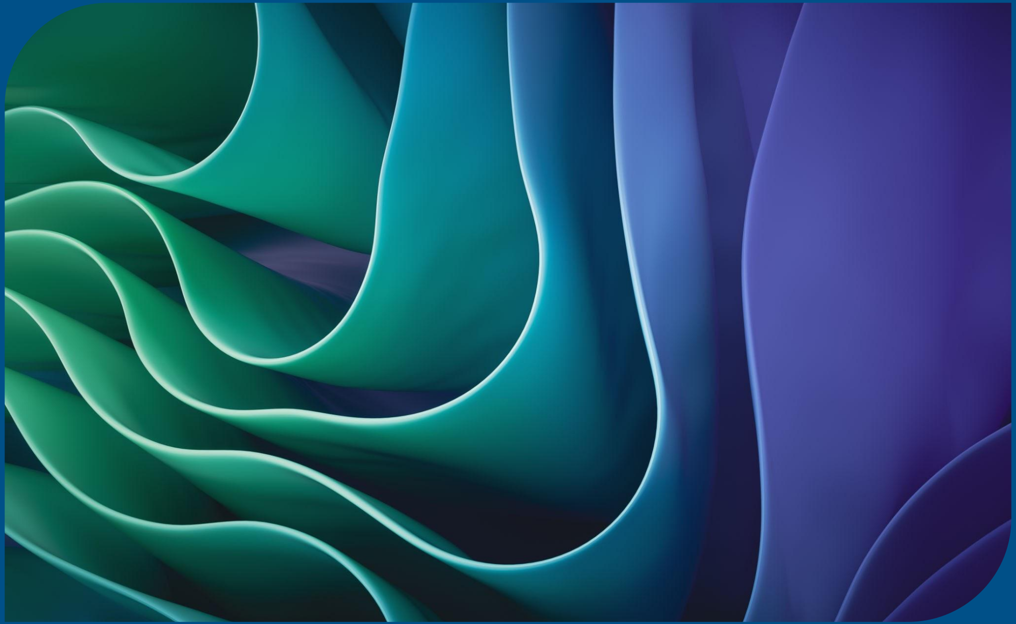
Cancer Detection via Deep Learning



Analyzed by : Thomas Sigmund

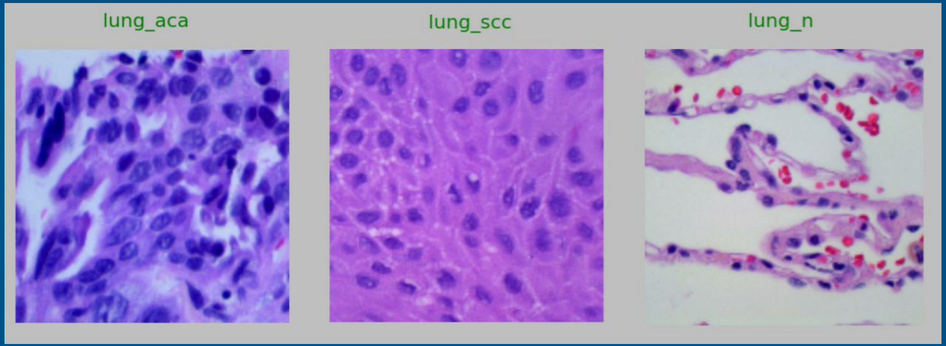
December 2024

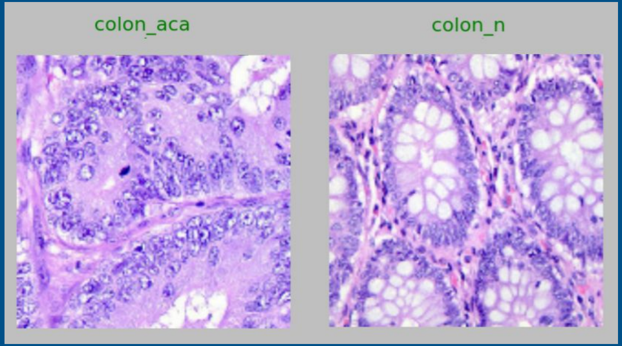
Analysis Data

Histopathological Images

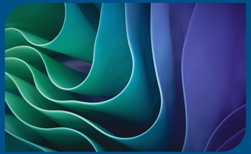
Lung tissue
- Adenocarcinoma (ACA)
- Squamous Cell Carcinoma (SCC)
- Benign

Colon tissue
- Adenocarcinoma (ACA)
- Benign





Key Objectives



- **Questions to answer**

  Is this a cancerous tissue sample?
  What type of cancer is it?

- **Generate the algorithm**

  Deep Learning using Convolutional Neural Networks (CNNs)

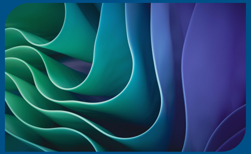
- **Tools to support Medical Doctors**

  Cancer Detection via Medical Apps

- **Commitment to Ongoing Improvement**

  Improve Algorithm
  Extending Analysis to Various Cancer Types

Evaluation



Data Sets

Cancerous images

lung_aca	Adenocarcinoma (ACA)	5000 images
lung_scc	Squamous Cell Carcinoma (SCC)	5000 images
lung_aca_scc	ACA & SCC	10000 images
colon_aca	ACA	5000 images

Benign images

lung_n	Benign tissue	5000 images
colon_n	Benign tissue	5000 images

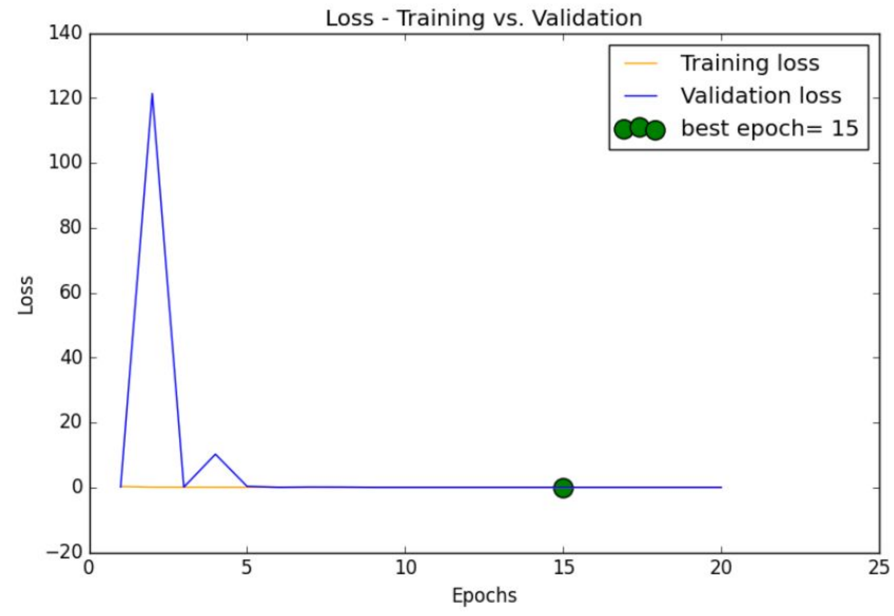
Loss / Accuracy

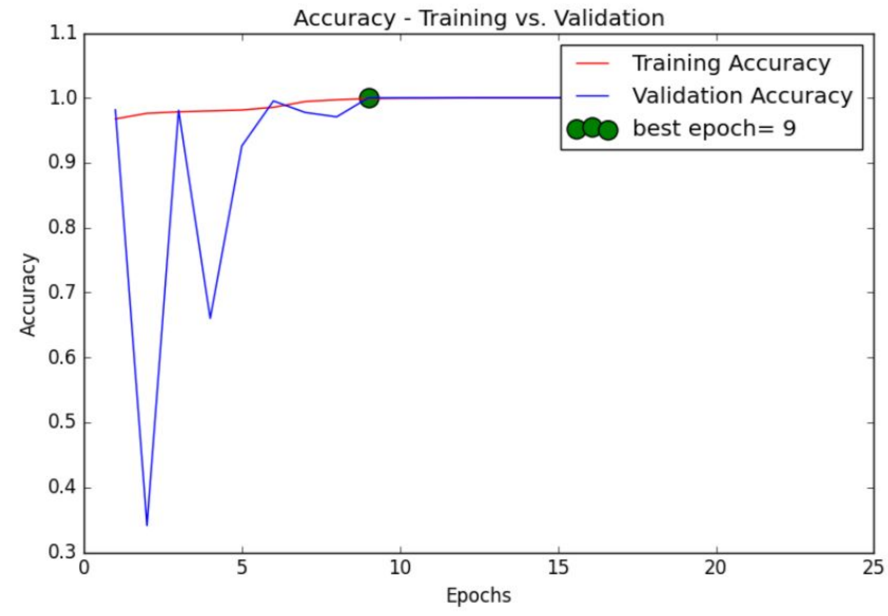
```
375/375 ━━━━━━━━━━━━━━━━━━━━ 4s 10ms/step
46/46 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step -
46/46 ━━━━━━━━━━━━━━━━━━━━ 25s 9ms/step -
Train Loss:  0.0007009753026068211
Train Accuracy:  0.9999166131019592
--------------------
Val Loss:  0.0008406831766478717
Val Accuracy:  1.0
--------------------
Test Loss:  0.0016087711555883288
Test Accuracy:  0.999320626258850l
```

Model Performance

Loss

Accuracy





Cancer Detection – Tools

Five Analysis Scenarios

Lung Tissue Classification

Cancer aca & scc vs. benign tissue

Cancer aca vs. benign tissue

Cancer scc vs. benign tissue

Cancer aca vs. scc

Colon Tissue Classification

Cancer aca vs. benign tissue

Insights

Supporting Diagnosis Process

Timely Diagnosis
Assisting Medical Doctors in Stressful Daily Routines
Reducing Human Error: A Second Opinion Backup

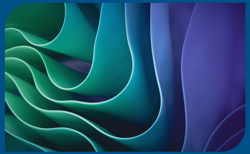
Conclusions from Results

Prioritize potentially cancerous samples
Identify the type of cancer
Faster initiation of cancer treatment

Reliability

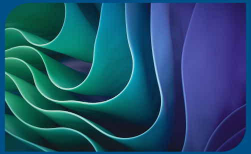
High Accuracy Rate Leading to Correct Predictions

Next Steps



- Improve Algorithm
- Extending Analysis to Various Cancer Types
- Analyze Multiple Images
- Robustness with Variable Image Sizes

Final Thoughts



- **Fascinating World of Deep Learning**
  Especially Convolutional Neural Networks
- **Promising Future for Model Improvement**
  Improved CNN Algorithms and More Powerful Computing Resources
- **Helpful Tools**
  Assisting Medical Doctors in Stressful Daily Routines
- But they are still tools – ***"the doctor has the final say."***

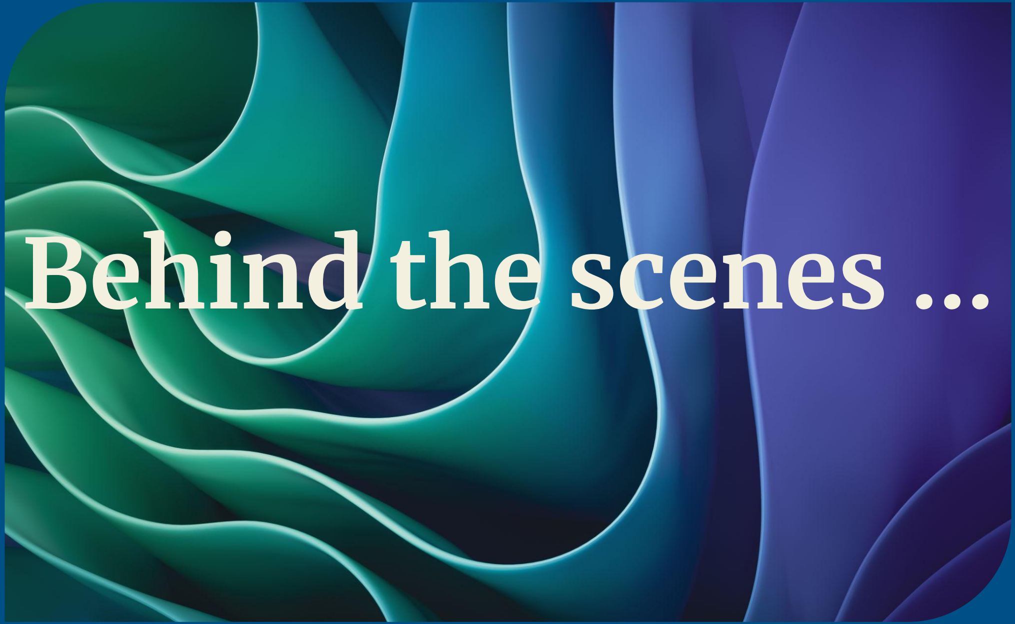
Cancer Detection via Deep Learning

Thank you :)

Analyzed by : Thomas Sigmund

December 2024

Cancer Detection via Deep Learning



Analyzed by : Thomas Sigmund

December 2024

Strategy 1: Differentiate between lung_aca_scc and lung_n

```
Uses a categorical classification approach.
The classes specified are ['lung_aca_scc', 'lung_n'].
Uses class_mode='categorical' and a final dense layer with two neurons and softmax activation for multi-class classification.
```

Strategy 2: Differentiate between lung_aca and lung_scc

```
Uses a categorical classification approach.
The classes specified are ['lung_aca', 'lung_scc'].
Uses class_mode='categorical' and a final dense layer with two neurons and softmax activation for multi-class classification.
```

Strategy 3: Differentiate between lung_aca and lung_n

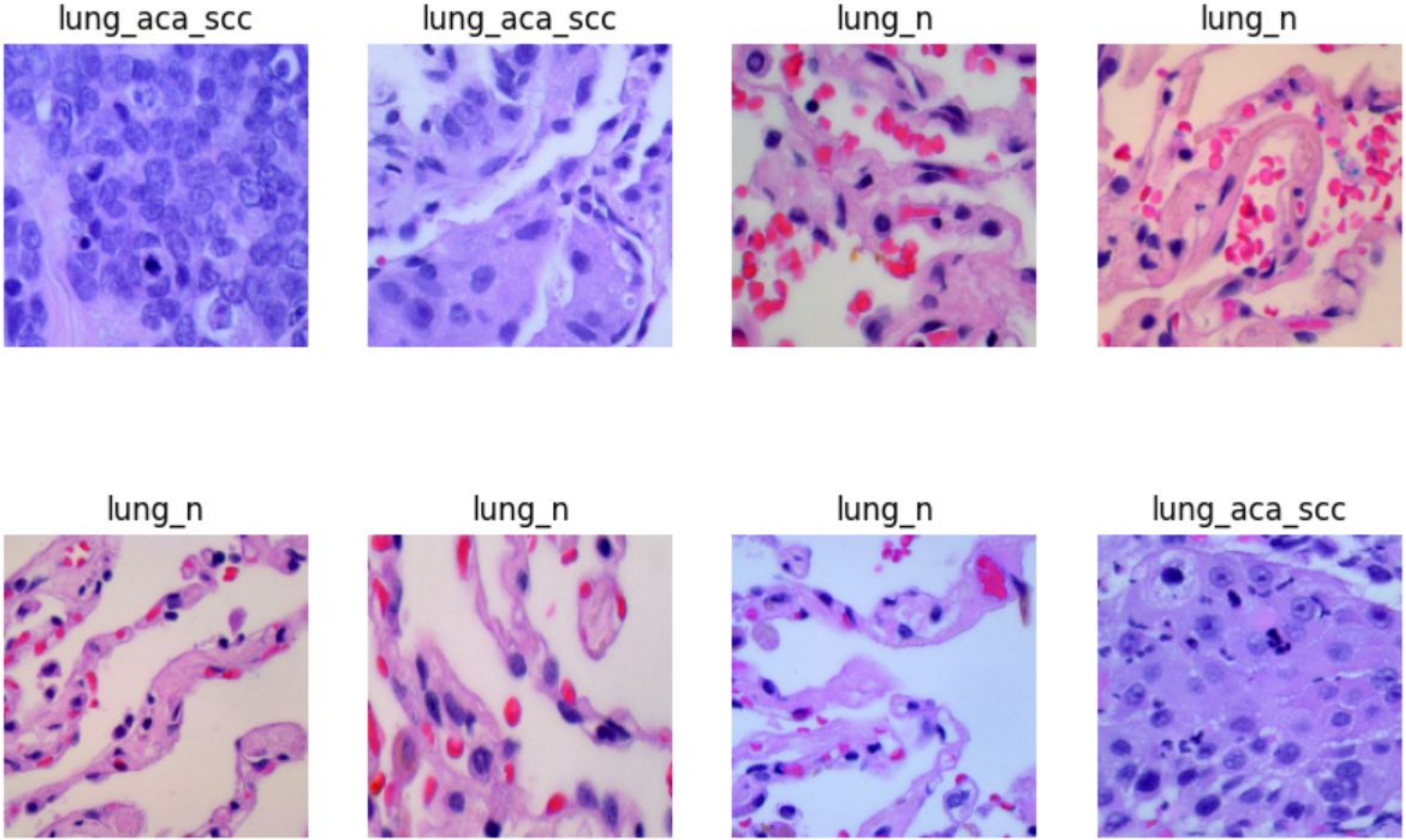
```
Uses a categorical classification approach.
The classes specified are ['lung_aca', 'lung_n'].
Uses class_mode='categorical' and a final dense layer with two neurons and softmax activation for multi-class classification.
```

Strategy 4: Differentiate between lung_scc and lung_n

```
Uses a categorical classification approach.
The classes specified are ['lung_scc', 'lung_n'].
Uses class_mode='categorical' and a final dense layer with two neurons and softmax activation for multi-class classification.
```

Strategy 5: Differentiate between colon_aca and colon_n

```
Uses a categorical classification approach.
The classes specified are ['colon_aca', 'colon_n'].
Uses class_mode='categorical' and a final dense layer with two neurons and softmax activation for multi-class classification.
```


lung_aca_scc
lung_aca_scc
lung_n
lung_n
lung_n
lung_n
lung_n
lung_aca_scc

BUILD MODEL

[14]

```
# Building a simple CNN model for classification
cnn_model = tf.keras.models.Sequential([
    tf.keras.layers.Input(shape=(150, 150, 3)),  # Input layer with image dimensions
    # tf.keras.layers.Conv2D(32, (3, 3), activation='relu'),  # First convolutional layer
    tf.keras.layers.Conv2D(32, (3, 3), padding = 'same', activation='relu'),  # First convolutional layer
    tf.keras.layers.MaxPooling2D(2, 2),  # Max-pooling layer to reduce dimensions
    tf.keras.layers.Conv2D(64, (3, 3), padding = 'same', activation='relu'),  # Second convolutional layer
    tf.keras.layers.MaxPooling2D(2, 2),  # Max-pooling layer
    tf.keras.layers.Conv2D(128, (3, 3), padding = 'same', activation='relu'),  # Third convolutional layer
    tf.keras.layers.MaxPooling2D(2, 2),  # Max-pooling layer
    tf.keras.layers.Conv2D(256, (3, 3), padding = 'same', activation='relu'),  # Fourth convolutional layer
    tf.keras.layers.MaxPooling2D(2, 2),  # Max-pooling layer
    tf.keras.layers.BatchNormalization(),
    tf.keras.layers.Flatten(),  # Flatten the feature maps
    tf.keras.layers.Dense(512, activation='relu'),  # Fully connected layer with 512 units
    tf.keras.layers.Dense(2, activation='softmax')  # Output layer for categorical classification
])
```

```
1 cnn_model.summary()
```

Model: "sequential"

Layer (type)	Output Shape	Param #
conv2d (Conv2D)	(None, 150, 150, 32)	896
max_pooling2d (MaxPooling2D)	(None, 75, 75, 32)	0
conv2d_1 (Conv2D)	(None, 75, 75, 64)	18,496
max_pooling2d_1 (MaxPooling2D)	(None, 37, 37, 64)	0
conv2d_2 (Conv2D)	(None, 37, 37, 128)	73,856
max_pooling2d_2 (MaxPooling2D)	(None, 18, 18, 128)	0
conv2d_3 (Conv2D)	(None, 18, 18, 256)	295,168
max_pooling2d_3 (MaxPooling2D)	(None, 9, 9, 256)	0
batch_normalization (BatchNormalization)	(None, 9, 9, 256)	1,024
flatten (Flatten)	(None, 20736)	0
dense (Dense)	(None, 512)	10,617,344
dense_1 (Dense)	(None, 2)	1,026

Total params: 11,007,810 (41.99 MB)
Trainable params: 11,007,298 (41.99 MB)
Non-trainable params: 512 (2.00 KB)

```python
tensorboard = tf.keras.callbacks.TensorBoard(log_dir = 'logs')
checkpoint = tf.keras.callbacks.ModelCheckpoint("cancer.keras", monitor="val_loss", save_best_only=True, mode="auto", verbose = 1)
reduce_lr_on_plateau = tf.keras.callbacks.ReduceLROnPlateau(monitor = 'val_loss', factor = 0.3, patience = 2, min_delta = 0.001,
                              mode='auto', verbose = 1)
early_stopping = callbacks.EarlyStopping(monitor='val_loss', patience=5, mode='min', restore_best_weights=True)
```

```python
# cnn_model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])        # TS - initial variant
cnn_model.compile(optimizer='adam', loss='sparse_categorical_crossentropy', metrics=['accuracy'])

```

```python
# history = cnn_model.fit(train_ds, validation_data = val_ds, epochs = 10, batch_size = 32, verbose = 1,
#               callbacks=[tensorboard, checkpoint, early_stopping, reduce_lr_on_plateau])

# history = cnn_model.fit(train_ds, validation_data = val_ds, epochs = 20, batch_size = 32, verbose = 1,
#               callbacks=[tensorboard, checkpoint, early_stopping, reduce_lr_on_plateau])

history = cnn_model.fit(train_ds, validation_data = val_ds, epochs = 40, batch_size = 32, verbose = 1,
             callbacks=[tensorboard, checkpoint, early_stopping, reduce_lr_on_plateau])

```

```
Epoch 1/40
```

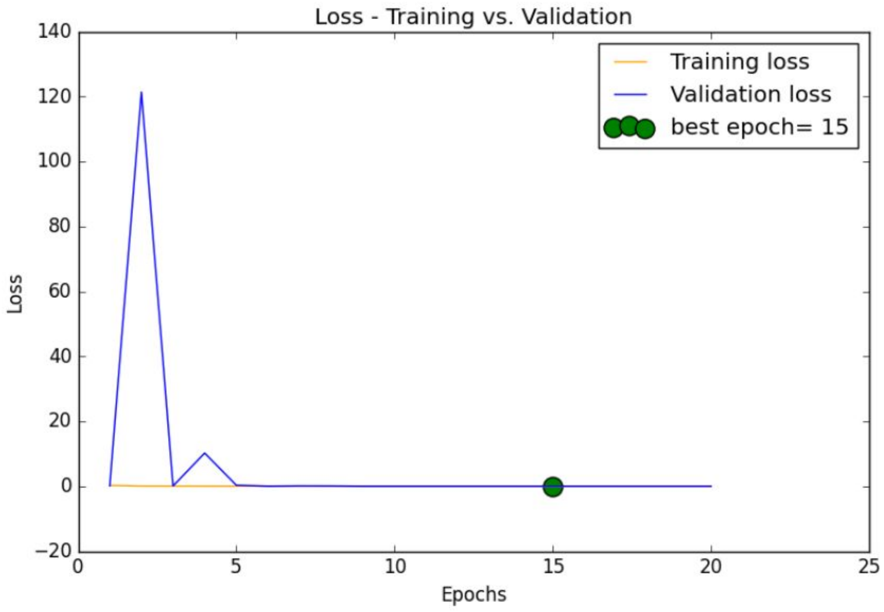
[18]

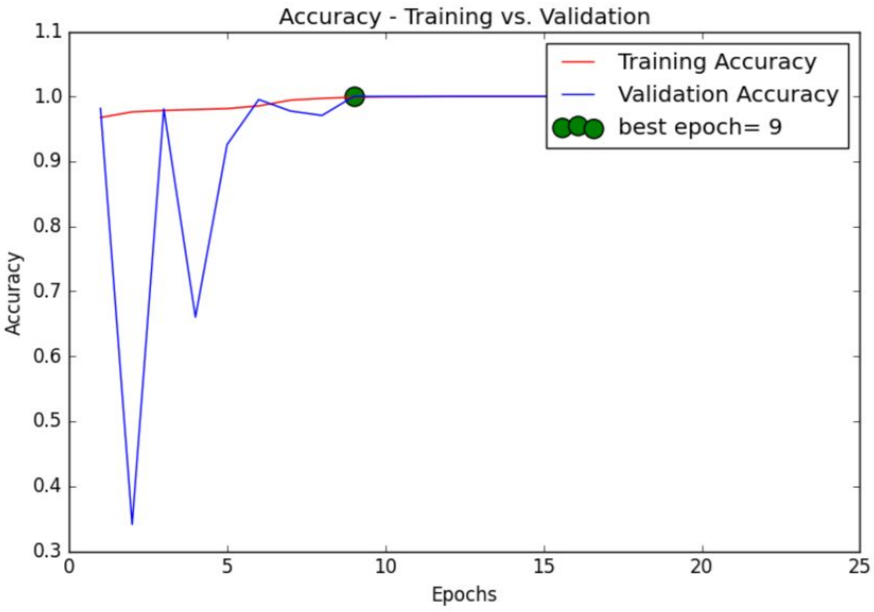
```
374/375 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step - accuracy: 1.0000 - loss: 9.7650e-04
Epoch 16: val_loss did not improve from 0.00084
375/375 ━━━━━━━━━━━━━━━━━━━━ 10s 28ms/step - accuracy: 1.0000 - loss: 9.7691e-04 - val_accuracy: 1.0000 - val_loss: 0.0010 - learning_rate: 2.4300e-06
Epoch 17/40
374/375 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step - accuracy: 1.0000 - loss: 9.4159e-04
Epoch 17: val_loss did not improve from 0.00084

Epoch 17: ReduceLROnPlateau reducing learning rate to 7.289999985005124e-07.
375/375 ━━━━━━━━━━━━━━━━━━━━ 10s 28ms/step - accuracy: 1.0000 - loss: 9.4210e-04 - val_accuracy: 1.0000 - val_loss: 9.2698e-04 - learning_rate: 2.4300e
Epoch 18/40
374/375 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step - accuracy: 1.0000 - loss: 8.4691e-04
Epoch 18: val_loss did not improve from 0.00084
375/375 ━━━━━━━━━━━━━━━━━━━━ 10s 28ms/step - accuracy: 1.0000 - loss: 8.4782e-04 - val_accuracy: 1.0000 - val_loss: 9.4034e-04 - learning_rate: 7.2900e
Epoch 19/40
374/375 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step - accuracy: 1.0000 - loss: 0.0010
Epoch 19: val_loss did not improve from 0.00084

Epoch 19: ReduceLROnPlateau reducing learning rate to 2.1870000637136398e-07.
375/375 ━━━━━━━━━━━━━━━━━━━━ 10s 28ms/step - accuracy: 1.0000 - loss: 0.0010 - val_accuracy: 1.0000 - val_loss: 9.5124e-04 - learning_rate: 7.2900e-07
Epoch 20/40
374/375 ━━━━━━━━━━━━━━━━━━━━ 0s 27ms/step - accuracy: 1.0000 - loss: 9.2422e-04
Epoch 20: val_loss did not improve from 0.00084
375/375 ━━━━━━━━━━━━━━━━━━━━ 10s 28ms/step - accuracy: 1.0000 - loss: 9.2466e-04 - val_accuracy: 1.0000 - val_loss: 9.7114e-04 - learning_rate: 2.1870e
```

EVALUATION

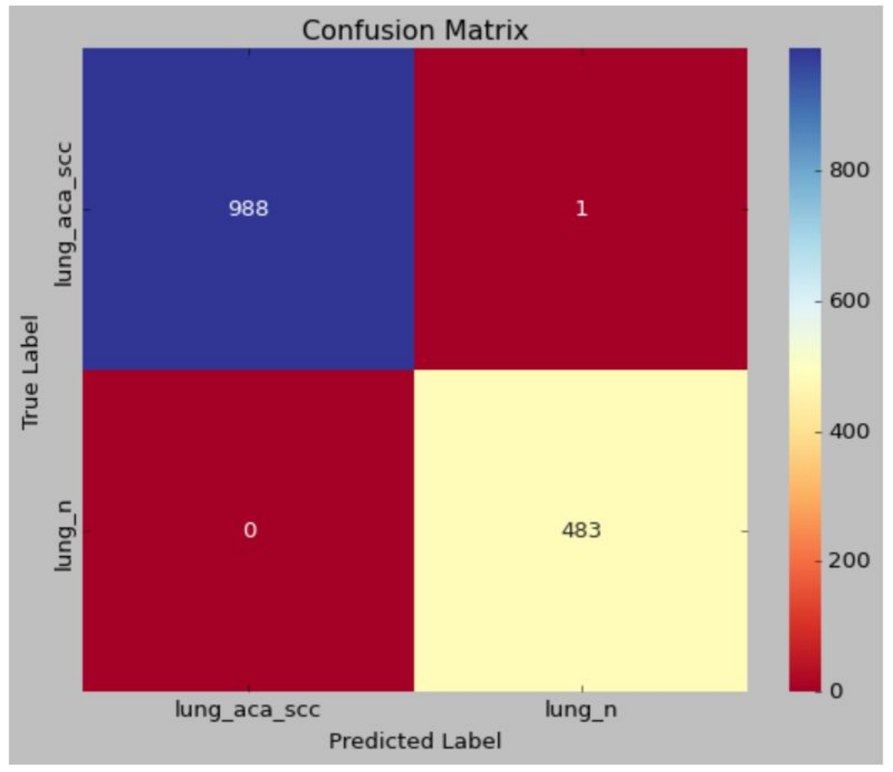
```
375/375 ━━━━━━━━━━━━━━━━━━━━ 4s 10ms/step - accuracy: 0.9999 - loss: 6.4018e-04
46/46 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - accuracy: 1.0000 - loss: 8.5412e-04
46/46 ━━━━━━━━━━━━━━━━━━━━ 25s 9ms/step - accuracy: 0.9990 - loss: 0.0020
Train Loss:  0.0007009753026068211
Train Accuracy:  0.9999166131019592
--------------------
Val Loss:  0.0008406831766478717
Val Accuracy:  1.0
--------------------
Test Loss:  0.0016087711555883288
Test Accuracy:  0.9993206262588501
```

MODEL PERFORMANCE



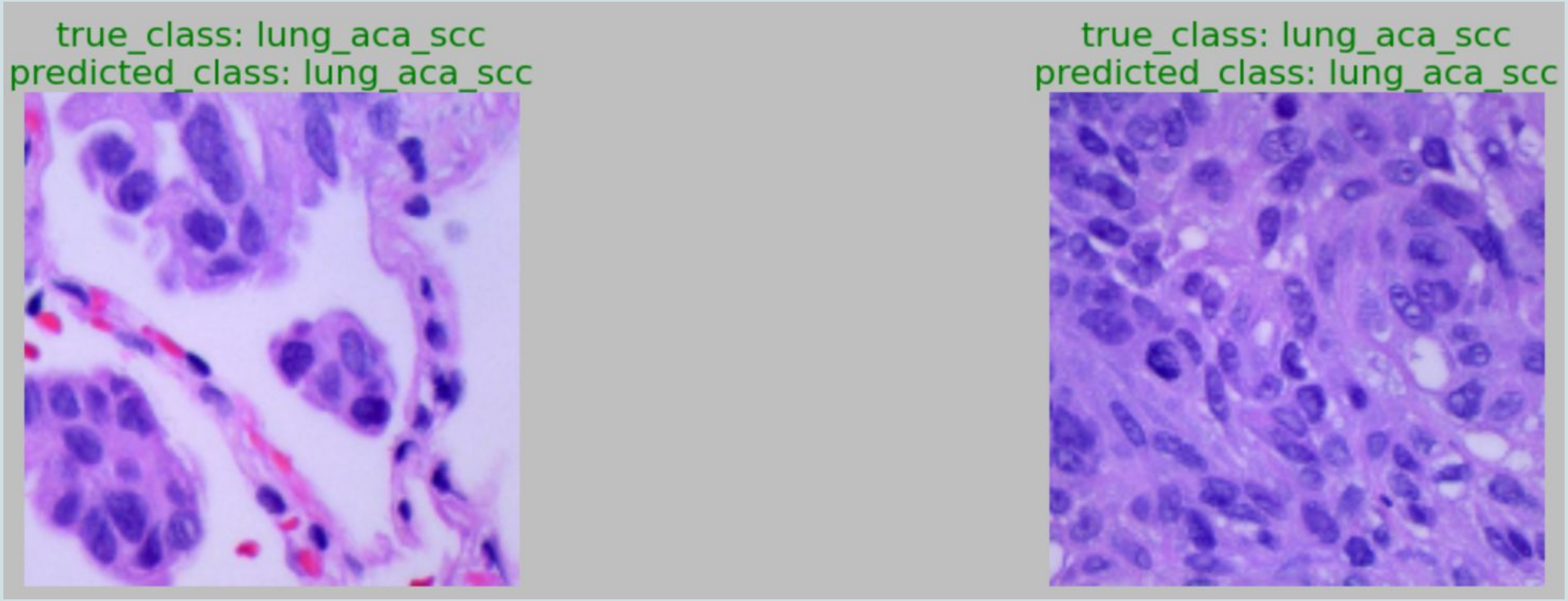


Confusion Matrix

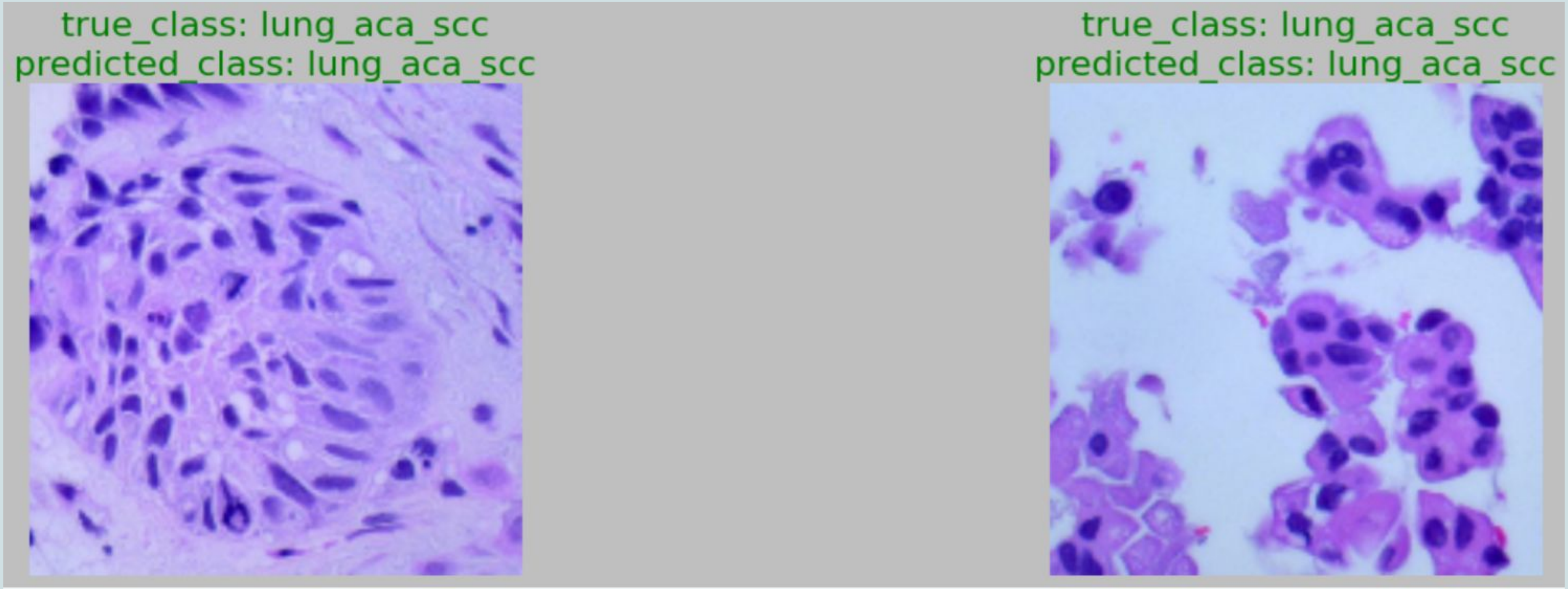


	precision	recall	f1-score	support
0	1.00	1.00	1.00	989
1	1.00	1.00	1.00	483
accuracy			1.00	1472
macro avg	1.00	1.00	1.00	1472
weighted avg	1.00	1.00	1.00	1472

Predictions

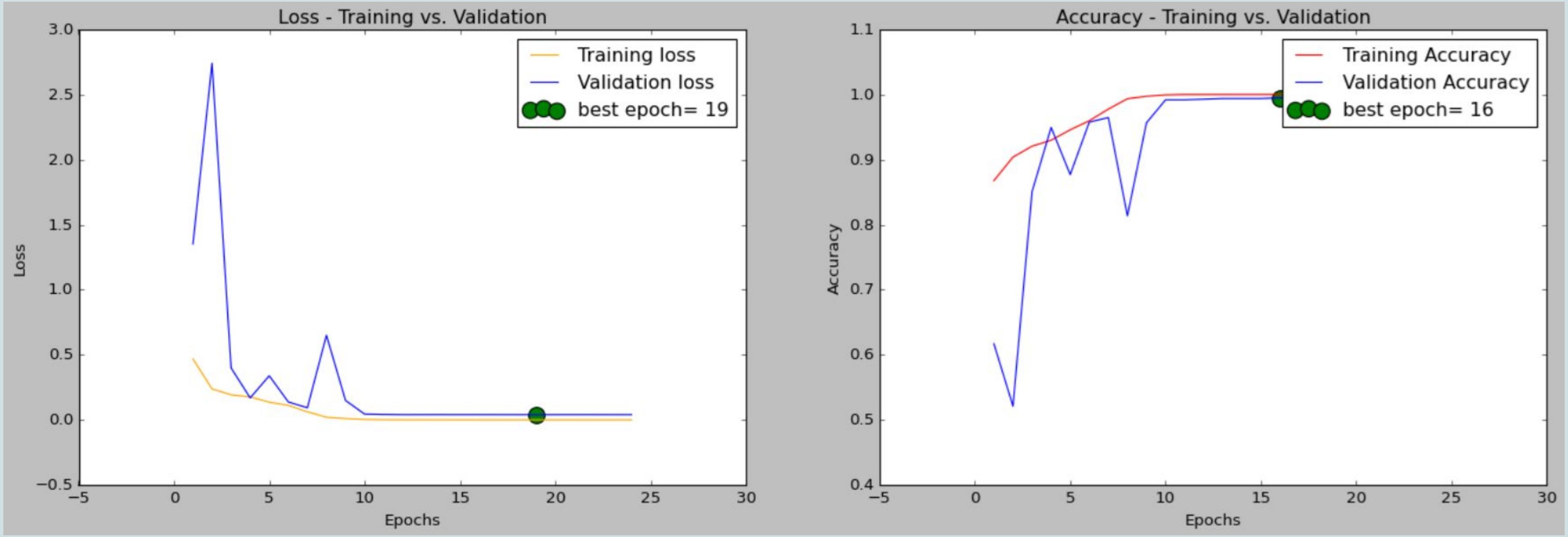


Predictions

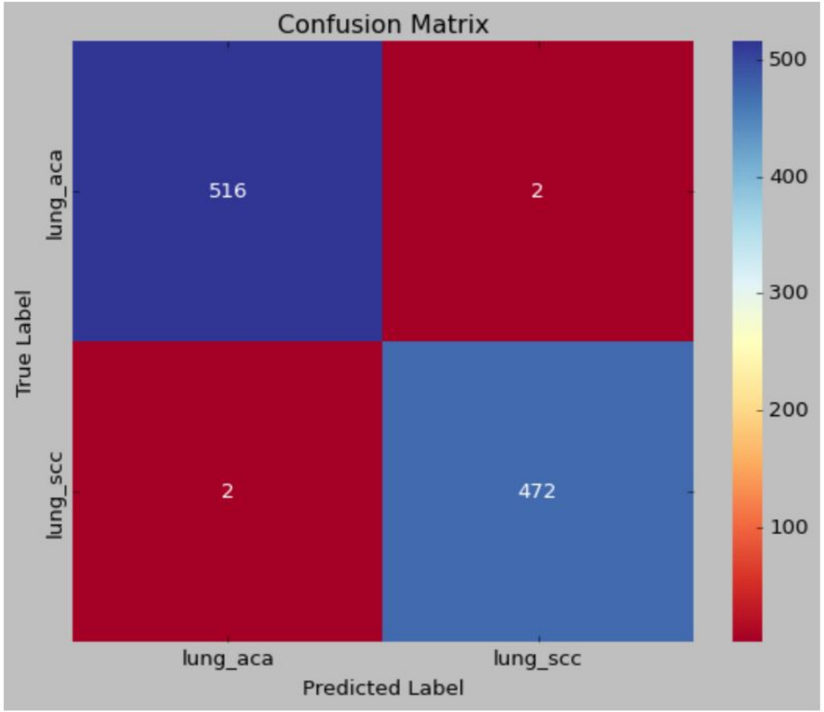


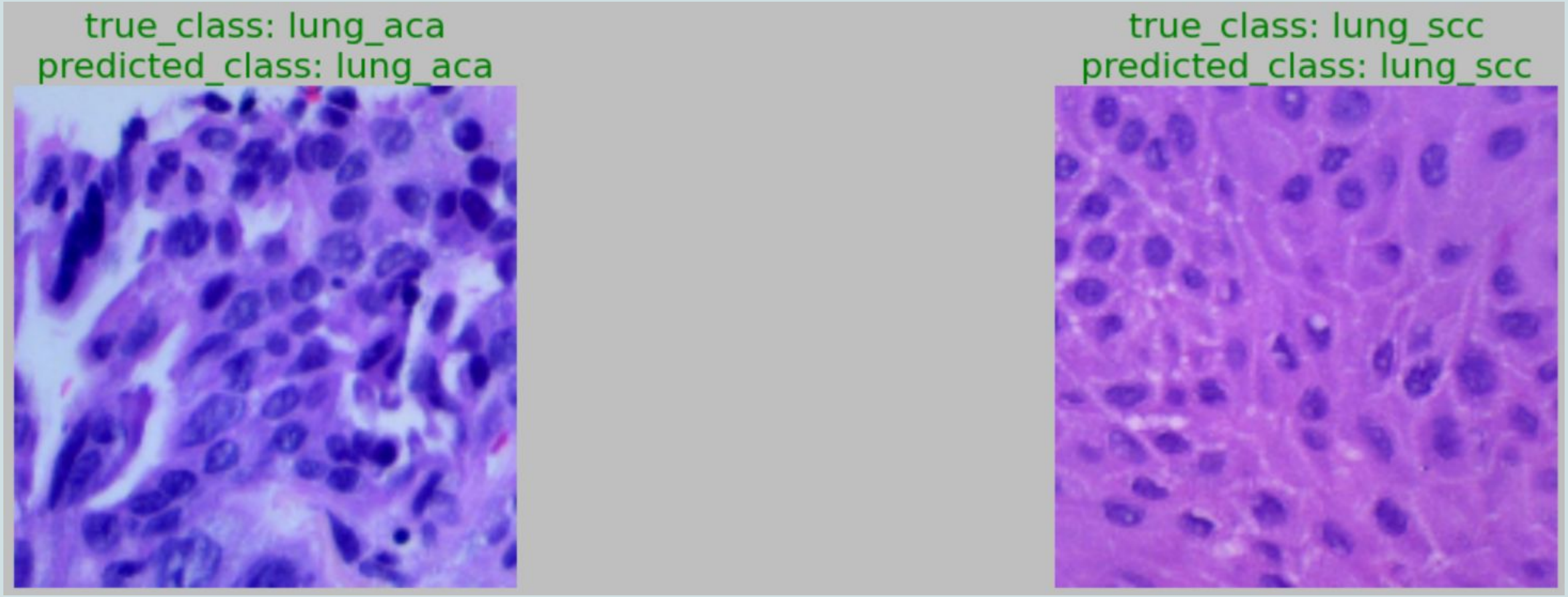
STRATEGY 2 – aca vs. scc

```
250/250 ━━━━━━━━━━━━━━━━━━━━ 3s 12ms/step - accuracy: 0.9990 - loss: 0.0019
31/31 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - accuracy: 0.9943 - loss: 0.0340
31/31 ━━━━━━━━━━━━━━━━━━━━ 17s 9ms/step - accuracy: 0.9961 - loss: 0.0249
Train Loss:  0.0015269367722794414
Train Accuracy:  0.9994990229606628
--------------------
Val Loss:  0.039725691080093384
Val Accuracy:  0.9939516186714172
--------------------
Test Loss:  0.021339187398552895
Test Accuracy:  0.9959677457809448
```


Loss - Training vs. Validation
Training loss
Validation loss
best epoch= 19
Loss
Epochs
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−0.5
−5
0
5
10
15
20
25
30
Accuracy - Training vs. Validation
Training Accuracy
Validation Accuracy
best epoch= 16
Accuracy
Epochs
1.1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
−5
0
5
10
15
20
25
30

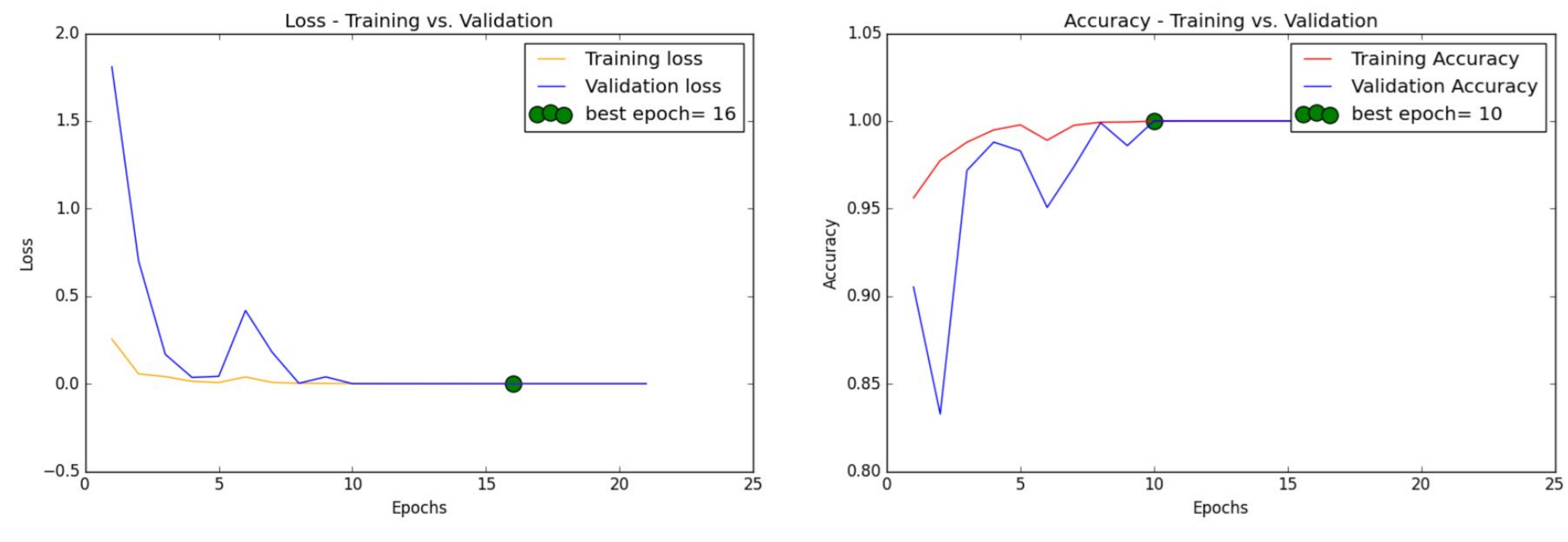
	precision	recall	f1-score	support
0	1.00	1.00	1.00	518
1	1.00	1.00	1.00	474
accuracy			1.00	992
macro avg	1.00	1.00	1.00	992
weighted avg	1.00	1.00	1.00	992



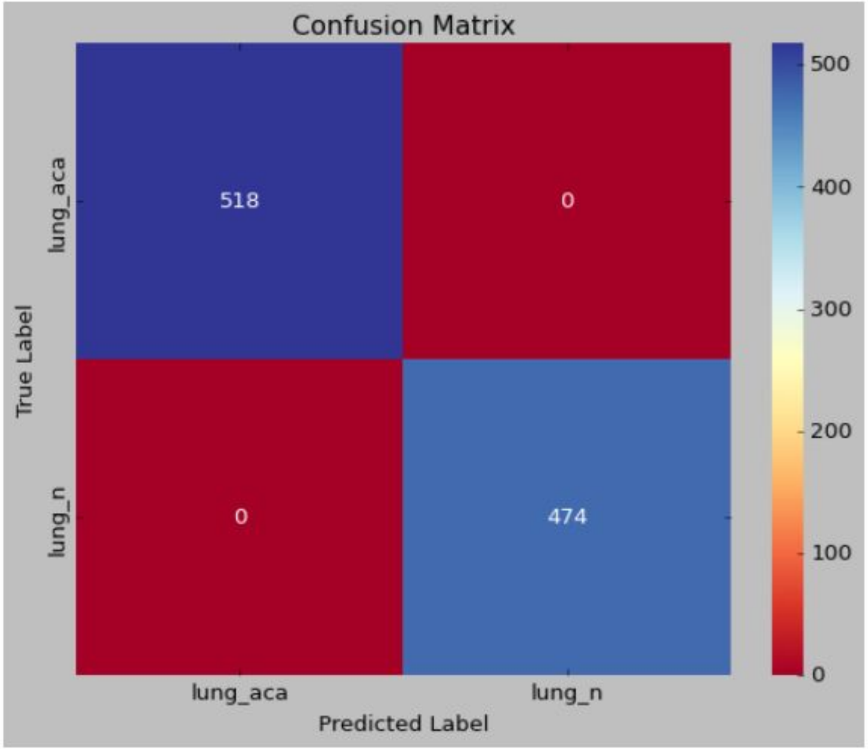

true_class: lung_aca
predicted_class: lung_aca
true_class: lung_scc
predicted_class: lung_scc

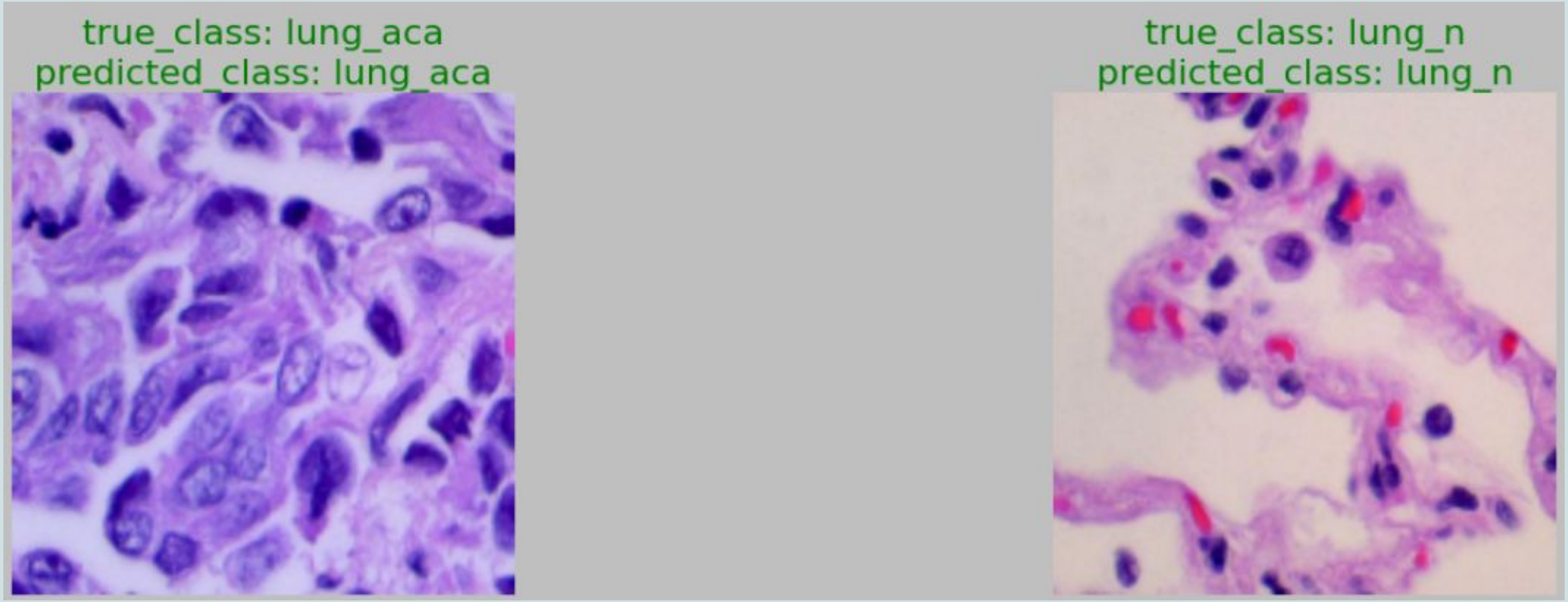
STRATEGY 3 – lung aca vs. n

```
250/250 ━━━━━━━━━━━━━━━━━━━━ 3s 11ms/step - accuracy: 1.0000 - loss: 1.6696e-04
31/31 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - accuracy: 1.0000 - loss: 1.4585e-04
31/31 ━━━━━━━━━━━━━━━━━━━━ 16s 10ms/step - accuracy: 1.0000 - loss: 2.2074e-04
Train Loss:  0.00015198372420854867
Train Accuracy:  1.0
--------------------
Val Loss:  0.00010955912875942886
Val Accuracy:  1.0
--------------------
Test Loss:  0.00022429967066273093
Test Accuracy:  1.0
```


Loss - Training vs. Validation
Training loss
Validation loss
best epoch= 16
Loss
Epochs
2.0
1.5
1.0
0.5
0.0
−0.5
0
5
10
15
20
25
Accuracy - Training vs. Validation
Training Accuracy
Validation Accuracy
best epoch= 10
Accuracy
Epochs
1.05
1.00
0.95
0.90
0.85
0.80

	precision	recall	f1-score	support
0	1.00	1.00	1.00	518
1	1.00	1.00	1.00	474
accuracy			1.00	992
macro avg	1.00	1.00	1.00	992
weighted avg	1.00	1.00	1.00	992

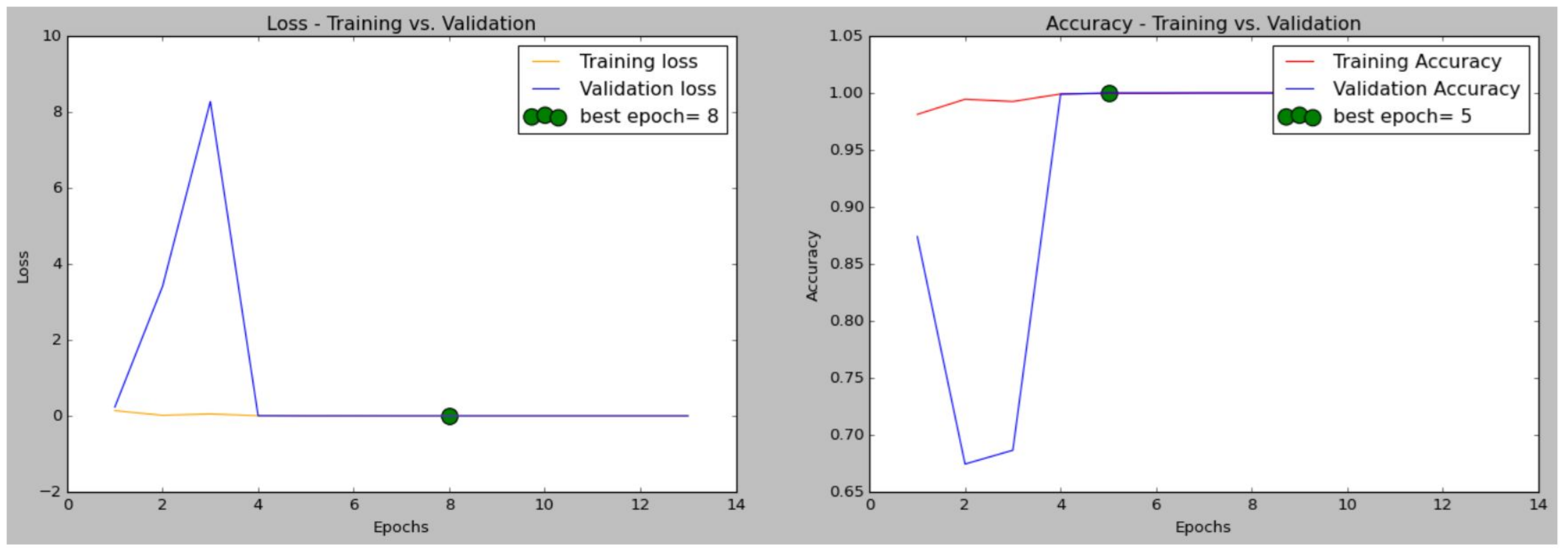




true_class: lung_aca
predicted_class: lung_aca
true_class: lung_n
predicted_class: lung_n

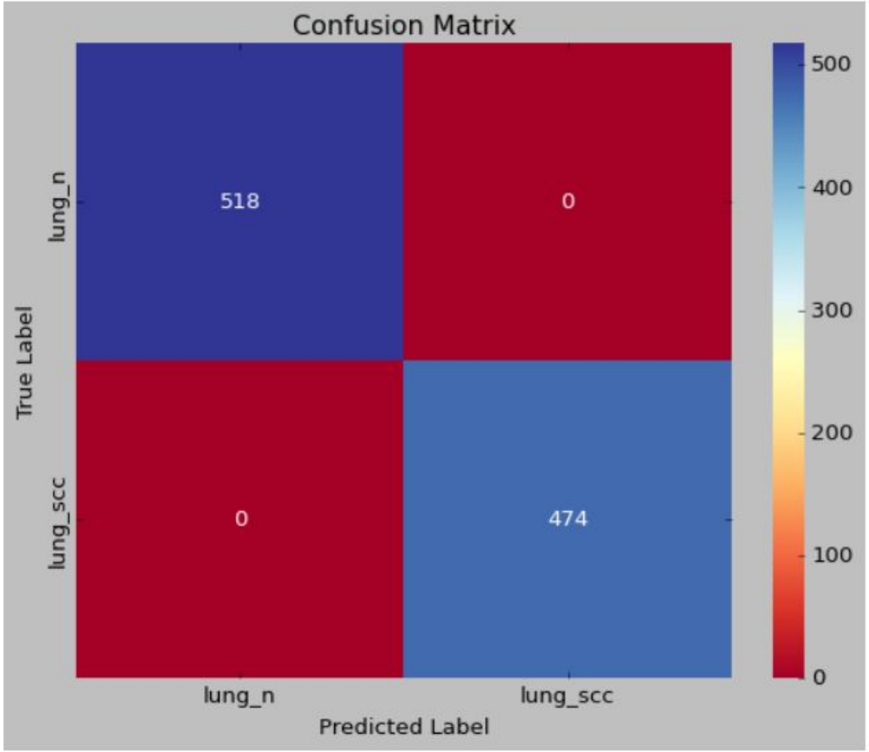
STRATEGY 4 – lung scc vs. n

```
250/250 ━━━━━━━━━━━━━━━━━━━━ 2s 10ms/step - accuracy: 1.0000 - loss: 2.4848e-05
31/31 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - accuracy: 1.0000 - loss: 5.2482e-06
31/31 ━━━━━━━━━━━━━━━━━━━━ 17s 10ms/step - accuracy: 1.0000 - loss: 3.1497e-04
Train Loss:  1.7837957784649916e-05
Train Accuracy:  1.0
--------------------
Val Loss:  3.531769380060723e-06
Val Accuracy:  1.0
--------------------
Test Loss:  0.00013467104872688652
Test Accuracy:  1.0
```

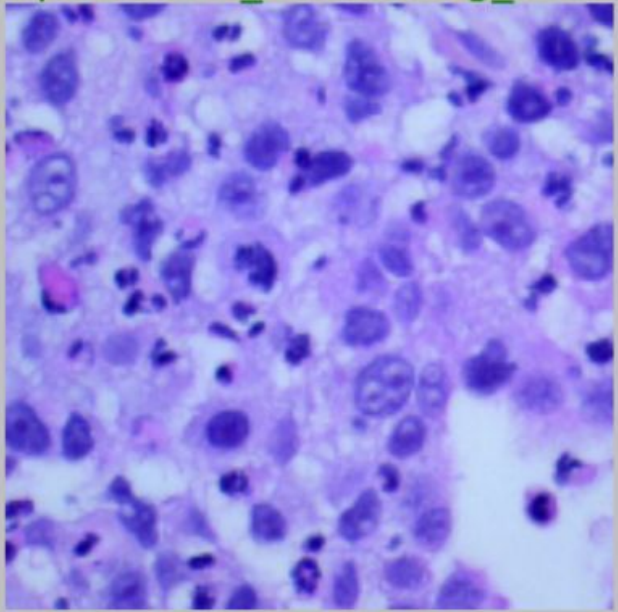


Loss - Training vs. Validation
Training loss
Validation loss
best epoch= 8
Loss
Epochs
10
8
6
4
2
0
−2
0
2
4
6
8
10
12
14
Accuracy - Training vs. Validation
Training Accuracy
Validation Accuracy
best epoch= 5
Accuracy
Epochs
1.05
1.00
0.95
0.90
0.85
0.80
0.75
0.70
0.65
0
2
4
6
8
10
12
14

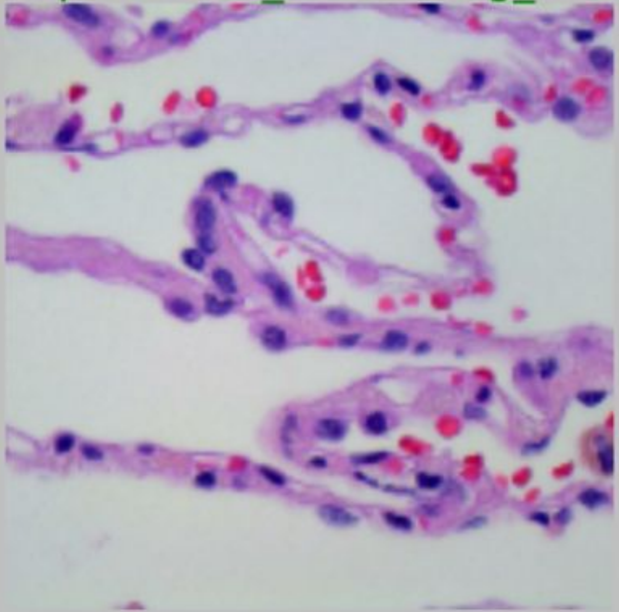
	precision	recall	f1-score	support
0	1.00	1.00	1.00	518
1	1.00	1.00	1.00	474
accuracy			1.00	992
macro avg	1.00	1.00	1.00	992
weighted avg	1.00	1.00	1.00	992



true_class: lung_scc
predicted_class: lung_scc

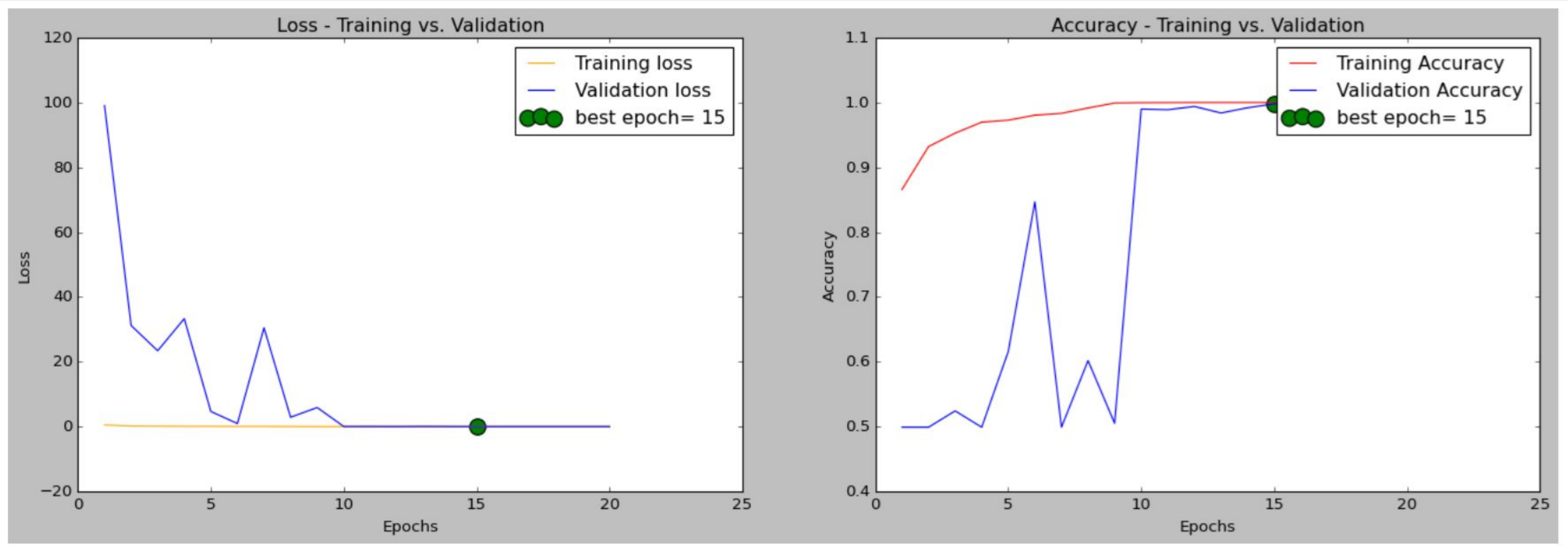


true_class: lung_n
predicted_class: lung_n

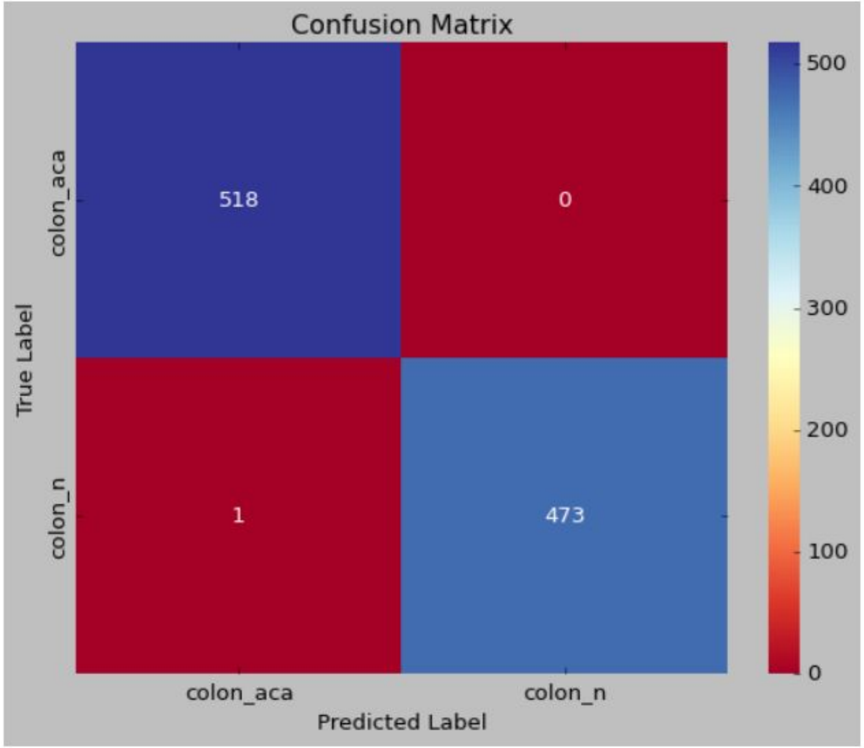


STRATEGY 5 – colon aca vs. n

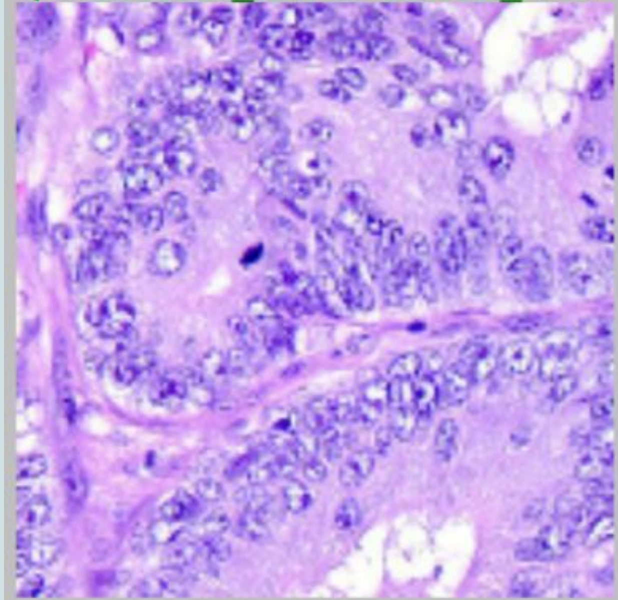
```
250/250 ━━━━━━━━━━━━━━━━━━━━ 2s 9ms/step - accuracy: 0.9999 - loss: 0.0013
31/31 ━━━━━━━━━━━━━━━━━━━━ 0s 7ms/step - accuracy: 0.9979 - loss: 0.0134
31/31 ━━━━━━━━━━━━━━━━━━━━ 17s 10ms/step - accuracy: 0.9986 - loss: 0.0074
Train Loss:  0.0012777356896549463
Train Accuracy:  0.999749481678009
--------------------
Val Loss:  0.009261605329811573
Val Accuracy:  0.9979838728904724
--------------------
Test Loss:  0.005203577224165201
Test Accuracy:  0.9989919066429138
```


Loss - Training vs. Validation
Training loss
Validation loss
best epoch= 15
Loss
Epochs
120
100
80
60
40
20
0
−20
0
5
10
15
20
25
Accuracy - Training vs. Validation
Training Accuracy
Validation Accuracy
best epoch= 15
Accuracy
Epochs
1.1
1.0
0.9
0.8
0.7
0.6
0.5
0.4
0
5
10
15
20
25

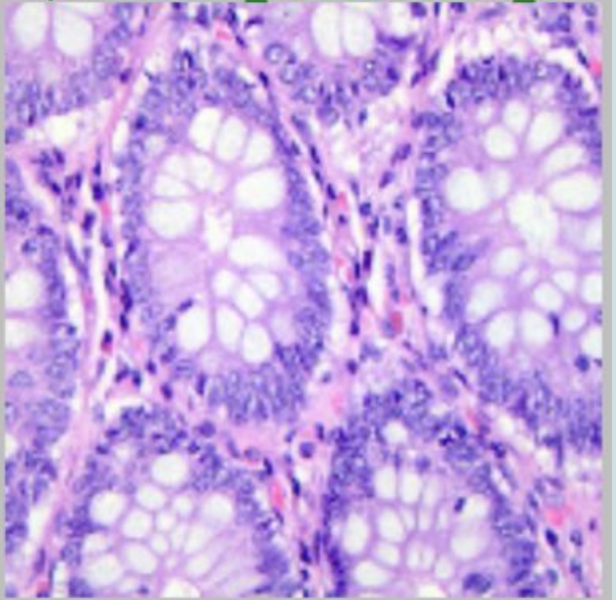
	precision	recall	f1-score	support
0	1.00	1.00	1.00	518
1	1.00	1.00	1.00	474
accuracy			1.00	992
macro avg	1.00	1.00	1.00	992
weighted avg	1.00	1.00	1.00	992



true_class: colon_aca
predicted_class: colon_aca



true_class: colon_n
predicted_class: colon_n




Time to rest :)