

2025

ColoVision

AI with a *gut* feeling.

Team

CAU Kiel MSc students

Mikhail Melnichenko
Daria Fedorova

Models

Image classification

Basic CNN
DenseNet121



Sessile serrated adenoma (SSA) and hyperplastic polyps (HP)

PREVALENCE

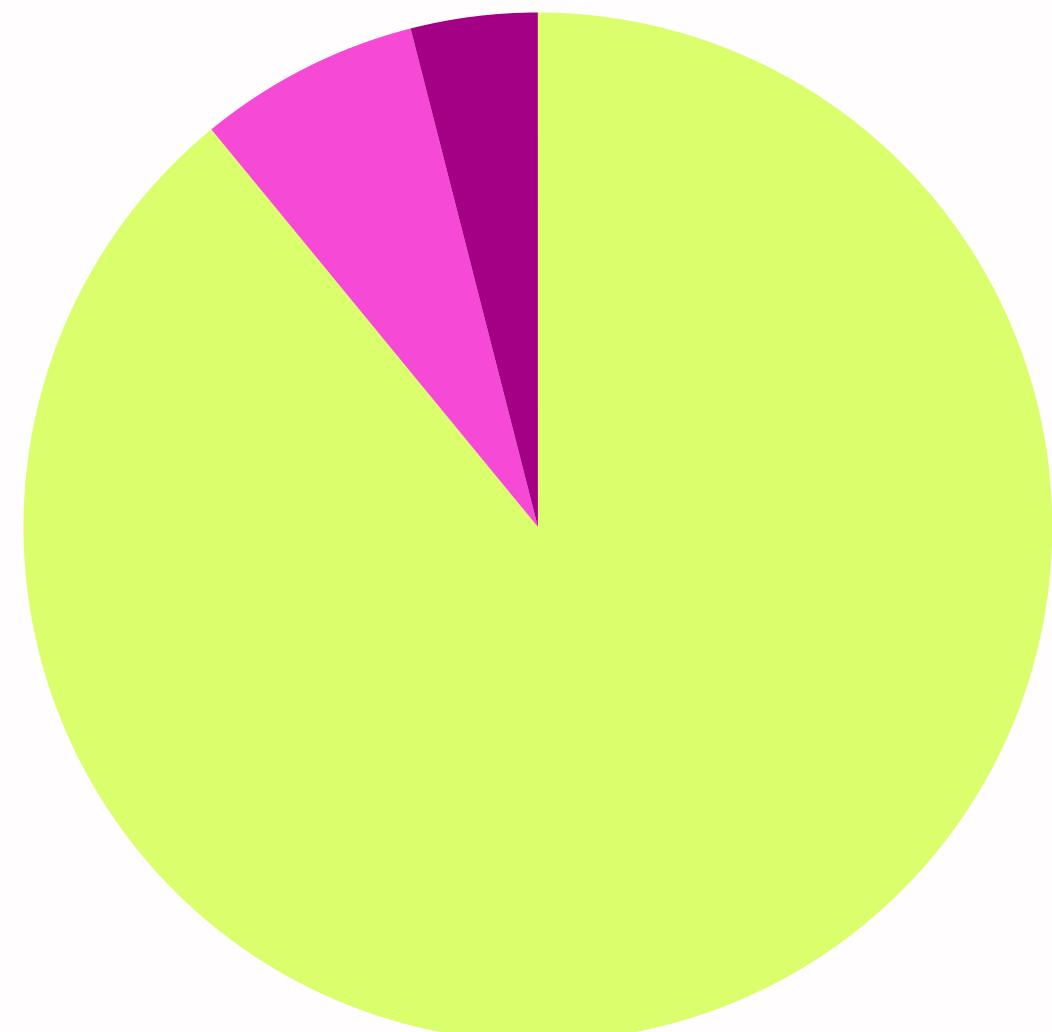
SSA, now often referred to as sessile serrated lesions (SSLs) are **precursors to 15–30% of colorectal cancers.**

Overall prevalence of SSAs is 8.2%.¹

CHALLENGE

HPs are typically benign.
Distinguish HP from SSA → “a challenging problem with considerable inter-pathologist variability.”²

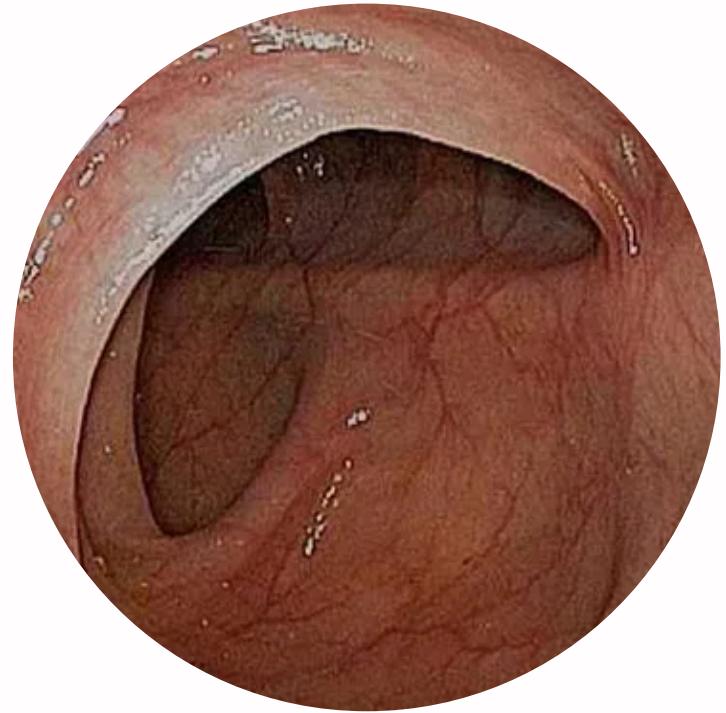
- Hyperplastic polyps (HP)
- Sessile Serrated Adenoma
- Traditional Serrated Adenoma



1- IJspéert JE et al, 2016

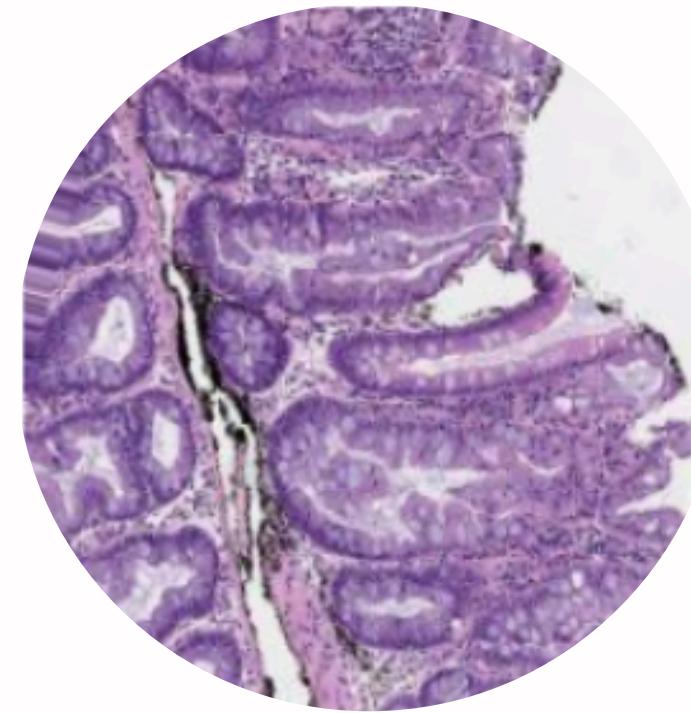
2 - Wei J. et al, 2021

Cancer diagnosis and AI



ENDOSCOPY

<https://mantasmd.com/endoscopy-colon-splenic-flexure-normal/>



HISTOLOGY

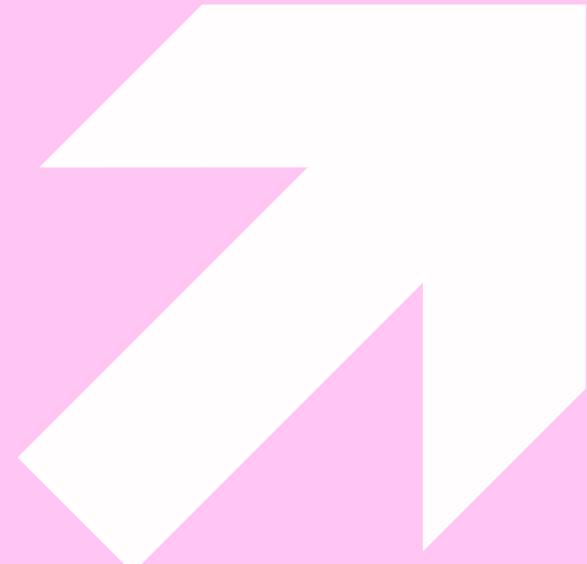
MHIST dataset



**MOLECULAR
DIAGNOSTICS**

<https://www.istockphoto.com/de/search/2/image?mediatype=illustration&phrase=dna+sequence+icon>

Literature



ACCURATE AND REPRODUCIBLE INVASIVE BREAST CANCER DETECTION IN WHOLE-SLIDE IMAGES: A DEEP LEARNING APPROACH FOR QUANTIFYING TUMOR EXTENT

2017 BY CRUZ-ROA, A. ET AL

Demonstrates CNNs' effectiveness in cancer region identification on Whole slide images (wsIs)

DEEP LEARNING IN CANCER GENOMICS AND HISTOPATHOLOGY

Informs the selection of appropriate architectures and supervision strategies for colon polyp classification

2024 BY UNGER, M. ET AL

A PETRI DISH FOR HISTOPATHOLOGY IMAGE ANALYSIS (MHIST DATASET)

2021 BY WEI, J. ET AL

Provides a directly relevant dataset for developing and testing computer vision models to distinguish benign from malignant colon polyps. ...and we want to **BEAT THE MODEL FROM THIS PAPER :)**

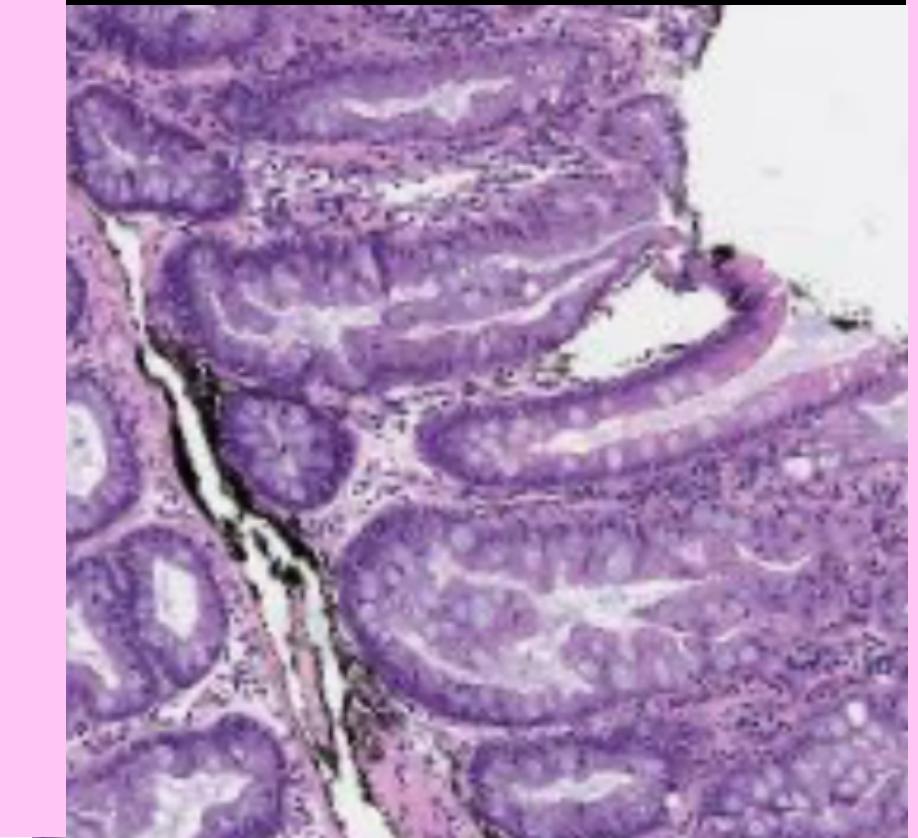
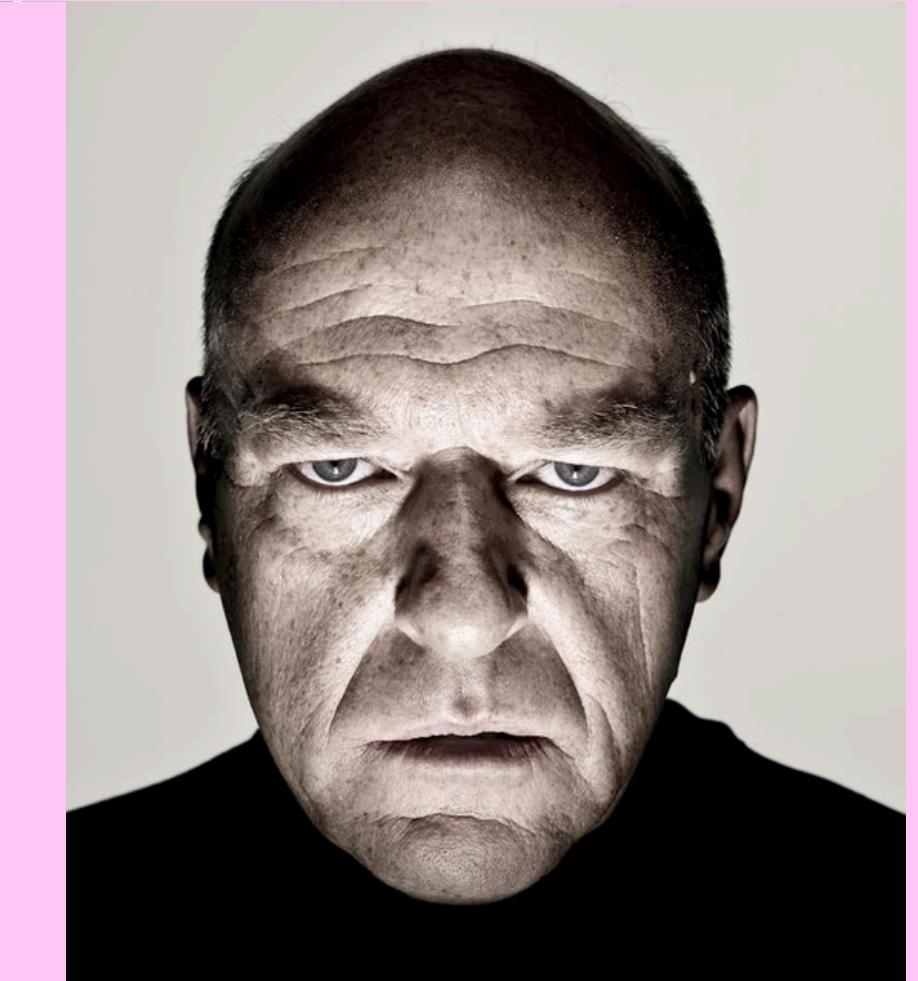
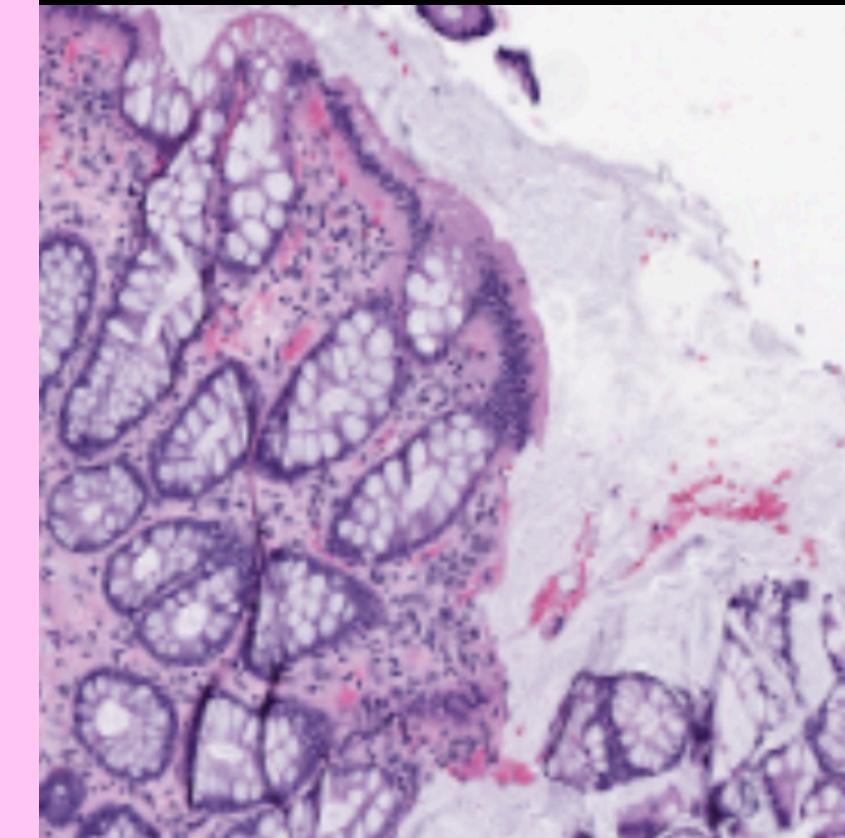
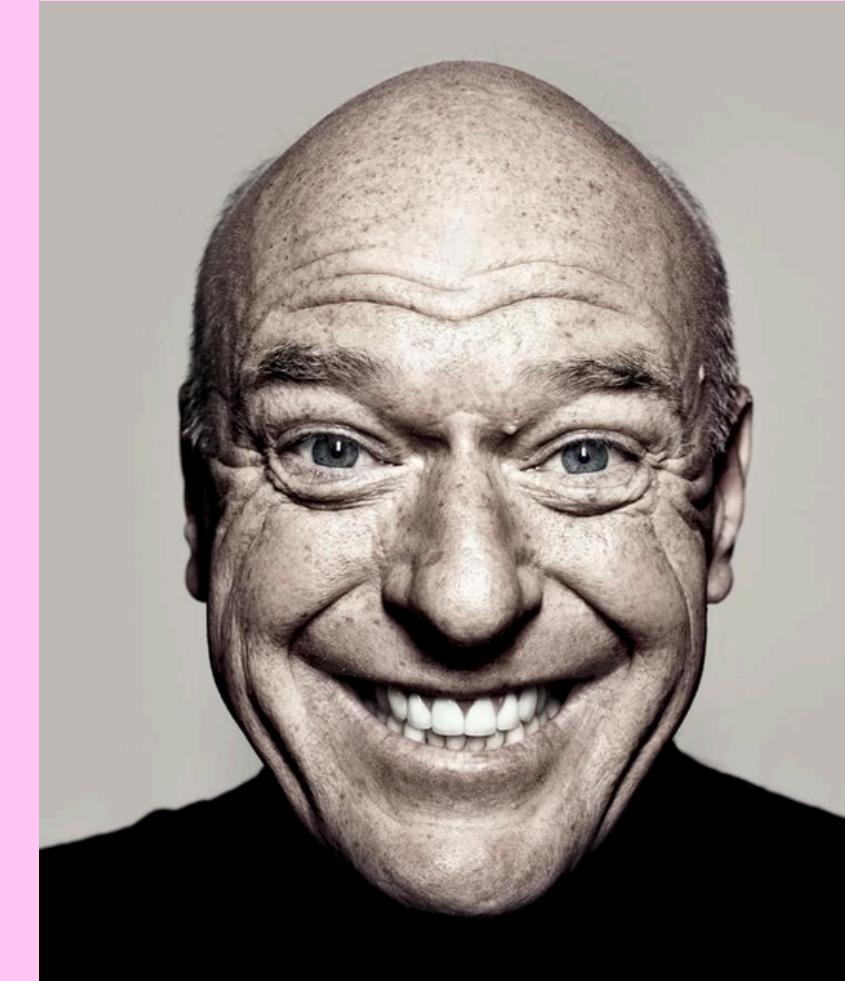
AUC = 92.7 ± 0.4

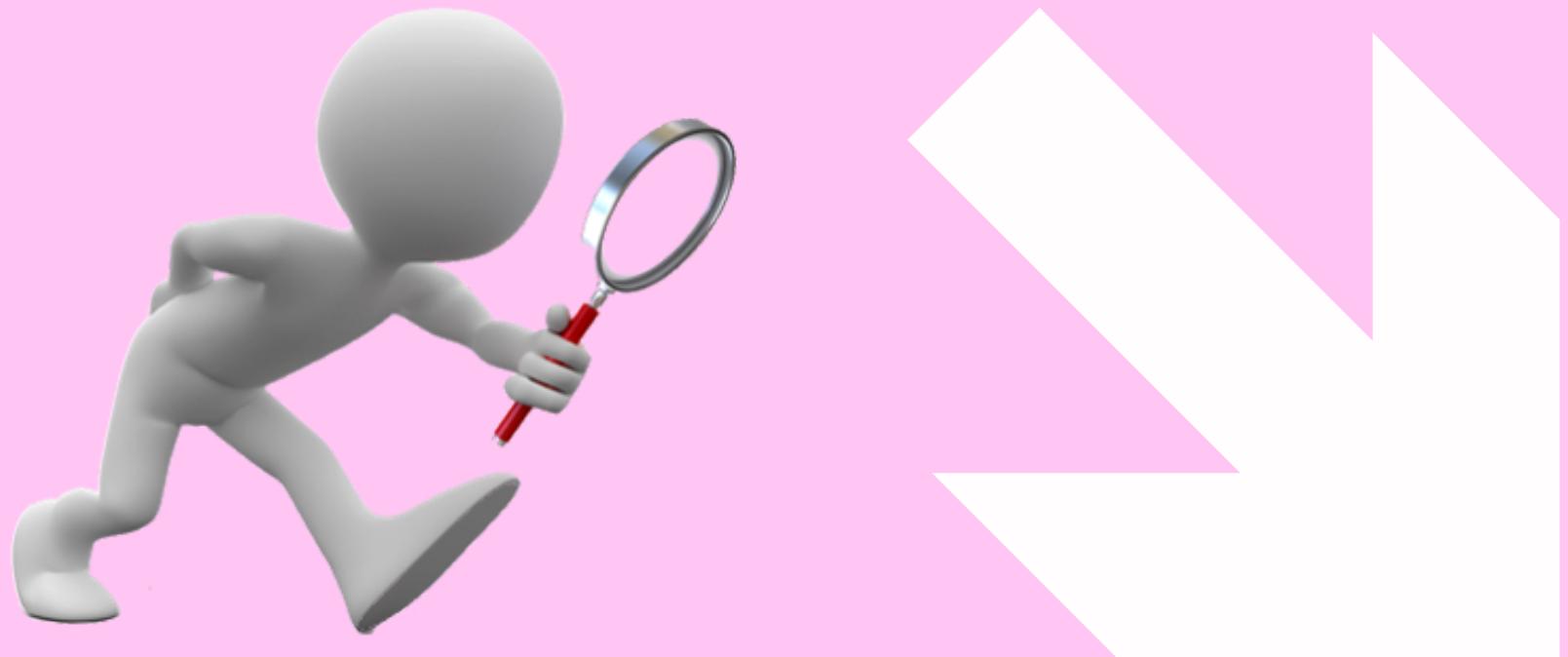
Good or bad?

MHIST dataset:

3,152 RGB images 224×224 px
from 328 clinical slides

- benign or HP, polyp
- malignant or SSA, adenoma





Feature engineering

- 7 pathologists voting
 - 3/7 (43%) and 4/7 (57%) excluded
- Augmentation
 - rescaling (0-255)
 - zoom (up to 10%)
 - rotation (up to 15°)
 - horizontal flip

GENERAL SCHEME

1. RANDOM SEARCH

CNN:

- number of filters/neurons in each layer (conv/dense)

Transfer learning ([DenseNet121](#)):

- number of neurons in the dense layer
- dropout rate
- alpha & gamma in focal loss function

2. RANDOM SEARCH FINE TUNING

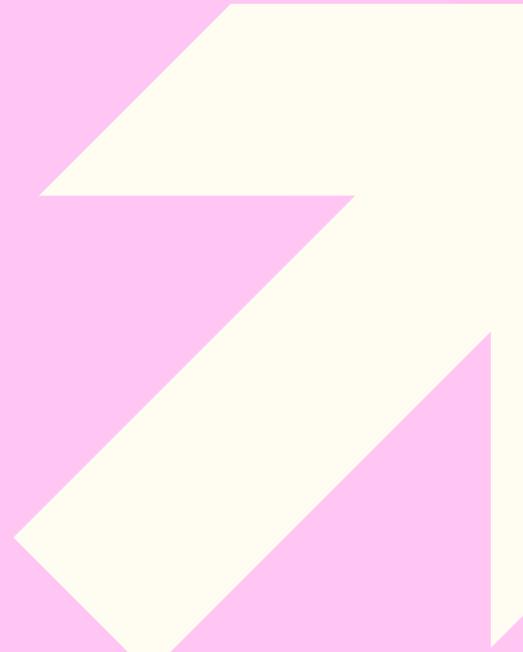
- number of frozen layers

3. TRAIN MODEL WITH THE BEST PARAMETERS

- CALLBACK function: automatic learning rate correction using ReduceLROnPlateau



```
def focal_loss(gamma=2.0, alpha=0.75):
    def loss(y_true, y_pred):
        y_pred = K.clip(y_pred, K.epsilon(), 1. - K.epsilon())
        alpha_t = tf.where(tf.equal(y_true, 1), alpha, 1 - alpha)
        pt = tf.where(tf.equal(y_true, 1), y_pred, 1 - y_pred)
        return -alpha_t * K.pow(1. - pt, gamma) * K.log(pt)
    return loss
```



Baseline:
CNN →
focal loss



```
def build_model(hp):
    model = models.Sequential([
        layers.Input(shape=(224, 224, 3)),

        layers.Conv2D(hp.Int('conv1_filters', 32, 128, step=32), (3, 3), padding='same'),
        layers.BatchNormalization(),
        layers.ReLU(),
        layers.MaxPooling2D(),
        layers.Dropout(hp.Float('dropout1', 0.2, 0.5, step=0.1)),

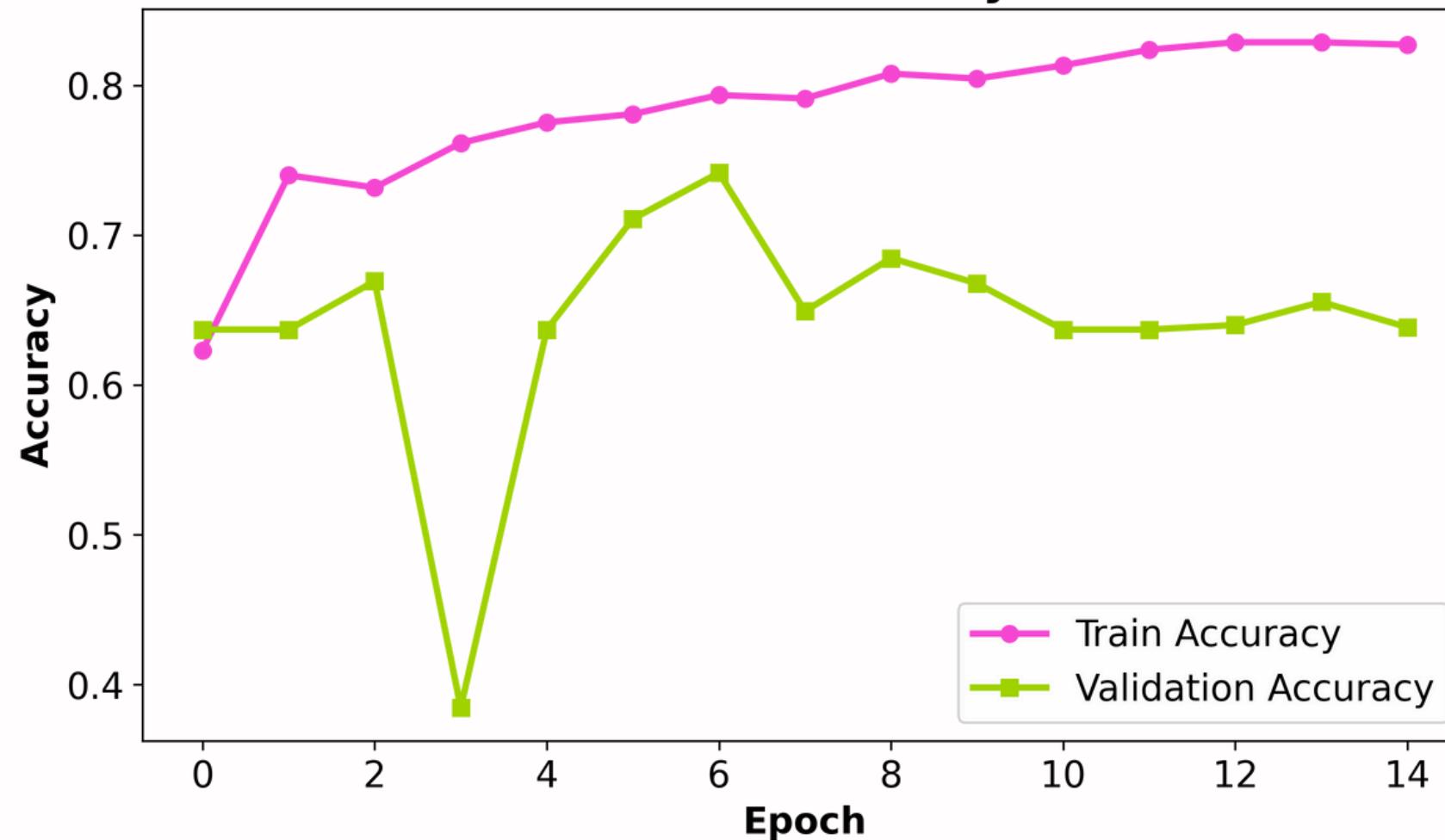
        layers.Conv2D(hp.Int('conv2_filters', 64, 256, step=64), (3, 3), padding='same'),
        layers.BatchNormalization(),
        layers.ReLU(),
        layers.MaxPooling2D(),
        layers.Dropout(hp.Float('dropout2', 0.2, 0.5, step=0.1)),

        layers.Conv2D(hp.Int('conv3_filters', 128, 512, step=128), (3, 3), padding='same'),
        layers.BatchNormalization(),
        layers.ReLU(),
        layers.MaxPooling2D(),
        layers.Dropout(hp.Float('dropout3', 0.2, 0.5, step=0.1)),

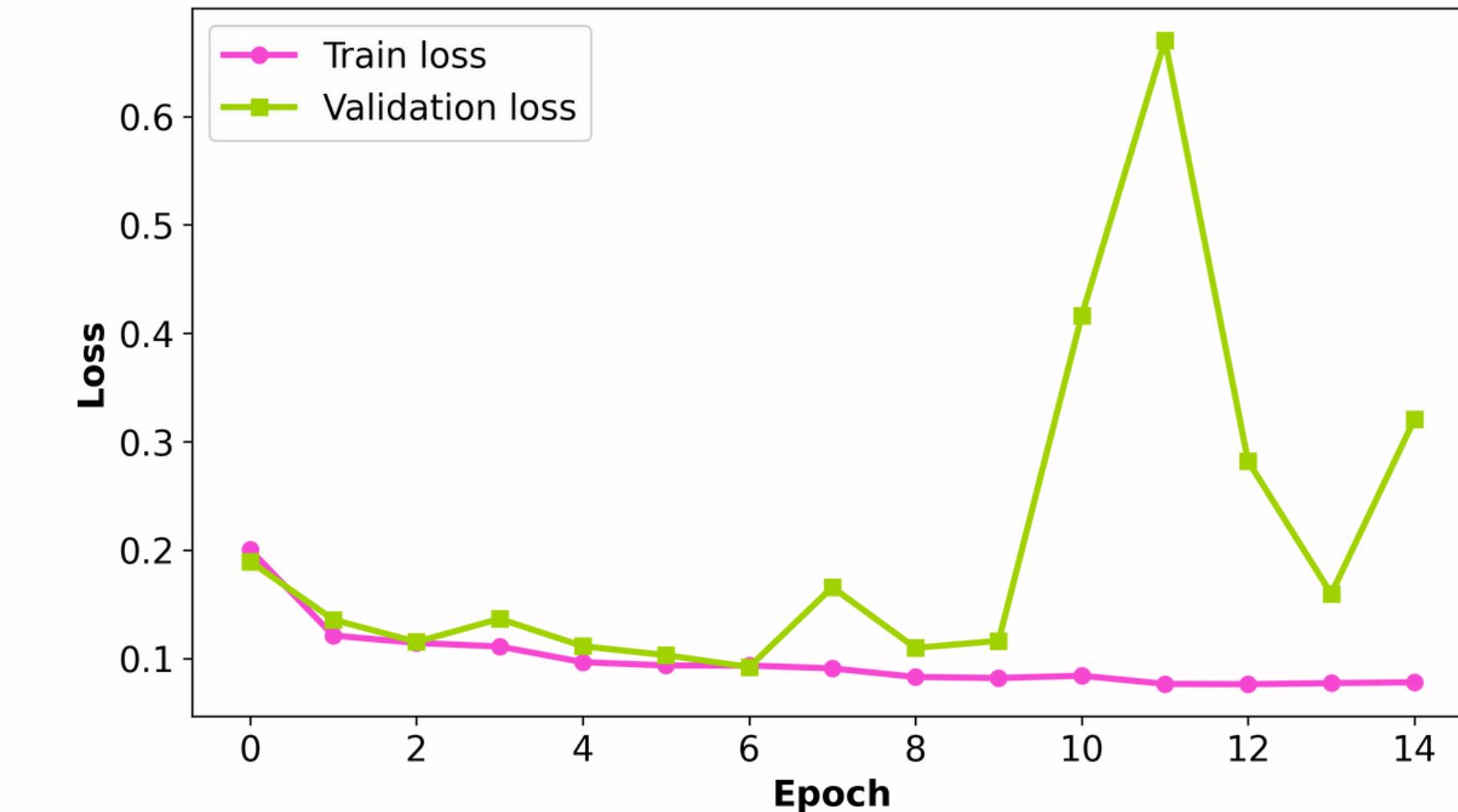
        layers.Flatten(),
        layers.Dense(hp.Int('dense_units', 128, 512, step=64)),
        layers.BatchNormalization(),
        layers.ReLU(),
        layers.Dropout(hp.Float('dropout4', 0.2, 0.5, step=0.1)),

        layers.Dense(1, activation='sigmoid')
    ])
```

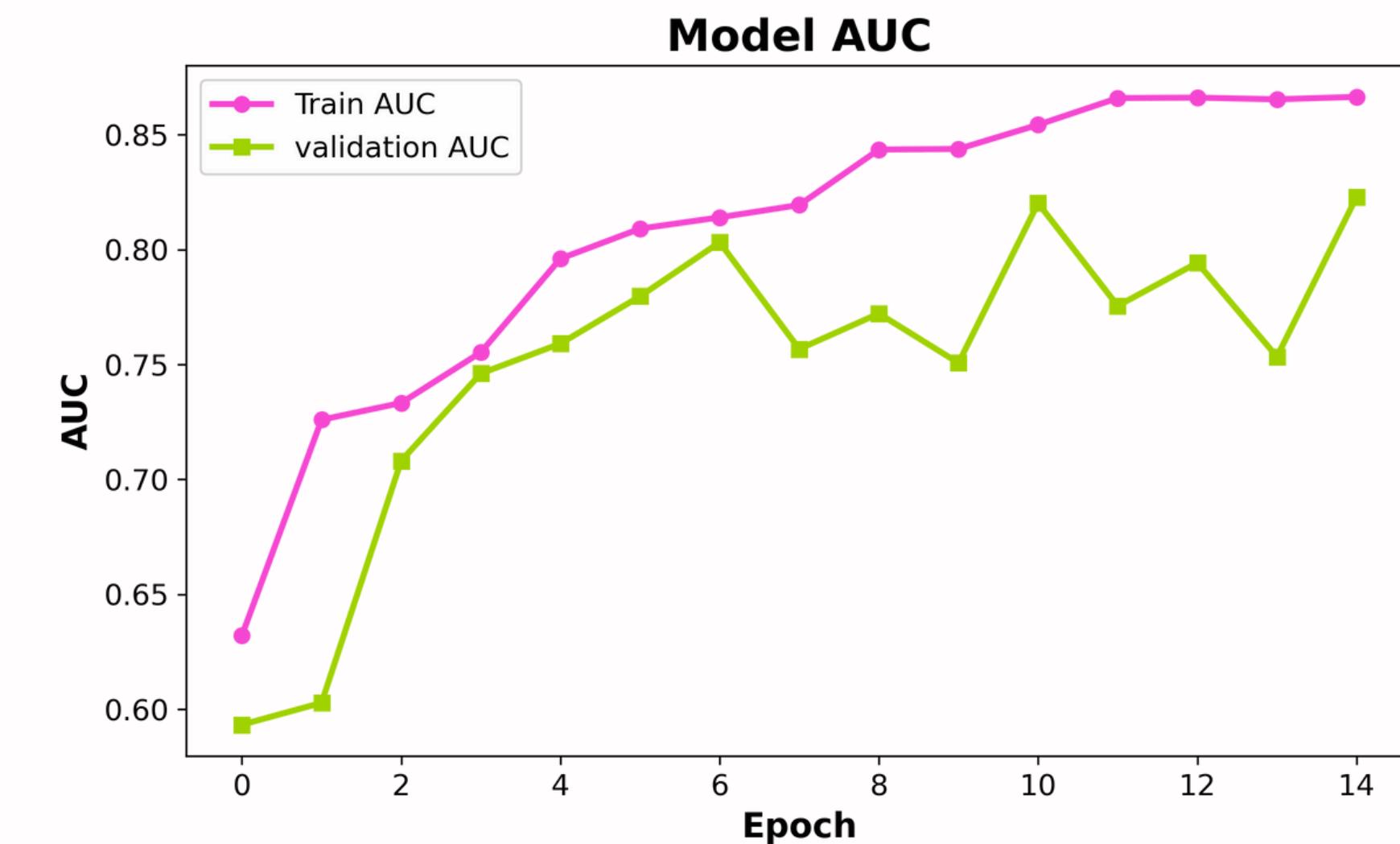
Model Accuracy



Model loss



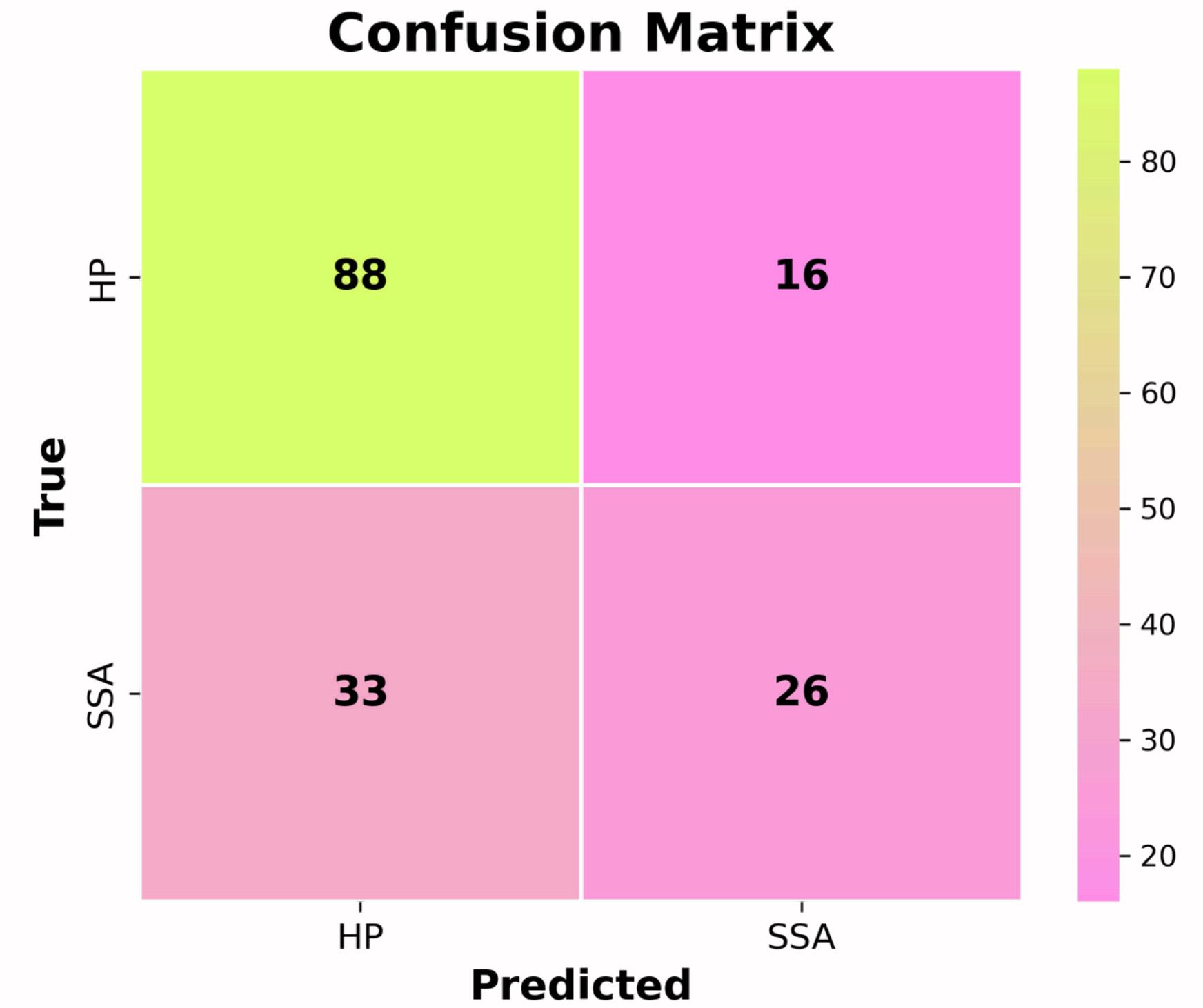
**Baseline:
CNN**



	Training	Validation
Accuracy	0.8216	0.6385
Loss	0.0819	0.3205
AUC	0.8574	0.8229



**Baseline:
CNN**



Transfer learning

DenseNet121

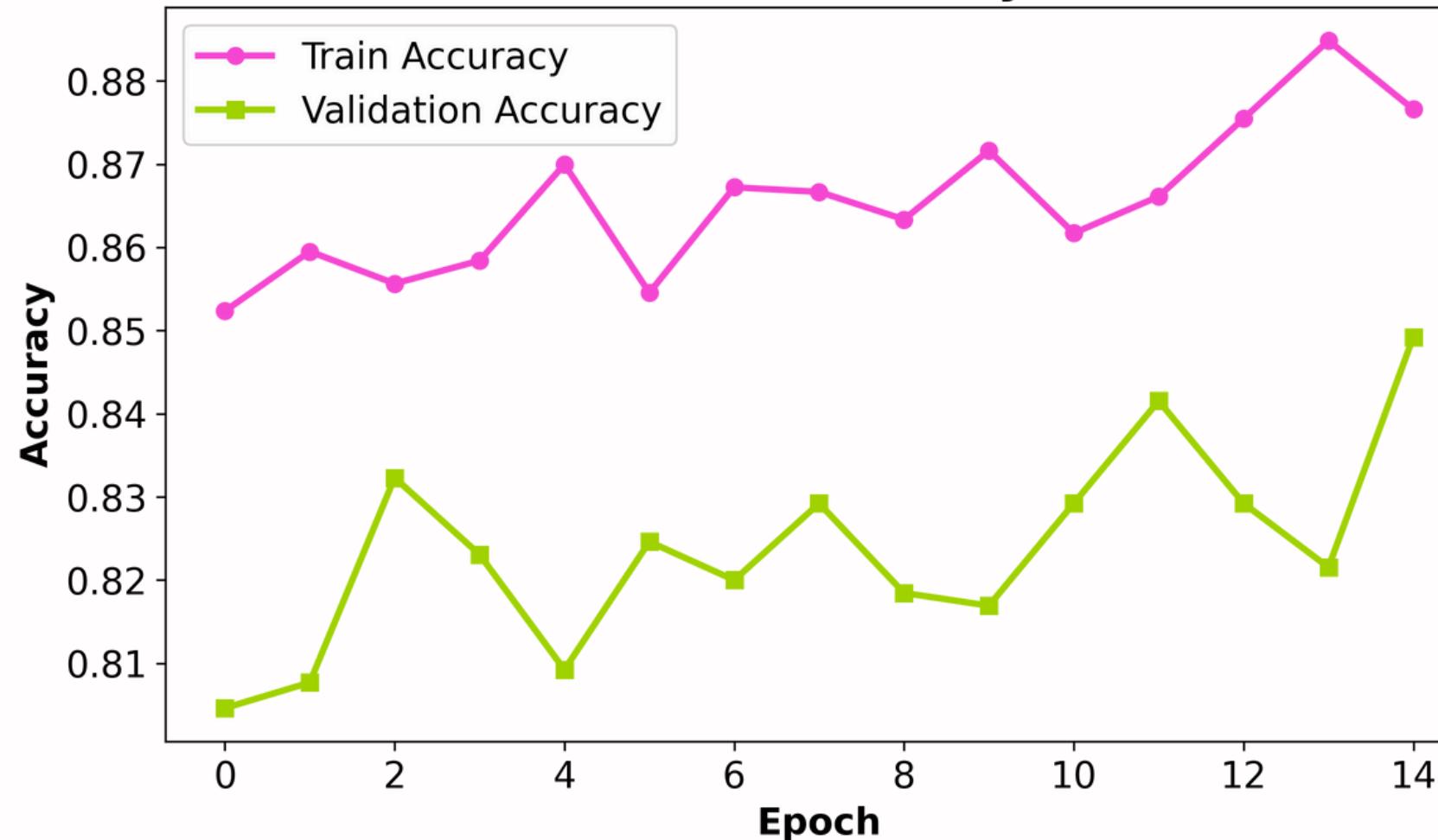
```
def build_feature_extraction_model(hp):
    base_model = DenseNet121(weights='imagenet', include_top=False,
input_shape=(224, 224, 3), name='densenet121_base')
    base_model.trainable = False

    x = base_model.output
    x = GlobalAveragePooling2D()(x)
    x = Dense(hp.Int('dense_units', 64, 256, step=64))(x)
    x = BatchNormalization()(x)
    x = tf.keras.layers.ReLU()(x)
    x = Dropout(hp.Float('dropout', 0.2, 0.5, step=0.1))(x)
    output = Dense(1, activation='sigmoid')(x)

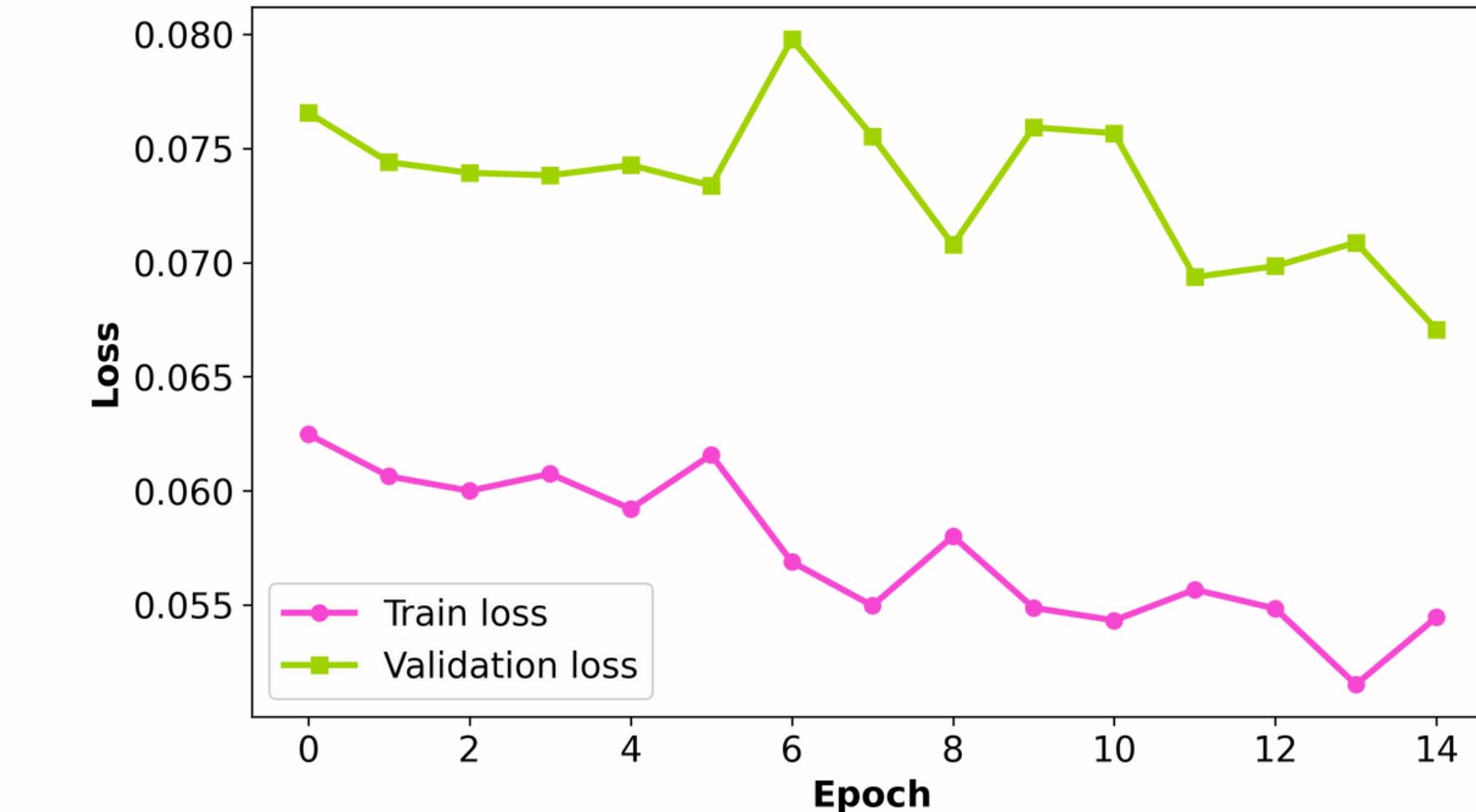
    model = Model(inputs=base_model.input, outputs=output)

    model.compile(
        optimizer=Adam(learning_rate=hp.Choice('lr', [1e-5])),
        loss=focal_loss(gamma=hp.Choice('gamma', [1.5, 2.0])),
        alpha=hp.Choice('alpha', [0.5, 0.75])),
        metrics=['accuracy', tf.keras.metrics.AUC(name='auc')])
    )
    return model
```

Model Accuracy

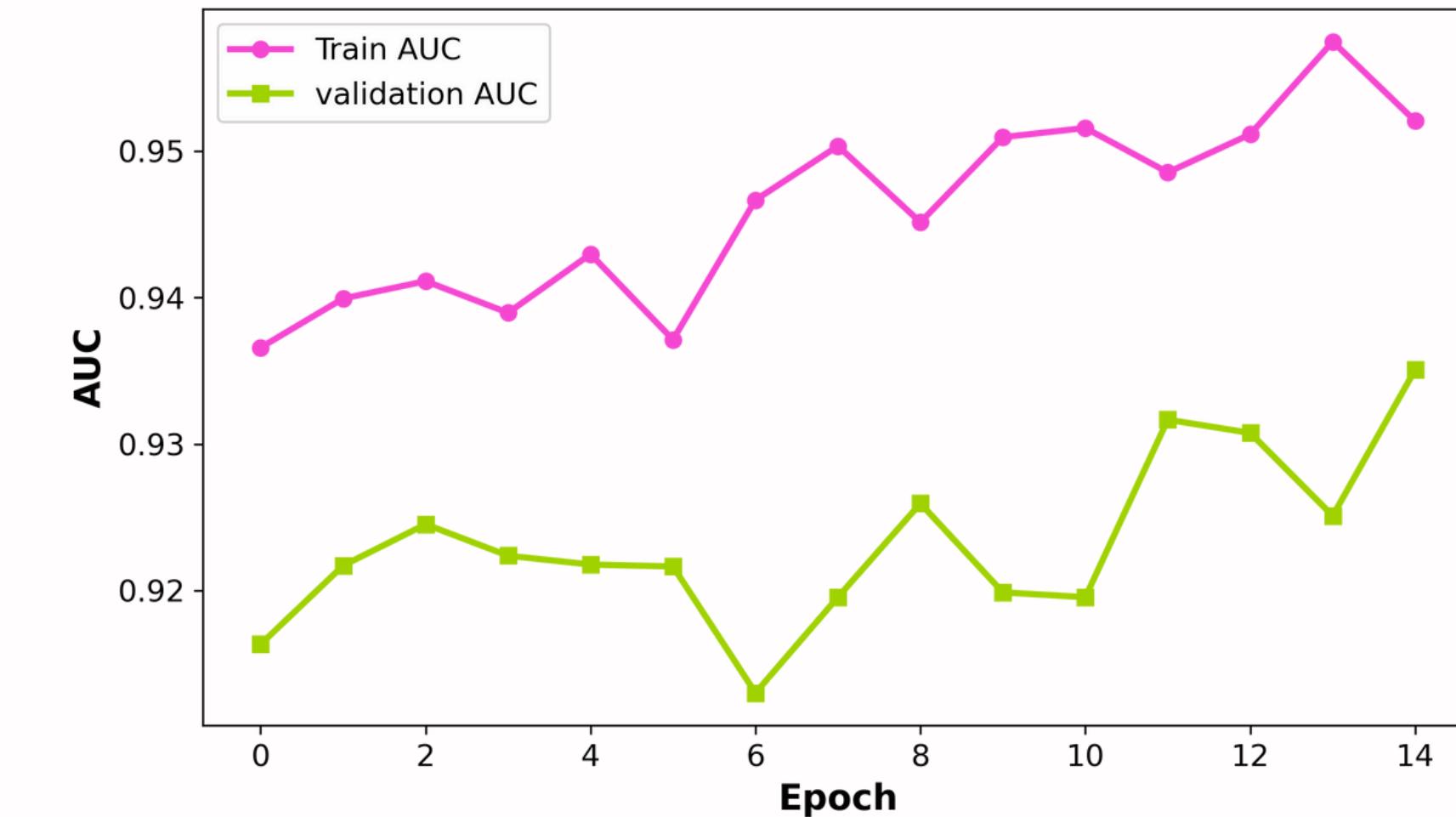


Model loss



DenseNet121:
after search
and fine-tuning

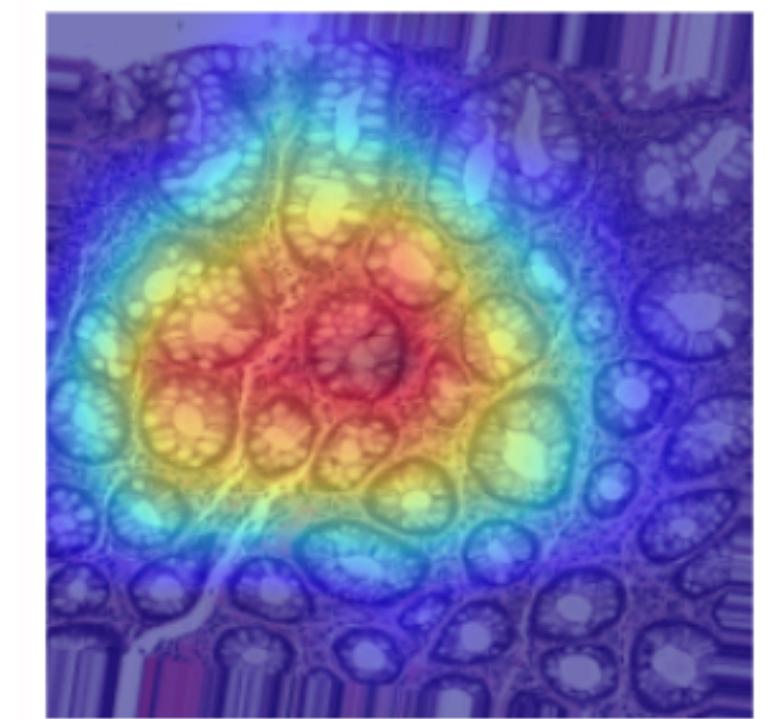
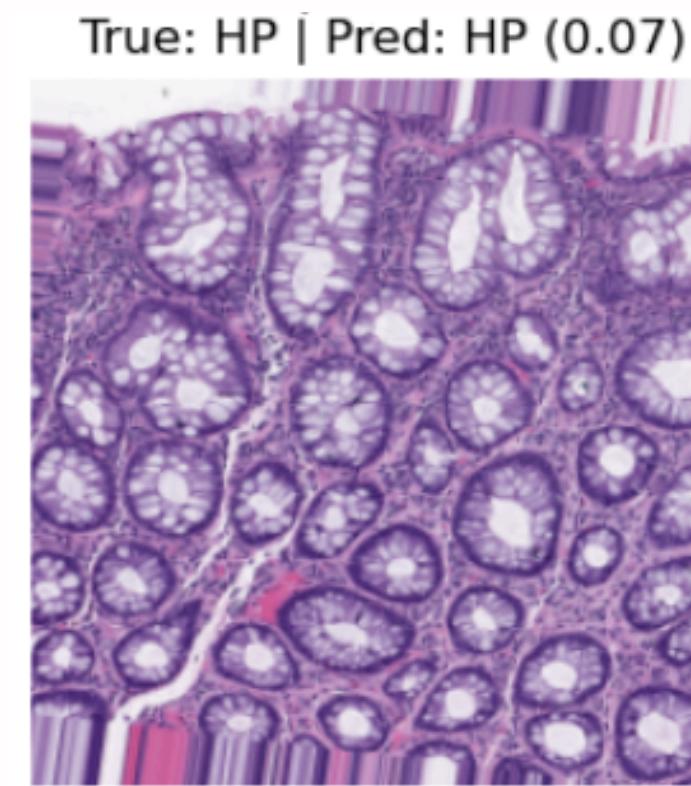
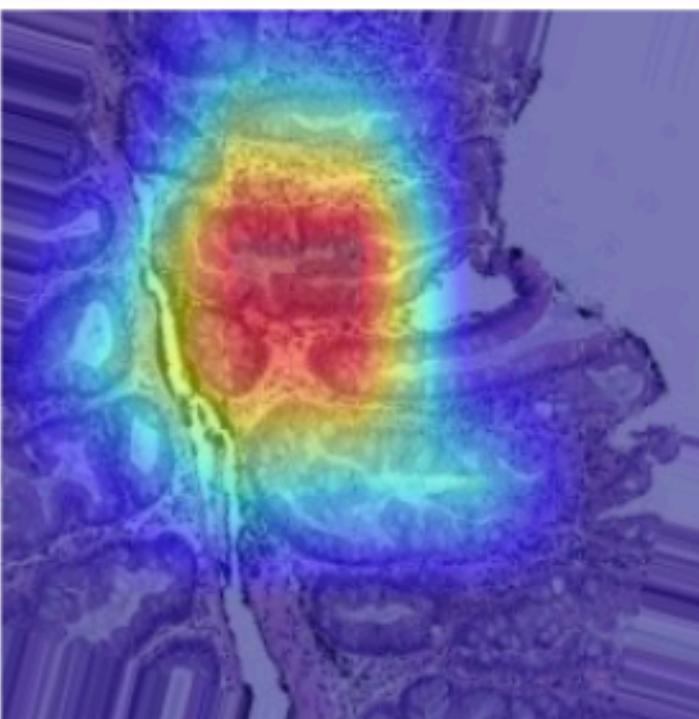
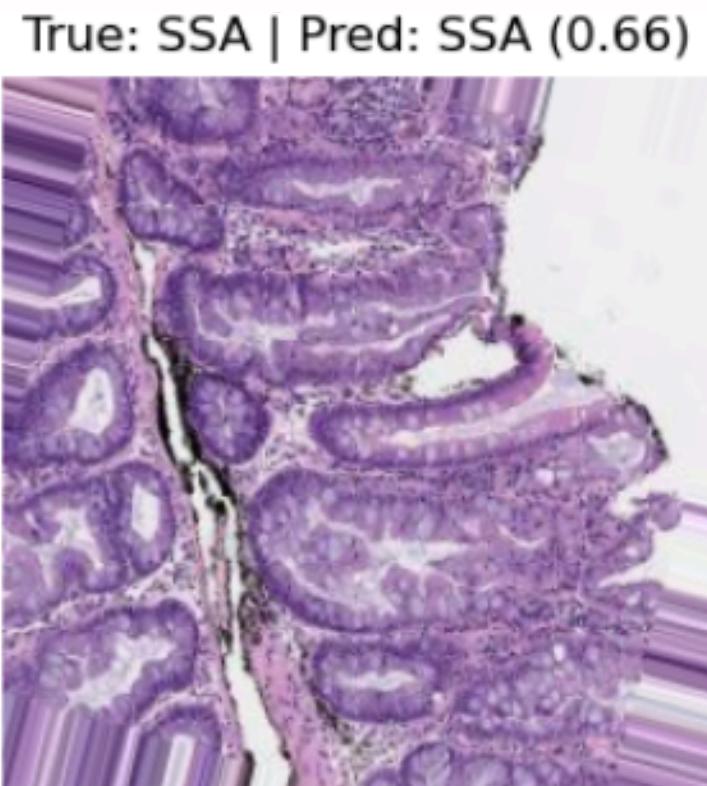
Model AUC



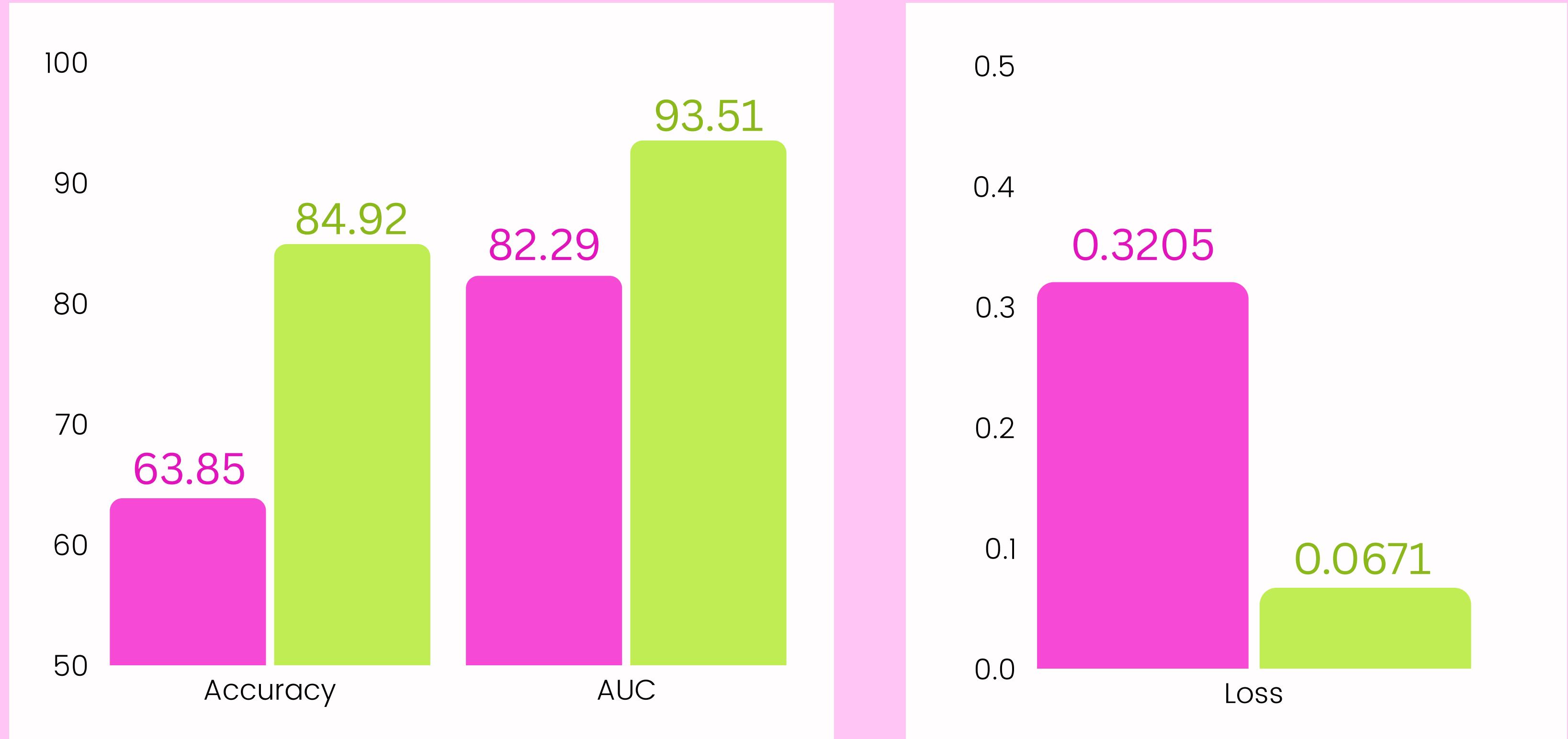
DenseNet121: after search and fine-tuning

	Training	Validation
Accuracy	0.8868	0.8492
Loss	0.0537	0.0671
AUC	0.9544	0.9351

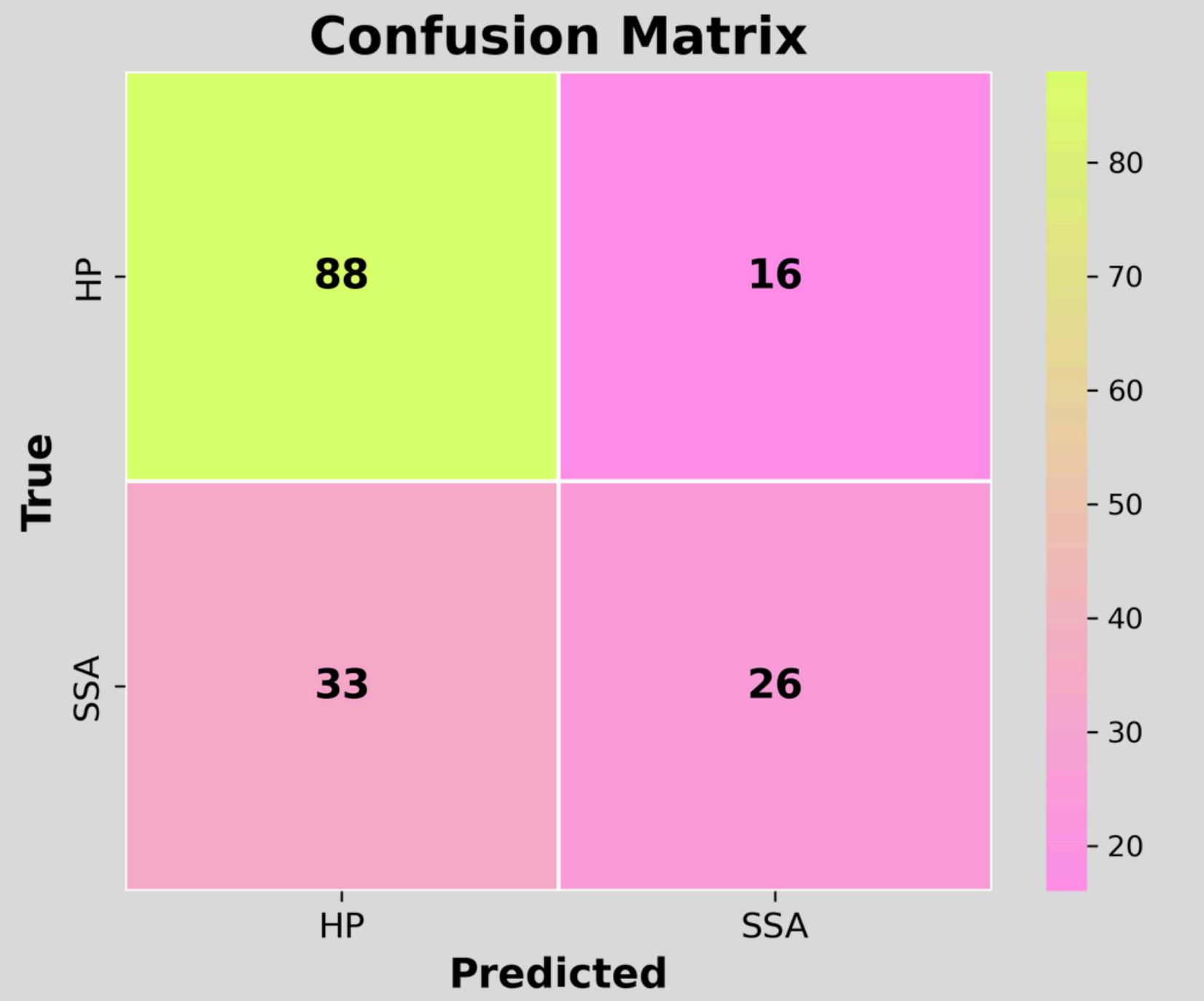
wow



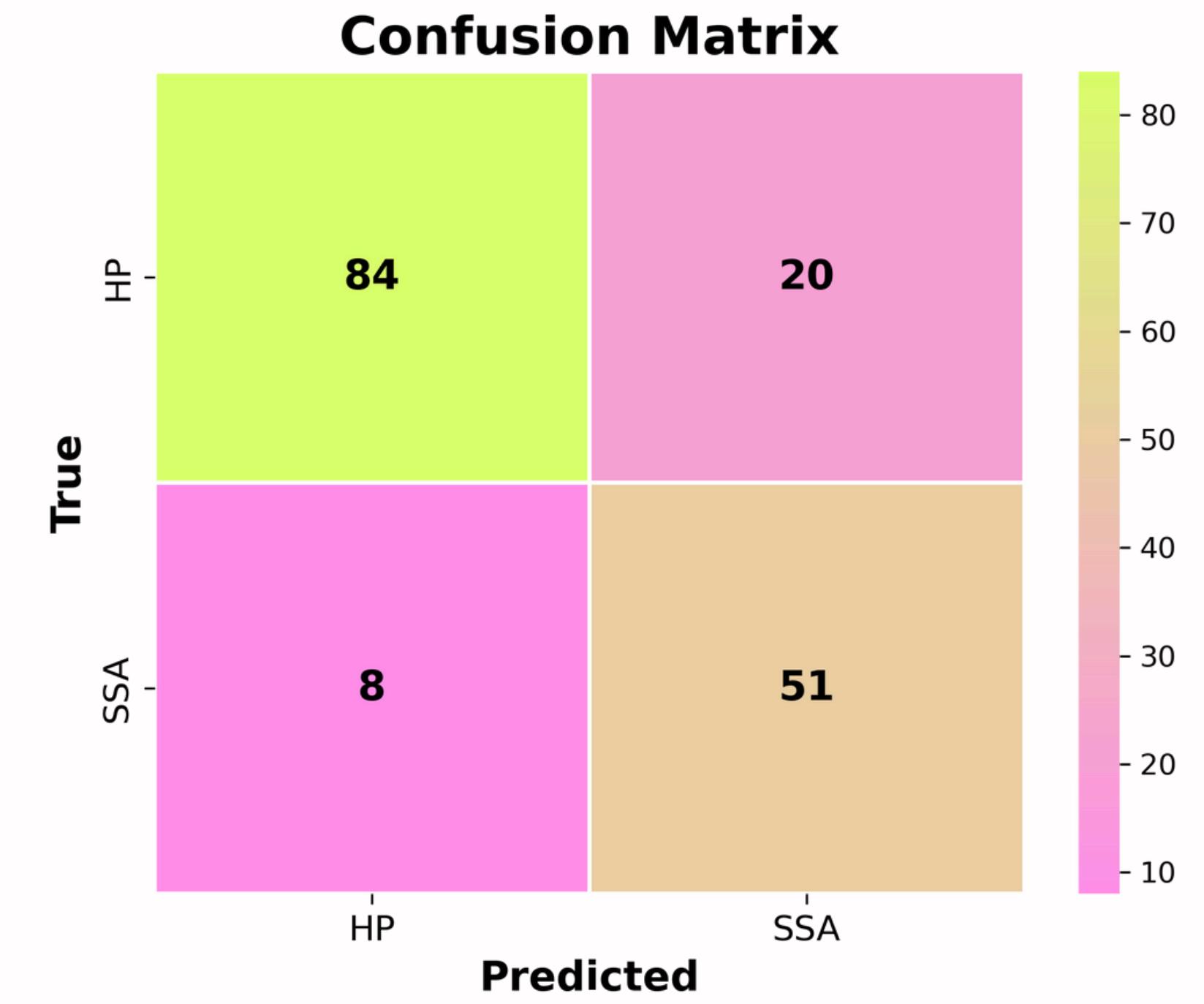
CNN vs DenseNet121



CNN



DenseNet121



Challenges



UNCERTAIN LABELING

Votes:
3 (4) out of 7
pathologists



IMBALANCE OF CASES

Less SSA in the
dataset - model is
biased



PREFER WORSE DIAGNOSIS

Prefer SSA over HP:
SSA needs urgent
treatment, we do not
want to miss it



**Exclude
uncertain cases**



**Focal loss function:
alpha=0.75**



**Focal loss function:
gamma = 2**

Conclusion & perspective

- Model may support pathologist's conclusion
- Needs additional work to eliminate false negatives to be more reliable as a diagnostic tool

Outcompeted:

- 2021 original MHIST dataset paper using ResNet18 by 0,41% AUC;
- 2023 paper [1] using DenseNet (84.39% AUC) by 9,12%

Did not reach:

- [1]'s ConvNeXt 93,91%

REDUCE IMPACT OF COLOR

RGB to YUV (or YCbCr) to separate the color information into luminance and chrominance.

MORE COMPUTATIONAL POWER

More epochs for longer training and better results

FIND AND COMBINE DIFFERENT DATASETS (HP & SSA HISTOLOGY)

Get more information for a fully usable model

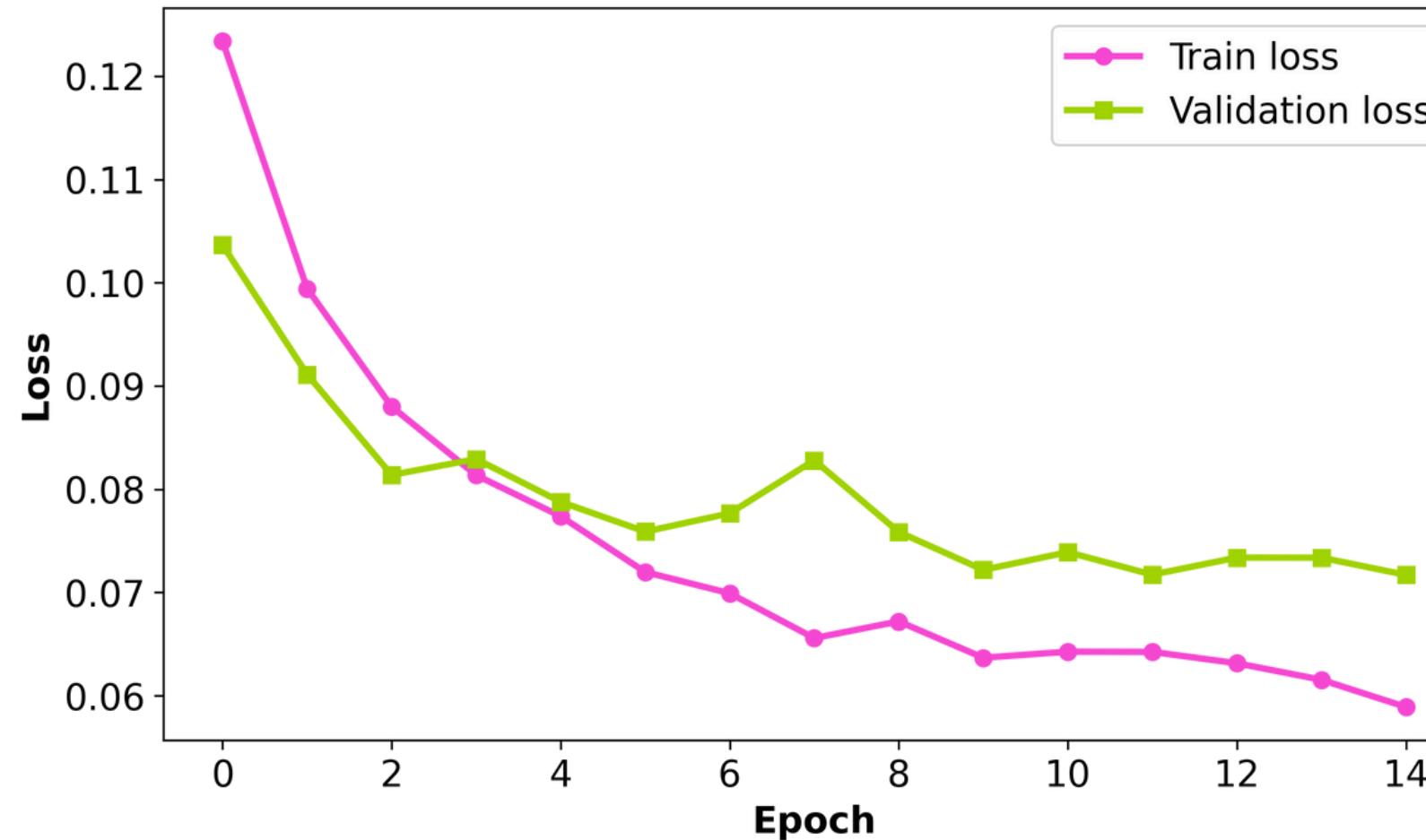


**Comments and
suggestions are
welcome!**

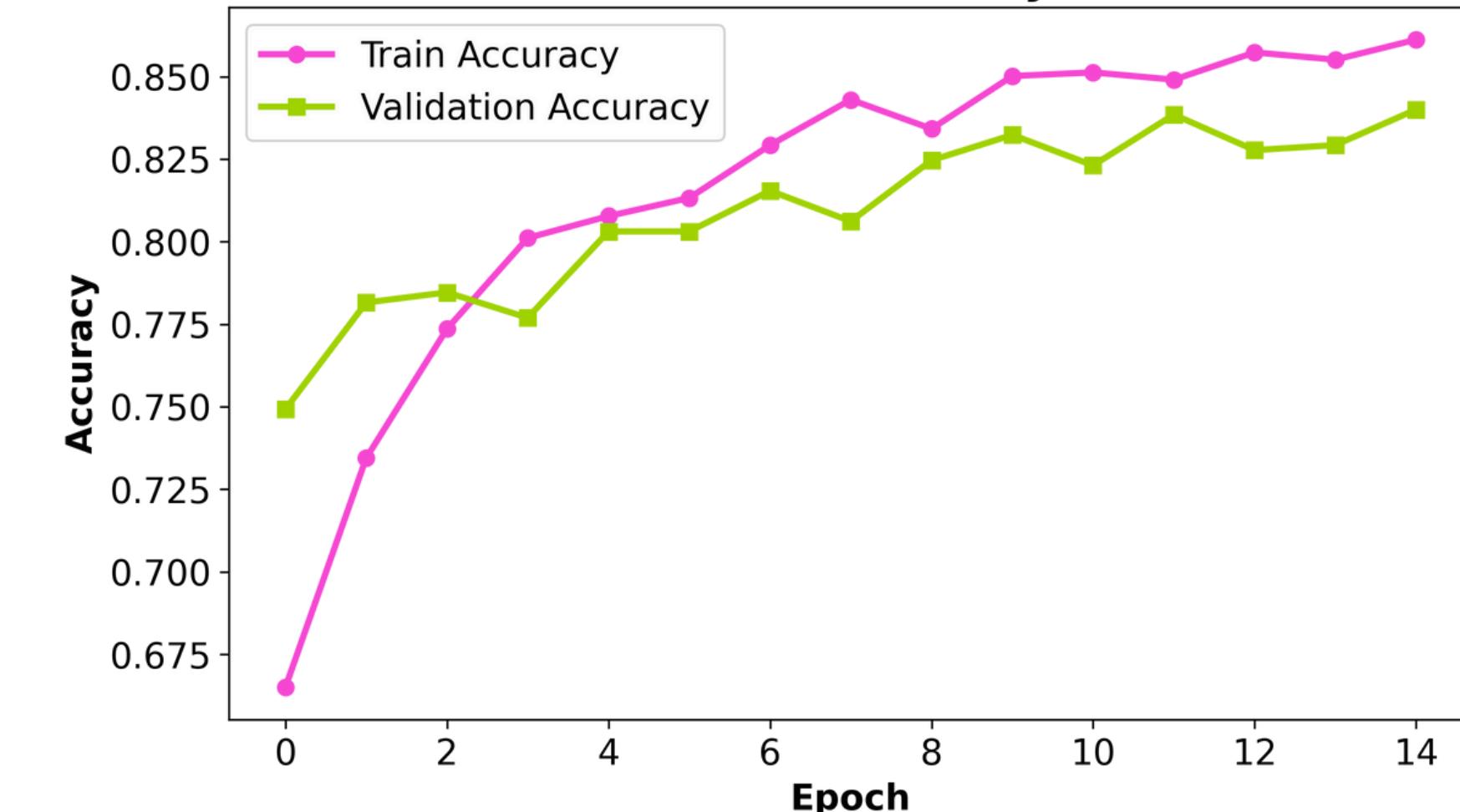
*Mikhail Melnichenko,
Daria Fedorova*



Model loss

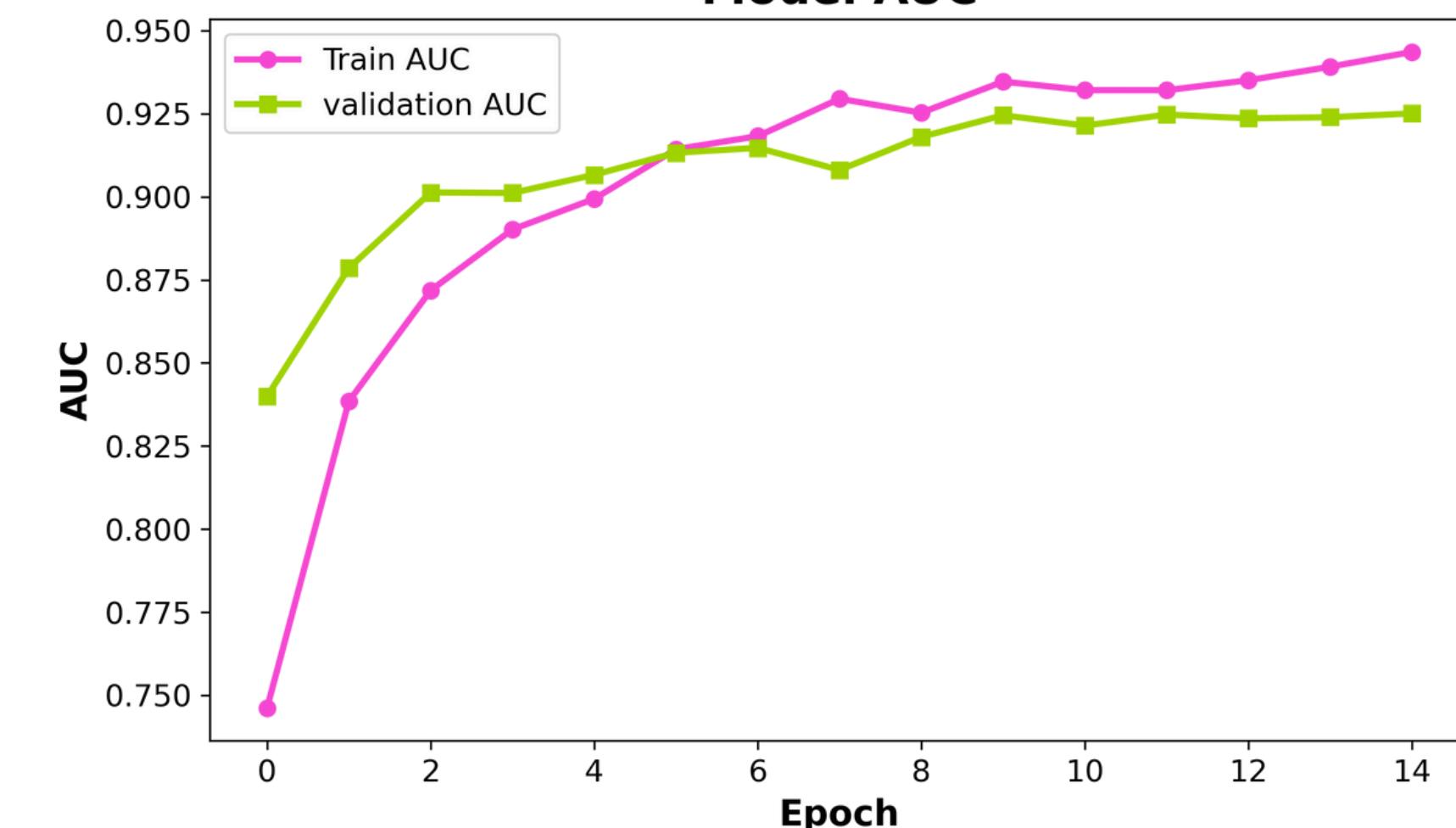


Model Accuracy



DenseNet121:
after search
and before
fine-tuning

Model AUC



```
1  def build_model(hp):
2      model = models.Sequential([
3          layers.Input(shape=(224, 224, 3)),
4
5          layers.Conv2D(hp.Int('conv1_filters', 32, 128, step=32), (3, 3), padding='same'),
6          layers.BatchNormalization(),
7          layers.ReLU(),
8          layers.MaxPooling2D(),
9          layers.Dropout(hp.Float('dropout1', 0.2, 0.5, step=0.1)),
10
11         layers.Conv2D(hp.Int('conv2_filters', 64, 256, step=64), (3, 3), padding='same'),
12         layers.BatchNormalization(),
13         layers.ReLU(),
14         layers.MaxPooling2D(),
15         layers.Dropout(hp.Float('dropout2', 0.2, 0.5, step=0.1)),
16
17         layers.Conv2D(hp.Int('conv3_filters', 128, 512, step=128), (3, 3), padding='same'),
18         layers.BatchNormalization(),
19         layers.ReLU(),
20         layers.MaxPooling2D(),
21         layers.Dropout(hp.Float('dropout3', 0.2, 0.5, step=0.1)),
22
23         layers.Flatten(),
24         layers.Dense(hp.Int('dense_units', 128, 512, step=64)),
25         layers.BatchNormalization(),
26         layers.ReLU(),
27         layers.Dropout(hp.Float('dropout4', 0.2, 0.5, step=0.1)),
28
29         layers.Dense(1, activation='sigmoid')
30     ])
31     model.compile(
32         optimizer=optimizers.Adam(learning_rate=hp.Choice('lr', [1e-5])),
33         loss=focal_loss(gamma=hp.Choice('gamma', [1.5, 2.0]), alpha=hp.Choice('alpha', [0.5, 0.75])),
34         metrics=[
35             'accuracy',
36             metrics.AUC(name='auc'),
37             metrics.Recall(name='tpr'),
38             metrics.FalsePositives(name='fp'),
39             metrics.TruePositives(name='tp'),
40         ]
41     )
42     return model
```

Repository:

<https://github.com/Jesuisgmo/ColoVision>