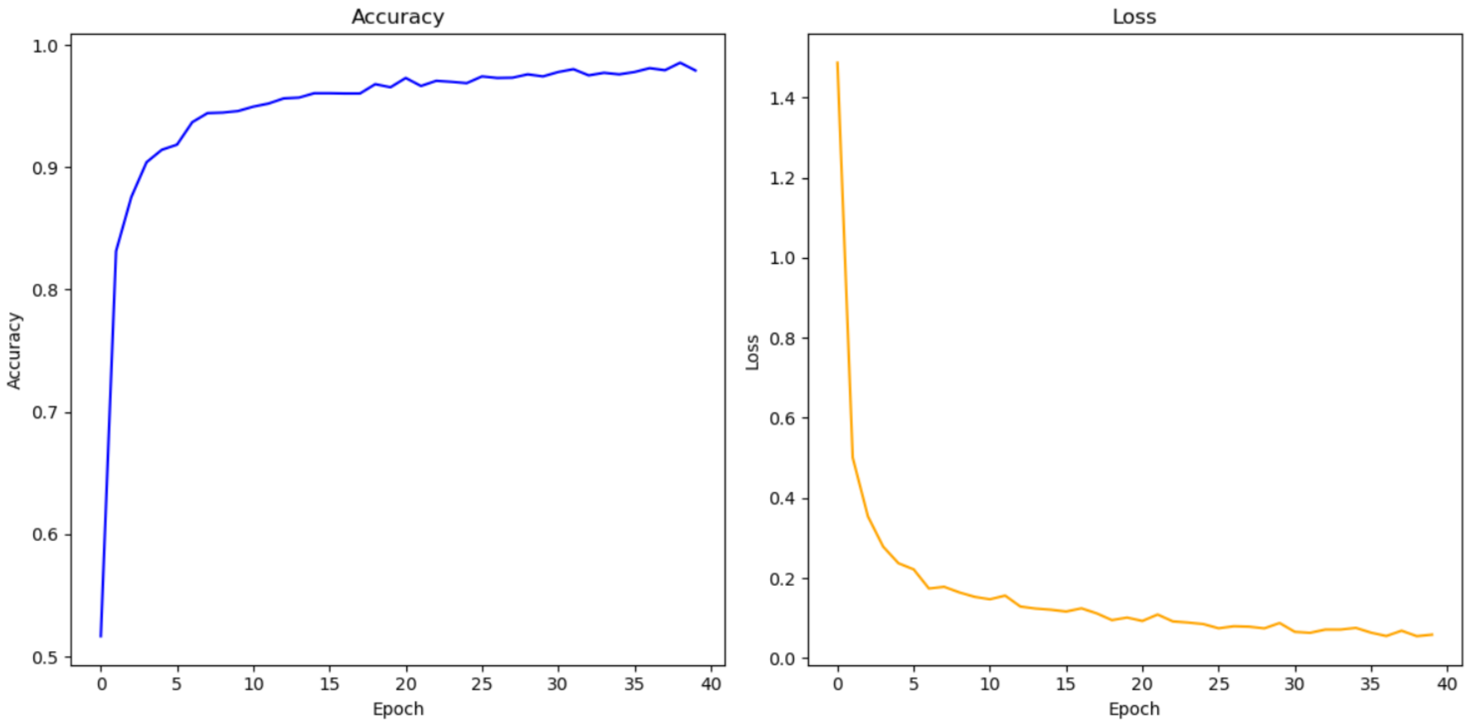
submission_resnet101.csv

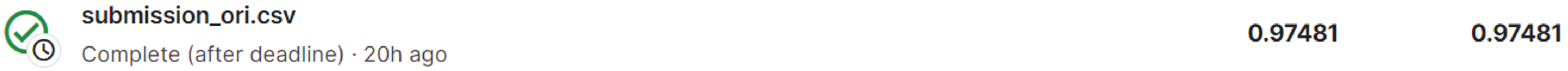
Complete (after deadline) · now

0.94962 **0.94962**

Accuracy & Loss

- InceptionResNetV2





submission_ori.csv

Complete (after deadline) · 20h ago

0.97481 0.97481

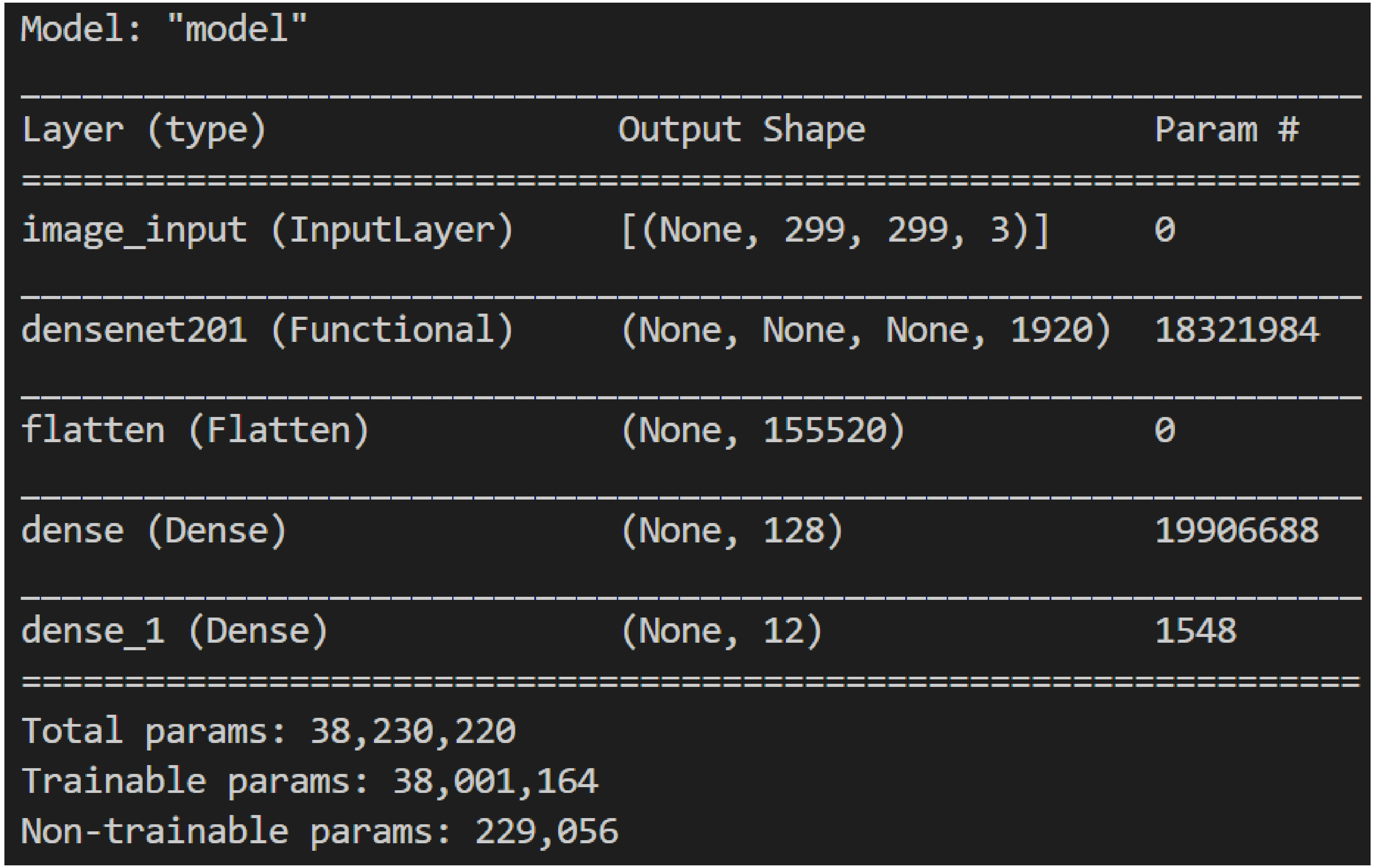
Result - Model Comparison

	DenseNet121	DenseNet169	DenseNet201	Efficientnet b0	Efficientnet b4
F1-score	0.9785	0.9798	**0.9811**	0.9798	0.9785

	MobileNetV3 Small	MobileNetV3 Large	Restnet50	Restnet101	InceptionResNetV2
F1-score	0.9521	0.9760	0.9710	0.9496	0.9748

- After comparing the performance of the listed models, it is found that DenseNet201 achieves the highest F1 score.
- This may be due to its larger model capacity, dense connectivity, specific architectural features, and adaptability to the characteristics of the evaluated dataset.

DenseNet201 Architecture



```
Model: "model"

_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
image_input (InputLayer)     [(None, 299, 299, 3)]     0

_________________________________________________________________
densenet201 (Functional)     (None, None, None, 1920)  18321984

_________________________________________________________________
flatten (Flatten)            (None, 155520)            0

_________________________________________________________________
dense (Dense)                (None, 128)               19906688

_________________________________________________________________
dense_1 (Dense)              (None, 12)                1548
=================================================================
Total params: 38,230,220
Trainable params: 38,001,164
Non-trainable params: 229,056
```