Brain Tumor MRI

Kaggle Data Set
This dataset contains **7023** images of human brain MRI images which are classified into 4 classes: **glioma** - **meningioma** - **no tumor** and **pituitary**.

In all the models here, 80% of train data is used for training and 20% for validation, test data is separate

Training data

Total 5715 images

category	Photos
Glioma	1312
meningioma	1339
no tumor	1595
pituitary	1457

Test data

Total 1311 images

category	Photos
Glioma	300
meningioma	306
no tumor	405
pituitary	300

Model 1

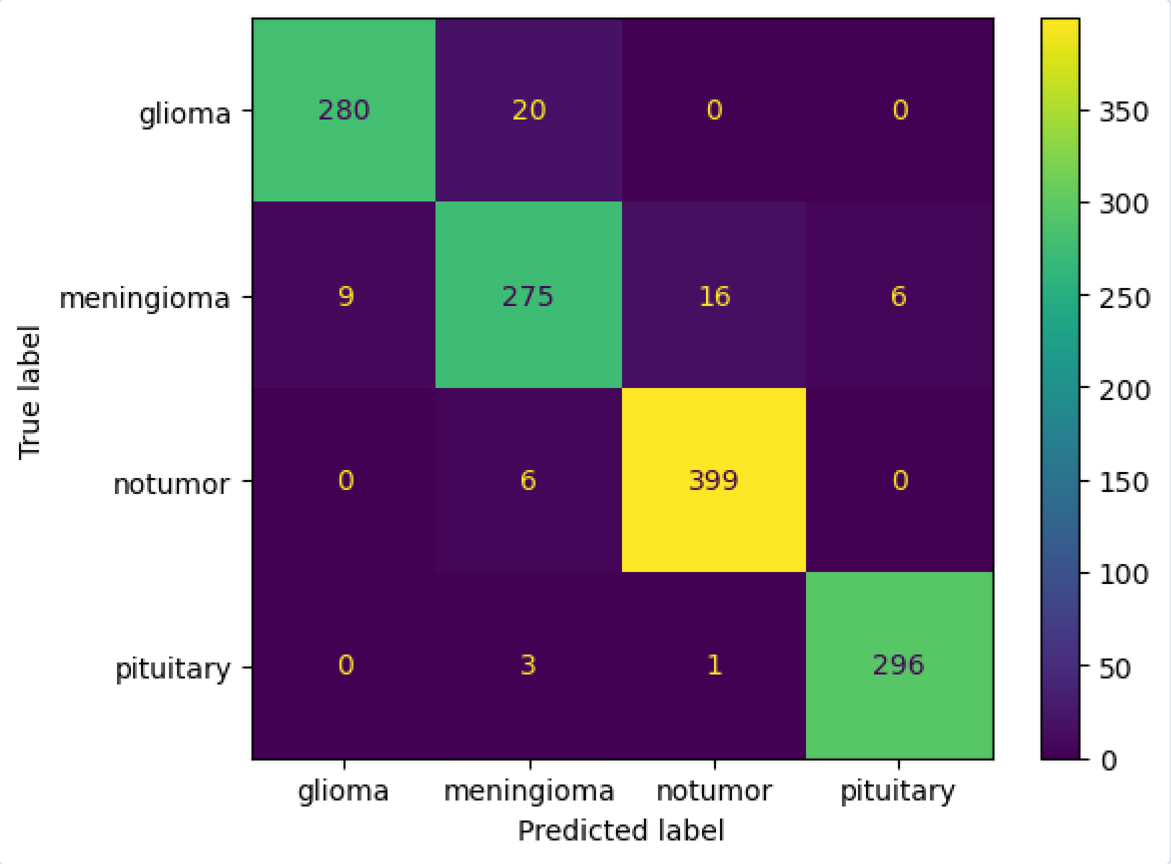
v1

- Custom Model
- all convolution layers have a `kernal size = 3` `stride = 1` `padding = valid`
- all max pooling layers have `pool_size = 2` `strides=1` `padding = valid`

```
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 rescaling_1 (Rescaling)     (None, 128, 128, 3)       0         
                                                                 
 conv2d_3 (Conv2D)           (None, 126, 126, 32)      896       
                                                                 
 max_pooling2d_3 (MaxPoolin  (None, 63, 63, 32)        0         
 g2D)                                                            
                                                                 
 conv2d_4 (Conv2D)           (None, 61, 61, 64)        18496     
                                                                 
 max_pooling2d_4 (MaxPoolin  (None, 30, 30, 64)        0         
 g2D)                                                            
                                                                 
 conv2d_5 (Conv2D)           (None, 28, 28, 128)       73856     
                                                                 
 max_pooling2d_5 (MaxPoolin  (None, 14, 14, 128)       0         
 g2D)                                                            
 flatten_1 (Flatten)         (None, 25088)             0

 dense_2 (Dense)             (None, 128)               3211392

 dense_3 (Dense)             (None, 4)                 516

=================================================================
Total params: 3305156 (12.61 MB)
Trainable params: 3305156 (12.61 MB)
Non-trainable params: 0 (0.00 Byte)
```

Performance Metrics

Confusion Matrix



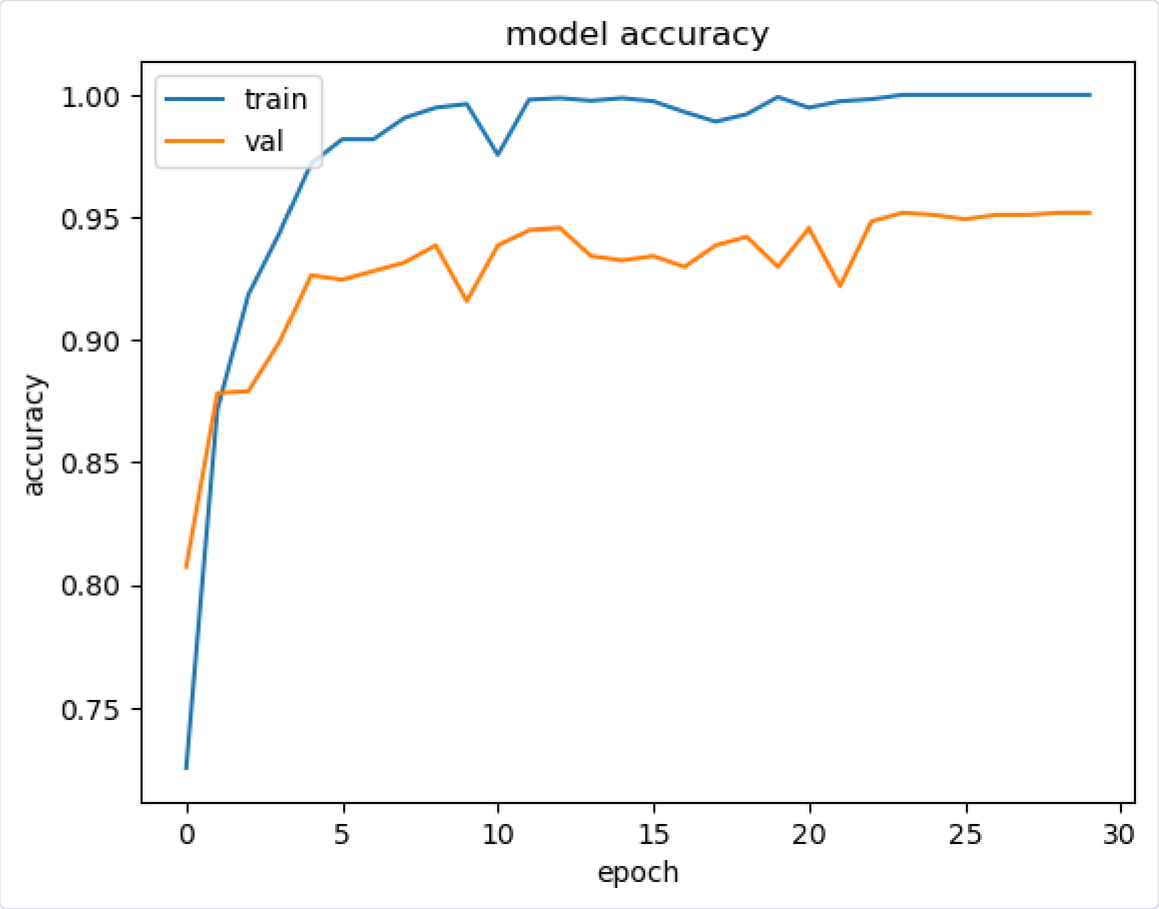
	glioma	meningioma	notumor	pituitary
glioma	280	20	0	0
meningioma	9	275	16	6
notumor	0	6	399	0
pituitary	0	3	1	296

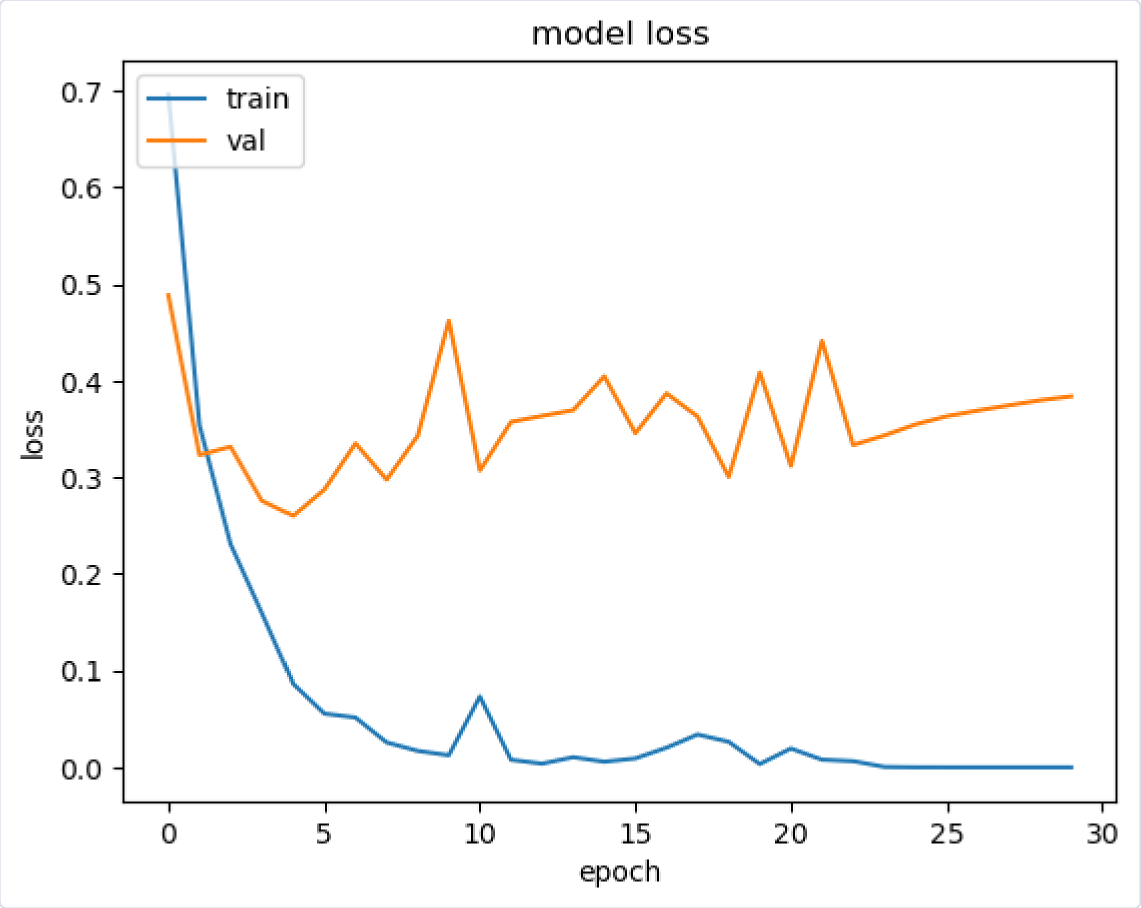
Classification Matrix

	Precision	Recall	F1-Score	Specificity	Support
glioma	0.96	0.93	0.94	0.98	300
meningioma	0.92	0.89	0.91	0.97	306
notumor	0.96	0.99	0.98	0.98	405
pituitary	0.98	0.99	0.99	0.99	300
Accuracy			**0.95**		1311
Macro Avg	0.95	0.95	0.95	0.98	1311
Weighted Avg	0.95	0.95	0.95	0.98	1311

loss and accuracy

At **30th** epoch
train_loss: 2.2074e-05 - train_accuracy: 1.0000 - val_loss: 0.3837 - val_accuracy: 0.9518





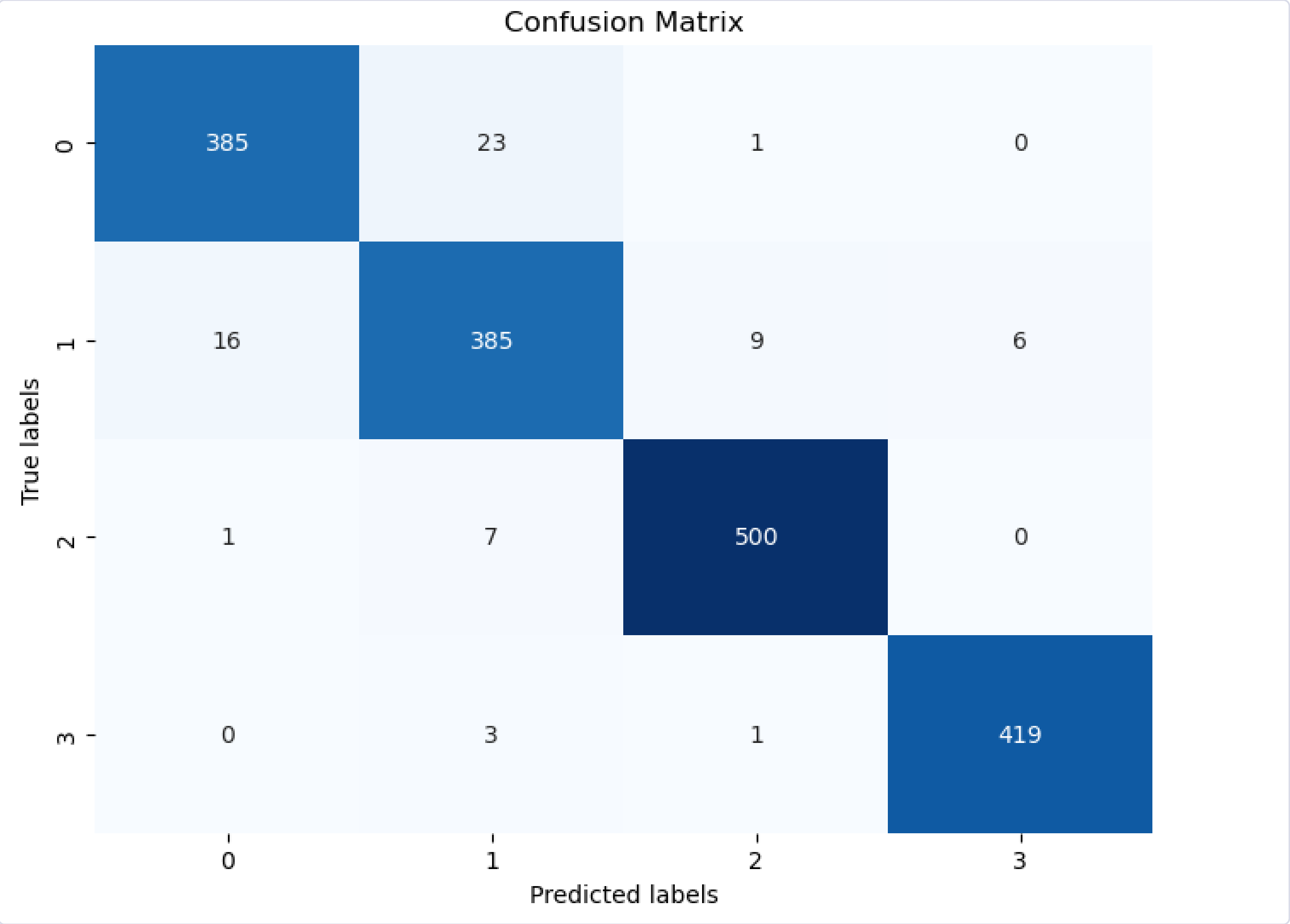
---

v2

- changed the learning rate to `0.005`

Performance metrics

Confusion Matrix



Classification Matrix

Class	Precision	Recall	F1-Score	specificity	Support
Glioma	0.96	0.94	0.95	0.981153	409
Meningioma	0.92	0.93	0.92	0.963129	416
Notumor	0.98	0.98	0.98	0.986301	508
Pituitary	0.99	0.99	0.99	0.993243	423
Accuracy			**0.96**		1756
Macro Avg	0.96	0.96	0.96	0.980957	1756
Weighted Avg	0.96	0.96	0.96	0.981303	1756

Different learning rate

when the learning rate is 0.01 the accuracy is quite bad i.e 0.29

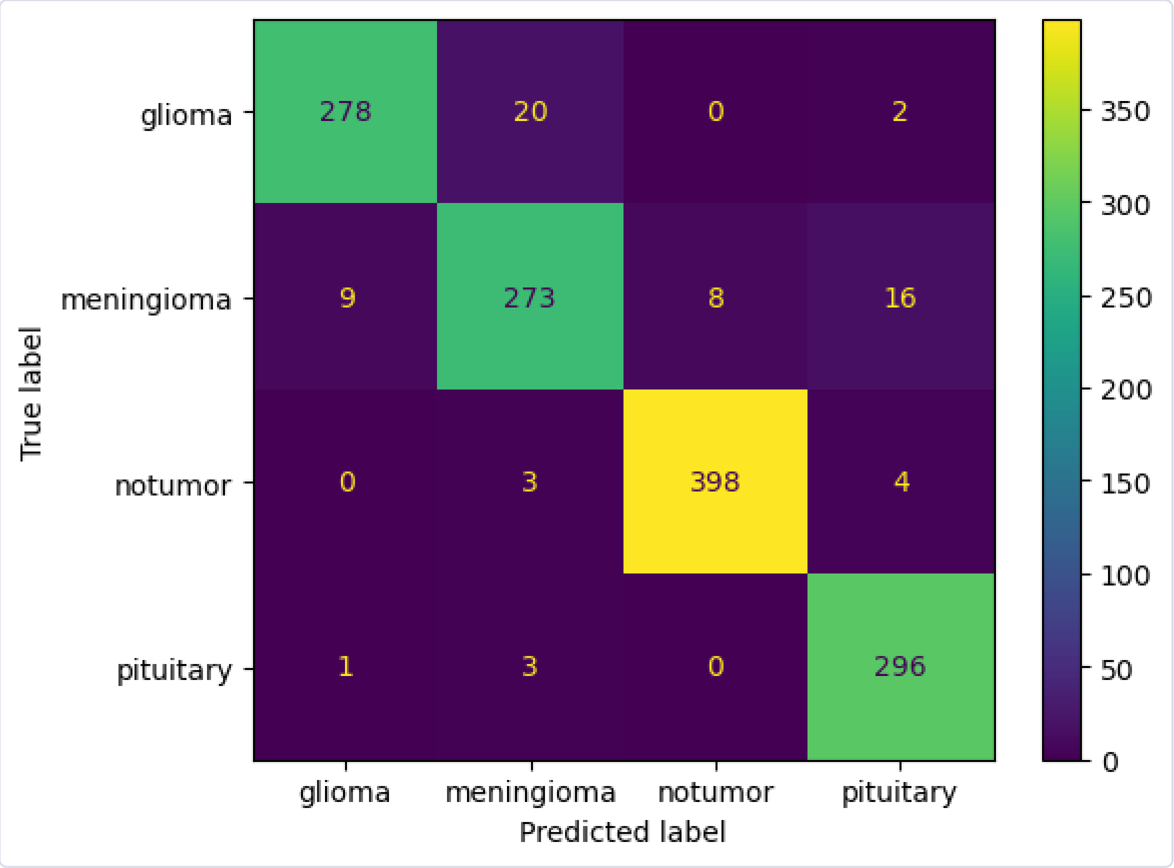
VGG19 Fine Tuned model

- Used pretrained VGG19 model with *imagenet* dataset
- VGG19 contains a total of 22 layers (including max_pool layers)
- of these 22, first 15 were frozen and remining 7 are trained
- model architecture

```
Model: "sequential_2"

_____________________________________________________________________________
 Layer (type)                Output Shape              Param #   Trainable
=============================================================================
 vgg19 (Functional)          (None, 4, 4, 512)         2002438   Y
                                                       4
|---------------------------------------------------------------------------|
| input_2 (InputLayer)       [(None, 128, 128, 3)]     0         N          |
|                                                                           |
| block1_conv1 (Conv2D)      (None, 128, 128, 64)      1792      N          |
|                                                                           |
| block1_conv2 (Conv2D)      (None, 128, 128, 64)      36928     N          |
|                                                                           |
| block1_pool (MaxPooling2D  (None, 64, 64, 64)        0          N         |
| )                                                                         |
|                                                                           |
| block2_conv1 (Conv2D)      (None, 64, 64, 128)       73856      N         |
|                                                                           |
| block2_conv2 (Conv2D)      (None, 64, 64, 128)       147584     N         |
|                                                                           |
| block2_pool (MaxPooling2D  (None, 32, 32, 128)       0          N         |
| )                                                                         |
|                                                                           |
| block3_conv1 (Conv2D)      (None, 32, 32, 256)       295168     N         |
|                                                                           |
| block3_conv2 (Conv2D)      (None, 32, 32, 256)       590080     N         |
|                                                                           |
| block3_conv3 (Conv2D)      (None, 32, 32, 256)       590080     N         |
|                                                                           |
| block3_conv4 (Conv2D)      (None, 32, 32, 256)       590080     N         |
|                                                                           |
| block3_pool (MaxPooling2D  (None, 16, 16, 256)       0          N         |
| )                                                                         |
|                                                                           |
| block4_conv1 (Conv2D)      (None, 16, 16, 512)       1180160    N         |
|                                                                           |
| block4_conv2 (Conv2D)      (None, 16, 16, 512)       2359808    N         |
|                                                                           |
| block4_conv3 (Conv2D)      (None, 16, 16, 512)       2359808    N         |
|                                                                           |
| block4_conv4 (Conv2D)      (None, 16, 16, 512)       2359808    Y         |
|                                                                           |
| block4_pool (MaxPooling2D  (None, 8, 8, 512)         0          Y         |
| )                                                                         |
|                                                                           |
| block5_conv1 (Conv2D)      (None, 8, 8, 512)         2359808    Y         |
|                                                                           |
| block5_conv2 (Conv2D)      (None, 8, 8, 512)         2359808    Y         |
|                                                                           |
| block5_conv3 (Conv2D)      (None, 8, 8, 512)         2359808    Y         |
|                                                                           |
| block5_conv4 (Conv2D)      (None, 8, 8, 512)         2359808    Y         |
|                                                                           |
| block5_pool (MaxPooling2D  (None, 4, 4, 512)         0          Y         |
| )                                                                         |
-----------------------------------------------------------------------------
 flatten_1 (Flatten)         (None, 8192)              0          Y

 dense_4 (Dense)             (None, 128)               1048704    Y

 dense_5 (Dense)             (None, 4)                 516        Y

=============================================================================
Total params: 21073604 (80.39 MB)
Trainable params: 12848260 (49.01 MB)
Non-trainable params: 8225344 (31.38 MB)

-----------------------------------------------------------------------------
```

Performance metrics

Confusion matrix



Classification report

	precision	recall	f1-score	specificity	support
glioma	0.97	0.93	0.95	0.99	300
meningioma	0.91	0.89	0.90	0.97	306
no tumor	0.98	0.98	0.98	0.99	405
pituitary	0.93	0.99	0.96	0.97	300
accuracy			**0.95**		1311
macro avg	0.95	0.95	0.95	0.98	1311
weighted avg	0.95	0.95	0.95	0.98	1311

loss and accuracy of train and validation

At **15th** epoch
train_loss: 0.1136 - train_accuracy: 0.9676 - val_loss: 0.2138 - val_accuracy: 0.9378

Resnet 50

Resnet50 v1

- Built from scratch
- model architecture

```Python
def resNet50(input_shape=(128,128,3)):
    X_input = tf.keras.layers.Input(shape=input_shape)
    X = tf.keras.layers.ZeroPadding2D((3,3))(X_input)

    X = tf.keras.layers.Conv2D(filters=64, kernel_size=7, strides=2)(X)
    X = tf.keras.layers.BatchNormalization(axis=3)(X)
    X = tf.keras.layers.ReLU()(X)
    X = tf.keras.layers.MaxPool2D(pool_size=(3,3), strides=2)(X)

    X = conv_block(X, filters=[64, 64, 256], s=1)
    X = identity_block(X, filters=[64, 64, 256])
    X = identity_block(X, filters=[64, 64, 256])

    X = conv_block(X, filters=[128, 128, 512], s=2)
    X = identity_block(X, filters=[128, 128, 512])
    X = identity_block(X, filters=[128, 128, 512])
    X = identity_block(X, filters=[128, 128, 512])

    X = conv_block(X, filters=[256, 256, 1024], s=2)
    X = identity_block(X, filters=[256, 256, 1024])
    X = identity_block(X, filters=[256, 256, 1024])
    X = identity_block(X, filters=[256, 256, 1024])
    X = identity_block(X, filters=[256, 256, 1024])
    X = identity_block(X, filters=[256, 256, 1024])

    X = conv_block(X, filters=[512, 512, 2048], s=2)
    X = identity_block(X, filters=[512, 512, 2048])
    X = identity_block(X, filters=[512, 512, 2048])

    X = tf.keras.layers.AveragePooling2D((2,2), padding='same')(X)

    X = tf.keras.layers.Flatten()(X)
    X = tf.keras.layers.Dense(4, activation='softmax')(X)

    model = tf.keras.Model(inputs=X_input, outputs=X, name='ResNet50')
    return model
```

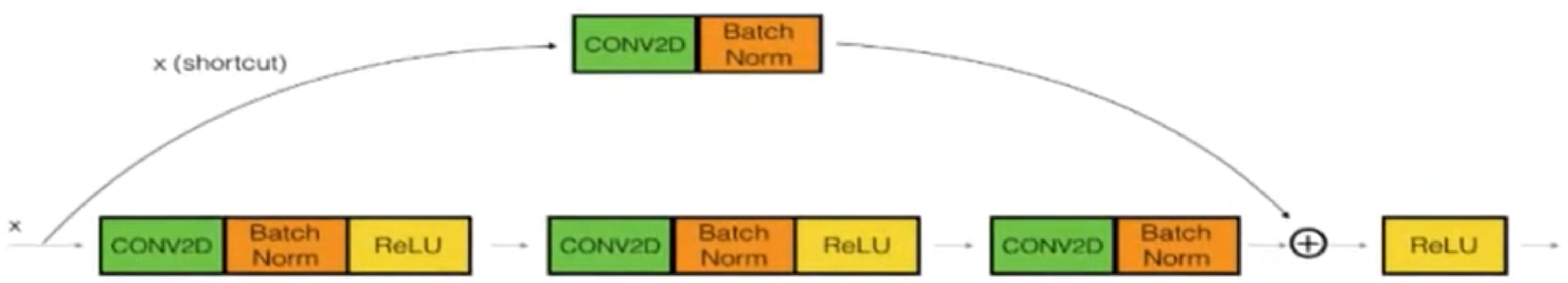


Figure 4 : Convolutional block

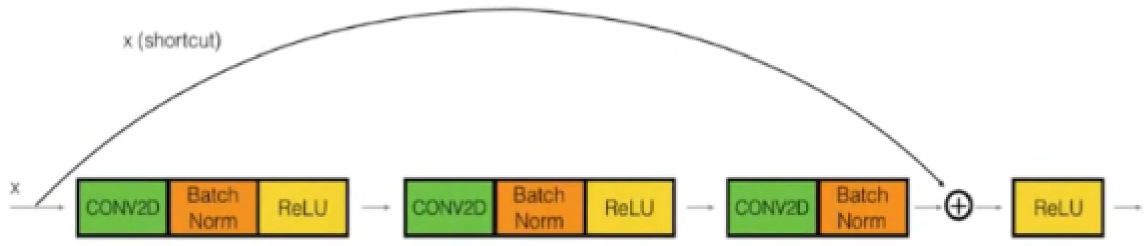
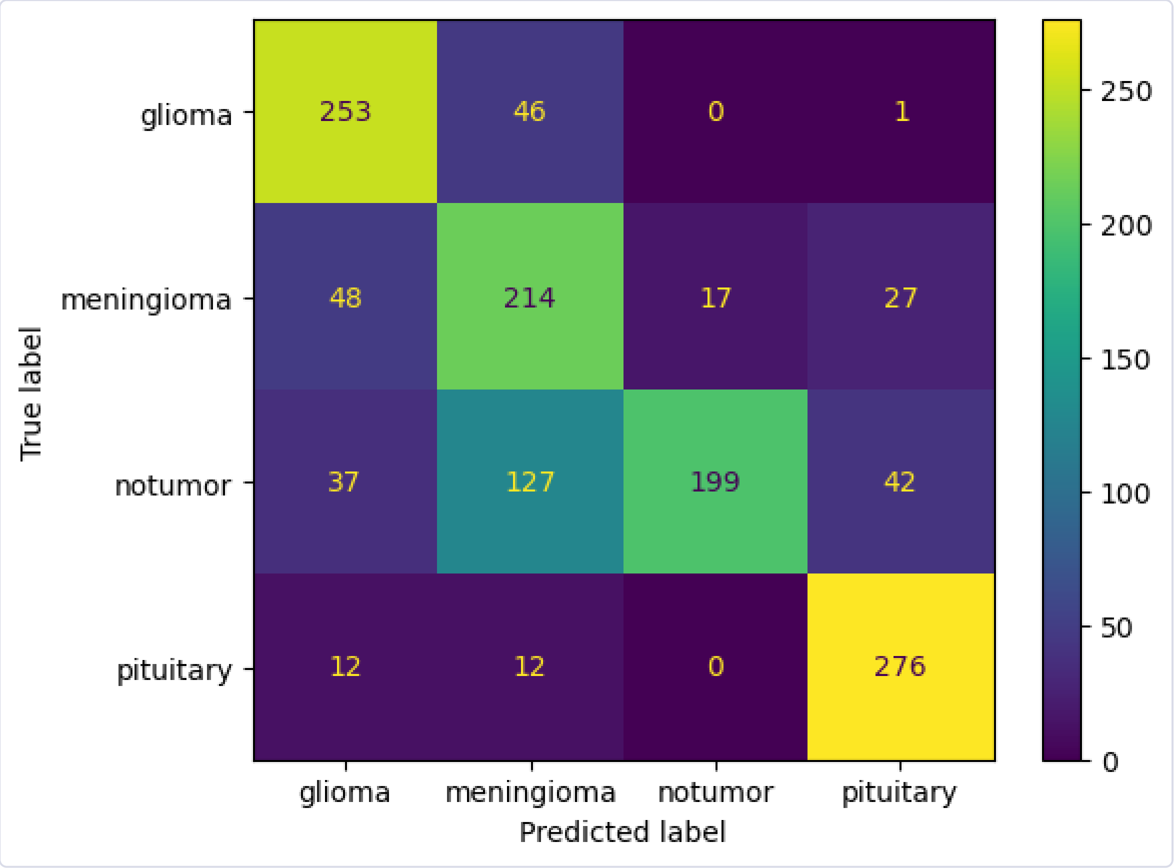


Figure 4 : **Identity block.** Skip connection "skips over" 3 layers.

Performance Metrics

Confusion Matrix

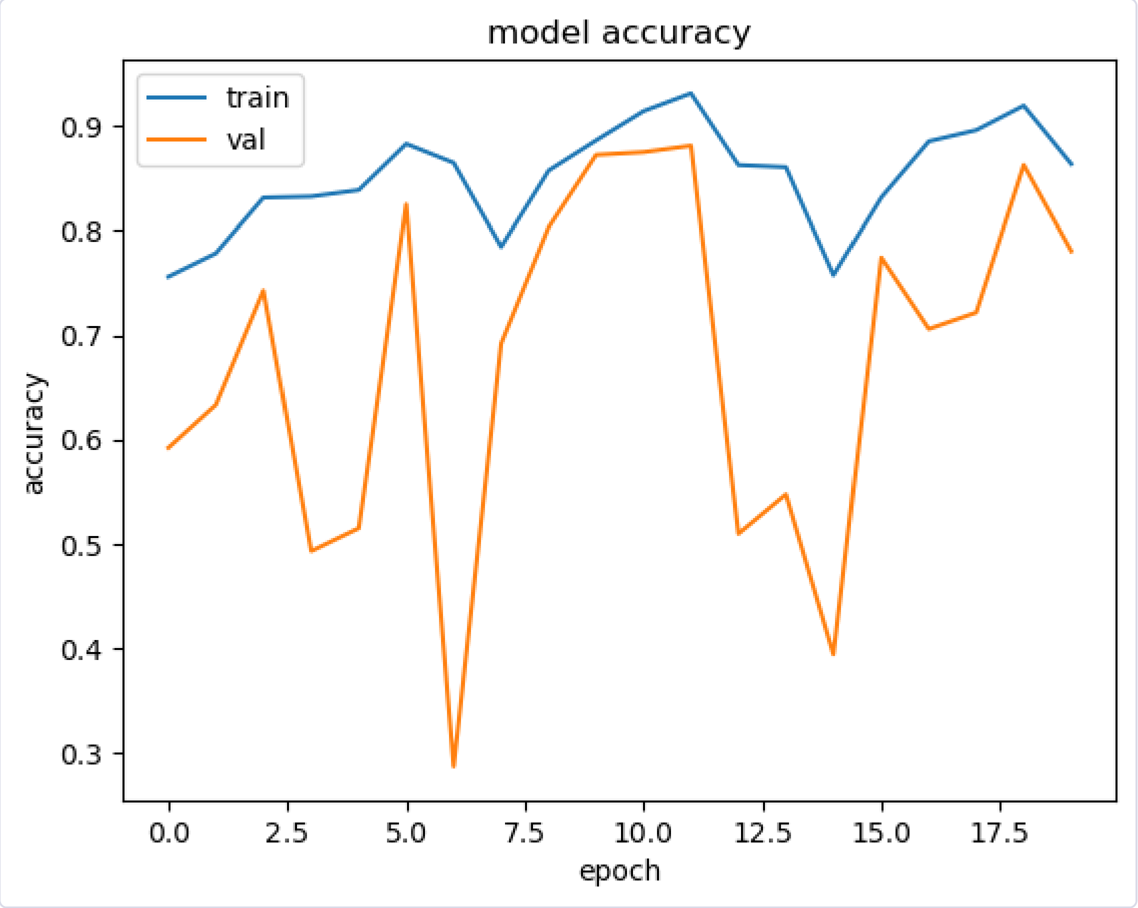


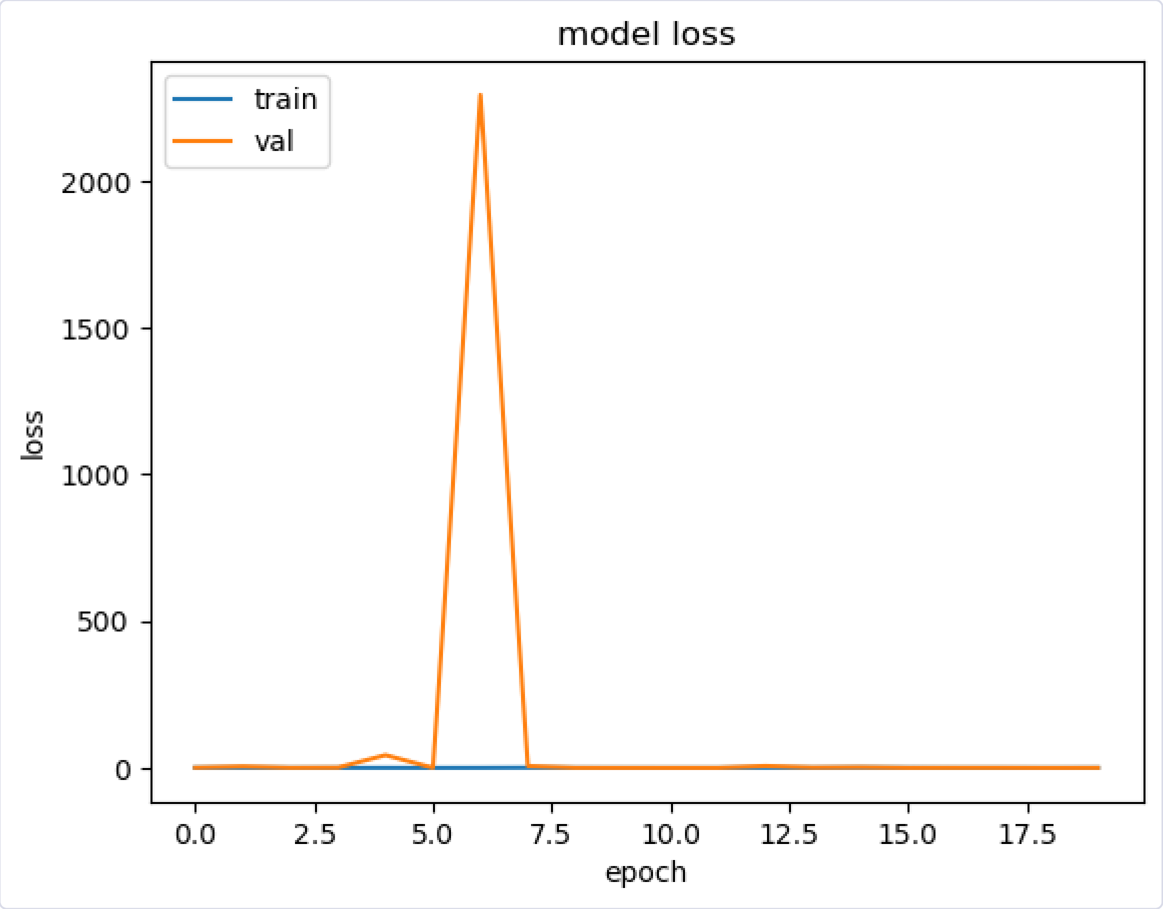
Classification report

	Precision	Recall	F1-Score	Specificity	Support
glioma	0.72	0.84	0.78	0.90	300
meningioma	0.54	0.70	0.61	0.81	306
notumor	0.92	0.49	0.64	0.98	405
pituitary	0.80	0.92	0.85	0.93	300
Accuracy			**0.72**		1311
Macro Avg	0.74	0.74	0.72	0.90	1311
Weighted Avg	0.76	0.72	0.71	0.91	1311

Loss and accuracy

At **20th** epoch
train_loss: 0.4255 - train_accuracy: 0.8635 - val_loss: 0.5622 - val_accuracy: 0.7802



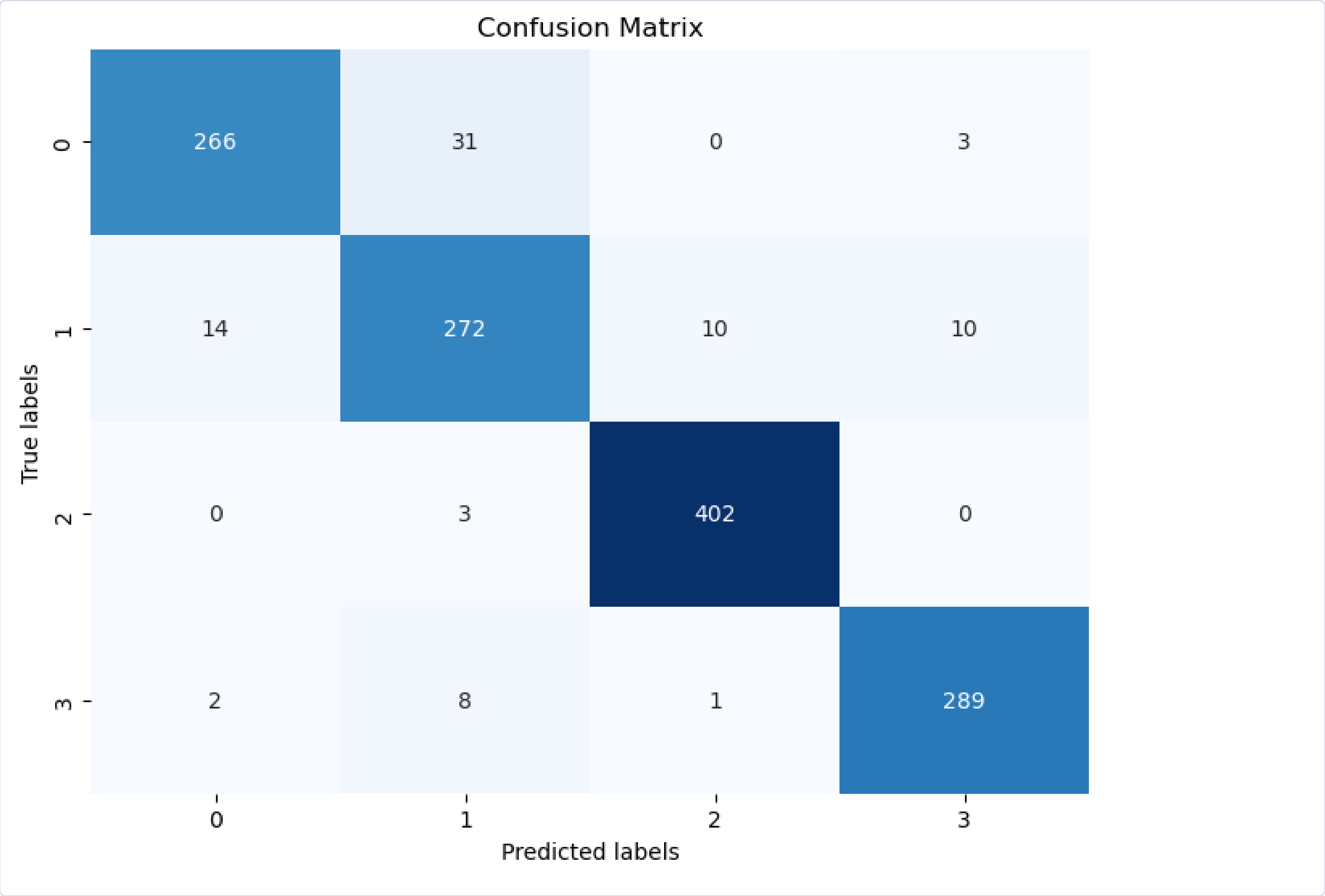


Resnet50 v2

- Used pretrained Resnet50 with *imagenet* dataset
- All the layers are frozen, except for the fully connected layer

Performance metrics

confusion matrix



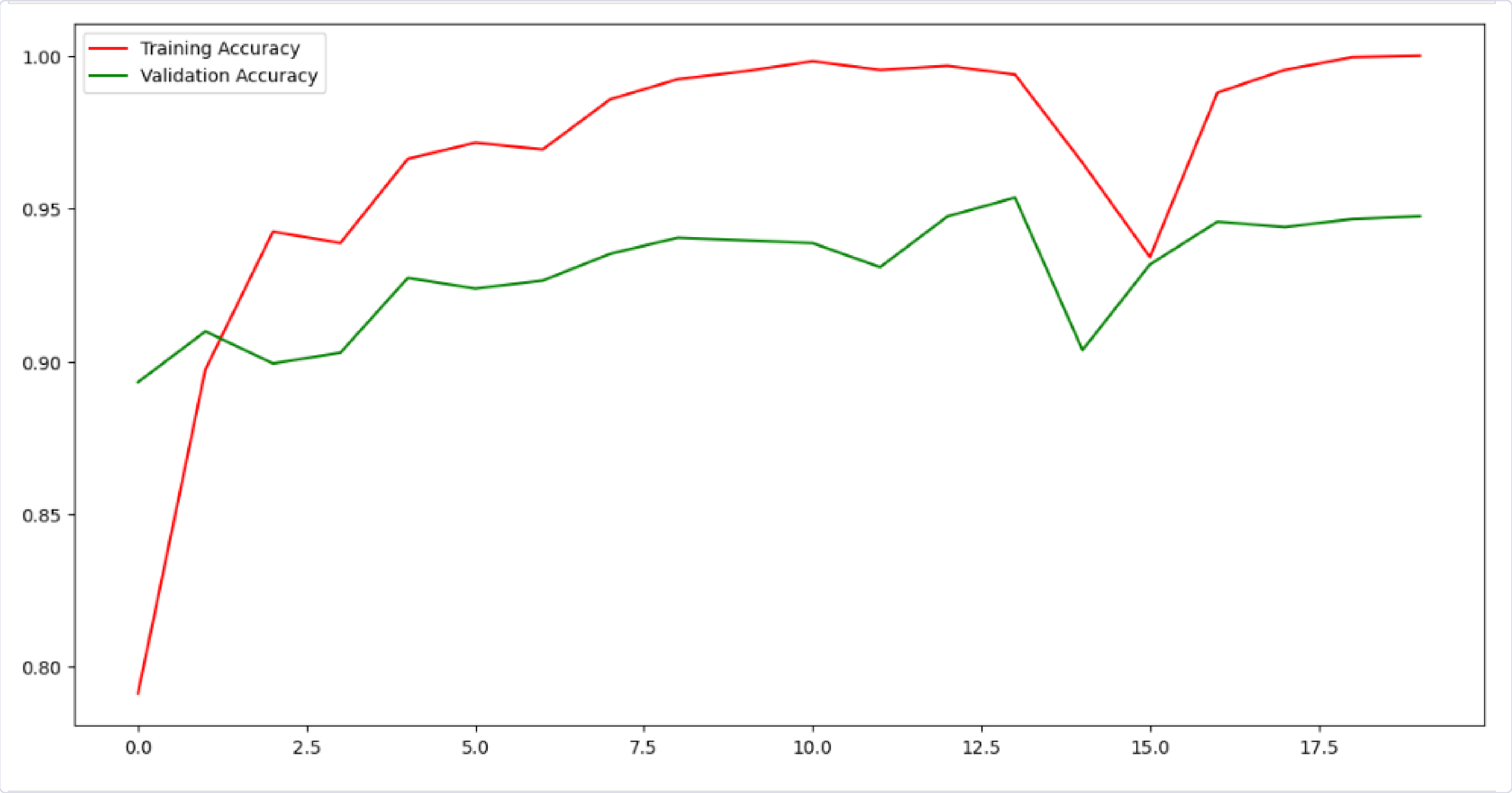
Classification Report

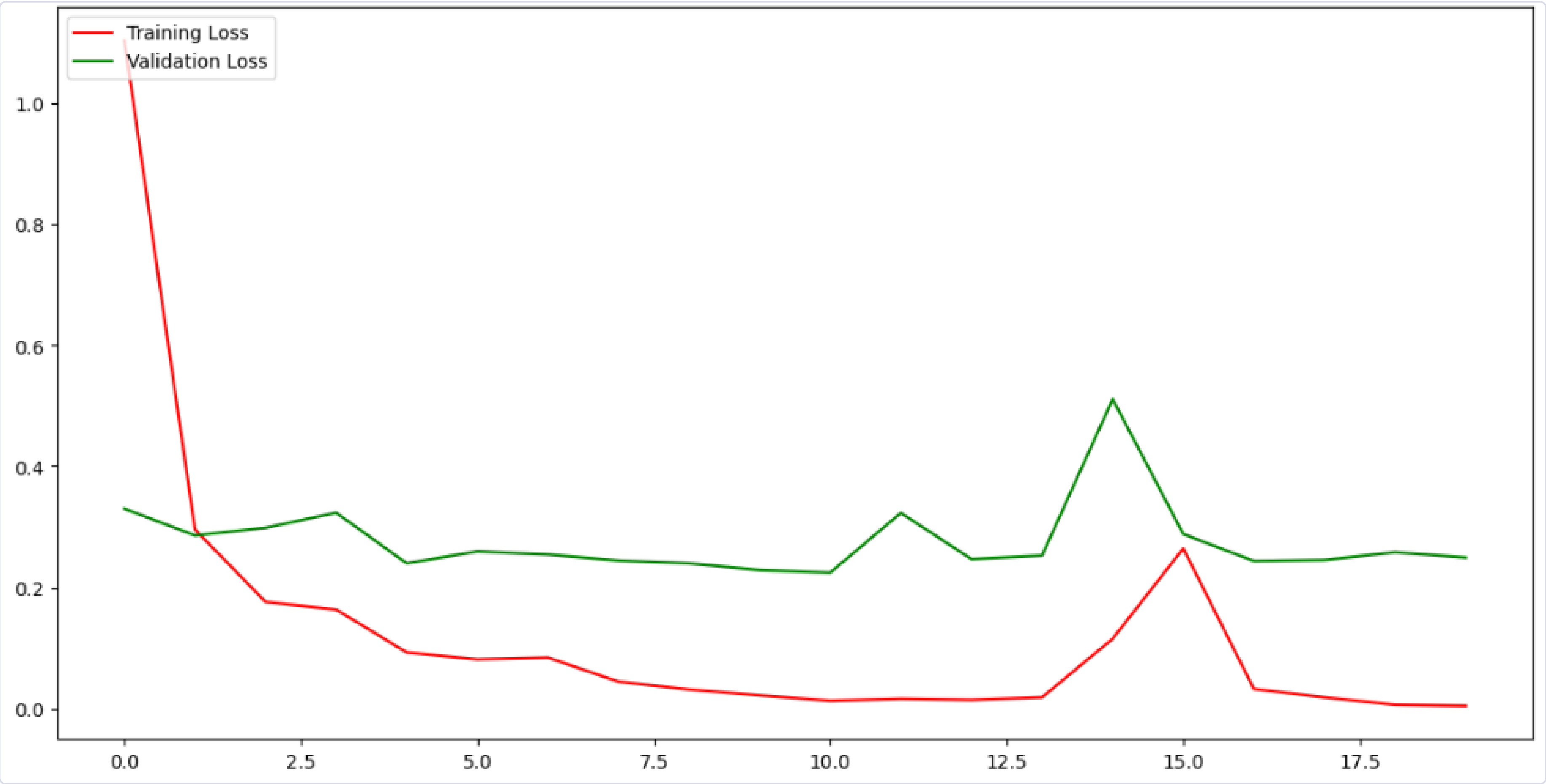
	Precision	**recall**	**f1-score**	**specificity**	**support**
glioma	0.94	0.89	0.91	0.98	300
meningioma	0.87	0.89	0.88	0.95	306
notumor	0.97	0.99	0.98	0.98	405
pituitary	0.96	0.96	0.96	0.98	300

	Precision	recall	f1-score	specificity	support
accuracy			0.94		1311
macro avg	0.93	0.93	0.93	0.97	1311
weighted avg	0.94	0.94	0.94	0.97	1311

Loss and accuracy

At **20th** epoch
train_loss: 0.0043 - train_accuracy: 1.0000 val_loss: 0.2492 - val_accuracy: 0.9475

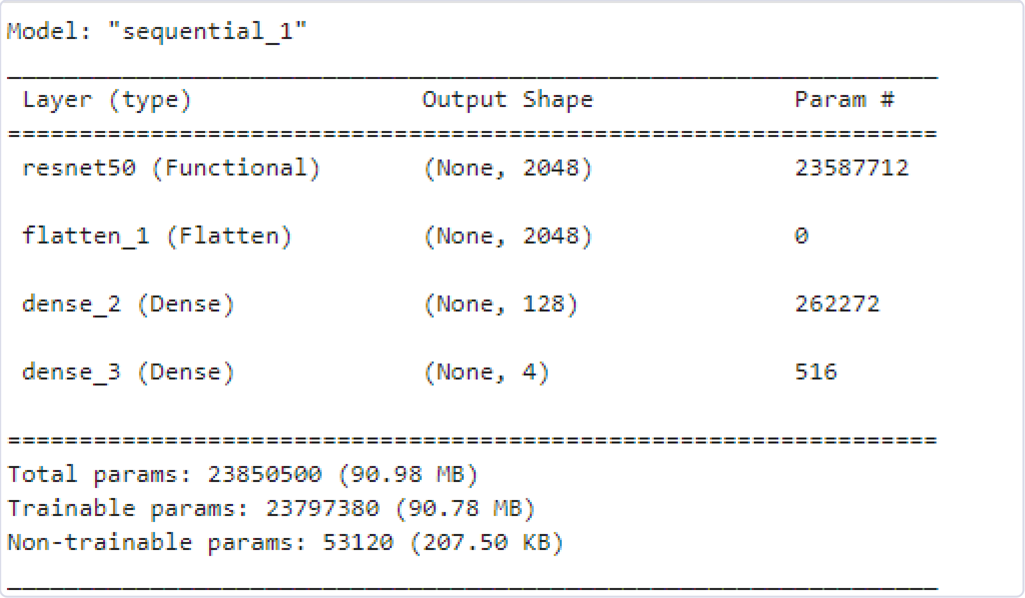




Resnet50 v3

- Used pretrained Resnet50 with *imagenet* dataset
- All layers are trainable

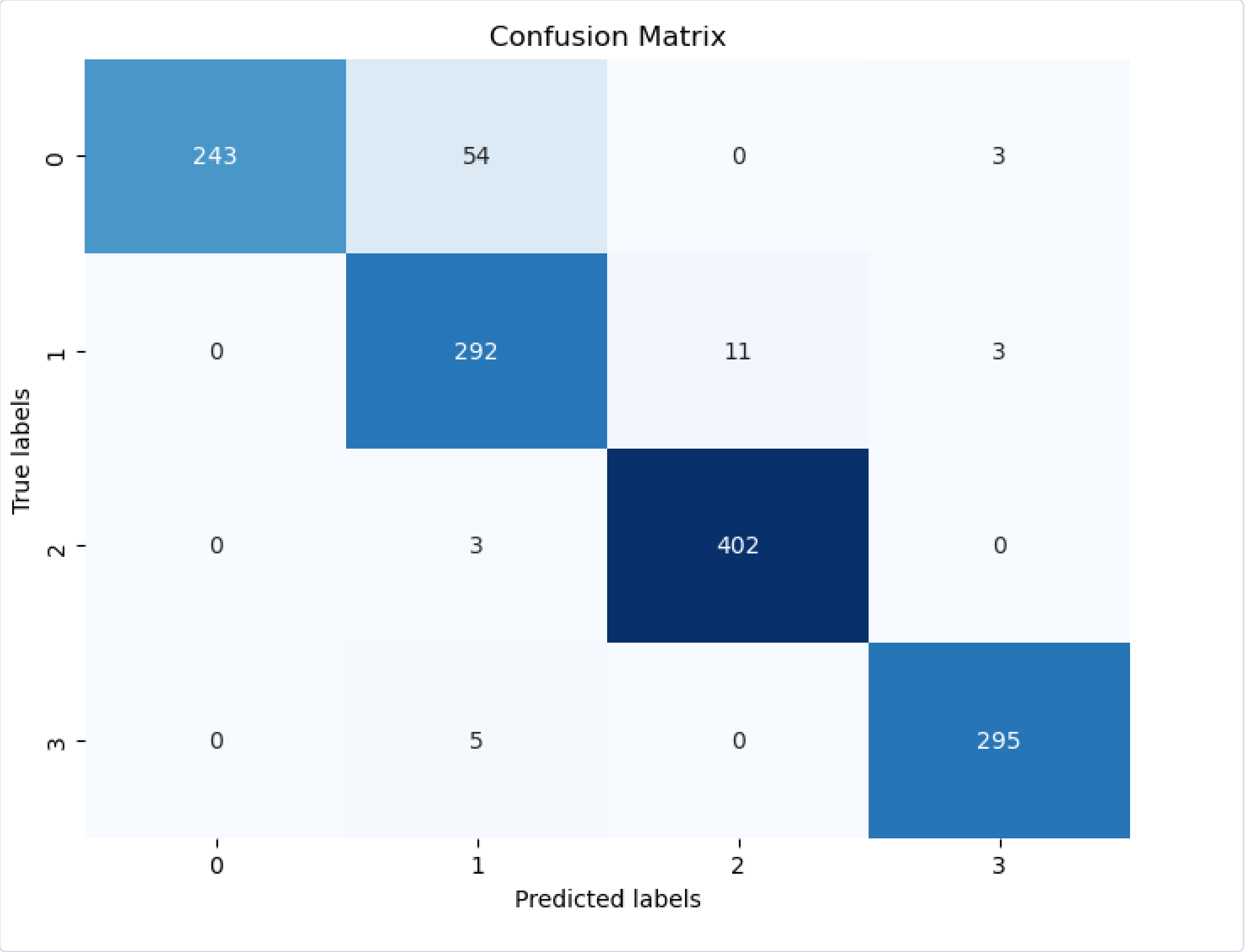
- model summary



```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 2048)              23587712

 flatten_1 (Flatten)         (None, 2048)              0

 dense_2 (Dense)             (None, 128)               262272

 dense_3 (Dense)             (None, 4)                 516

=================================================================
Total params: 23850500 (90.98 MB)
Trainable params: 23797380 (90.78 MB)
Non-trainable params: 53120 (207.50 KB)
_________________________________________________________________
```

Performance Metrics

confusion Matrix



Classification report

	Precision	recall	f1-score	specificity	support
glioma	1.00	0.81	0.90	1	300
meningioma	0.82	0.95	0.88	0.938308	306
notumor	0.97	0.99	0.98	0.987859	405
pituitary	0.98	0.98	0.98	0.994065	300
accuracy			**0.94**		1311
macro avg	0.94	0.94	0.94	0.980058	1311
weighted avg	0.95	0.94	0.94	0.980492	1311

Resnet101

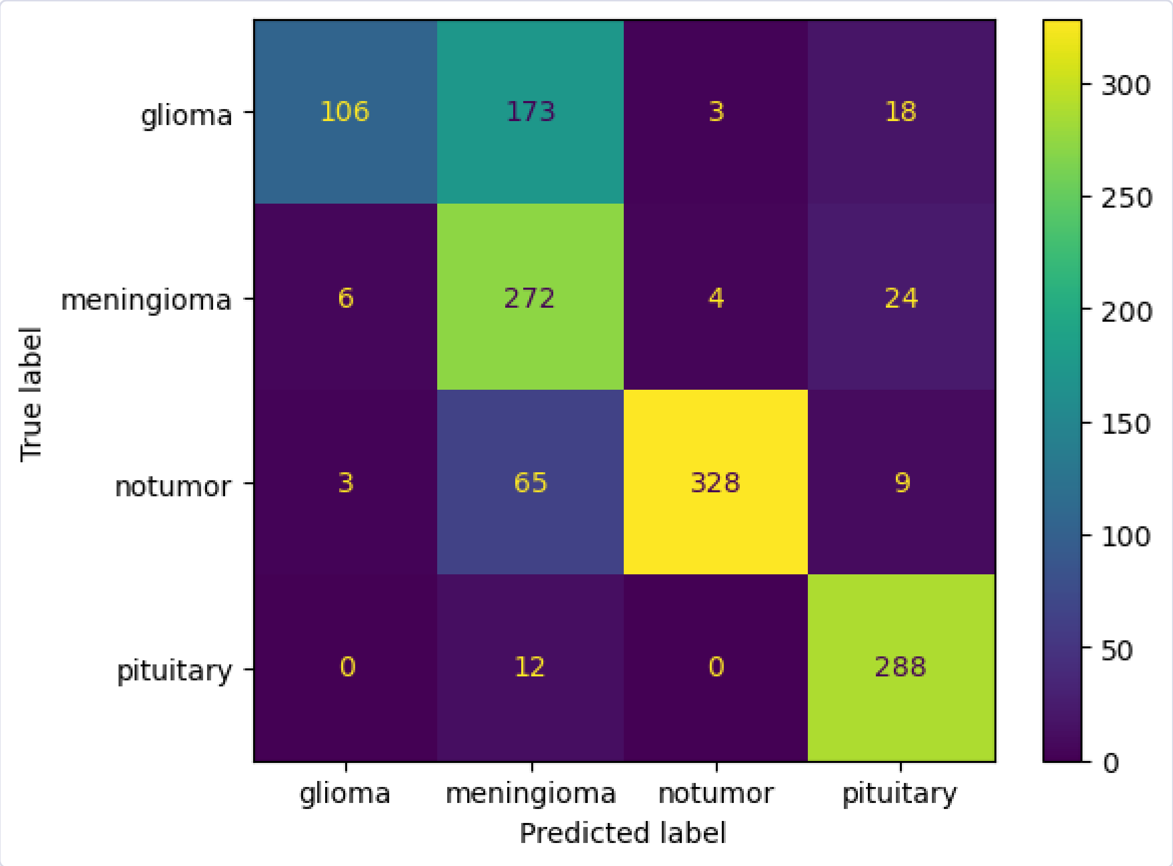
Resnet101 v1

- Used pretrained Resnet101 with *imagenet* dataset

- Of 345 layers froze all except last 10 layers
- Added a **dropout** layer after **flatten**

Performance Metrics

Confusion matrix

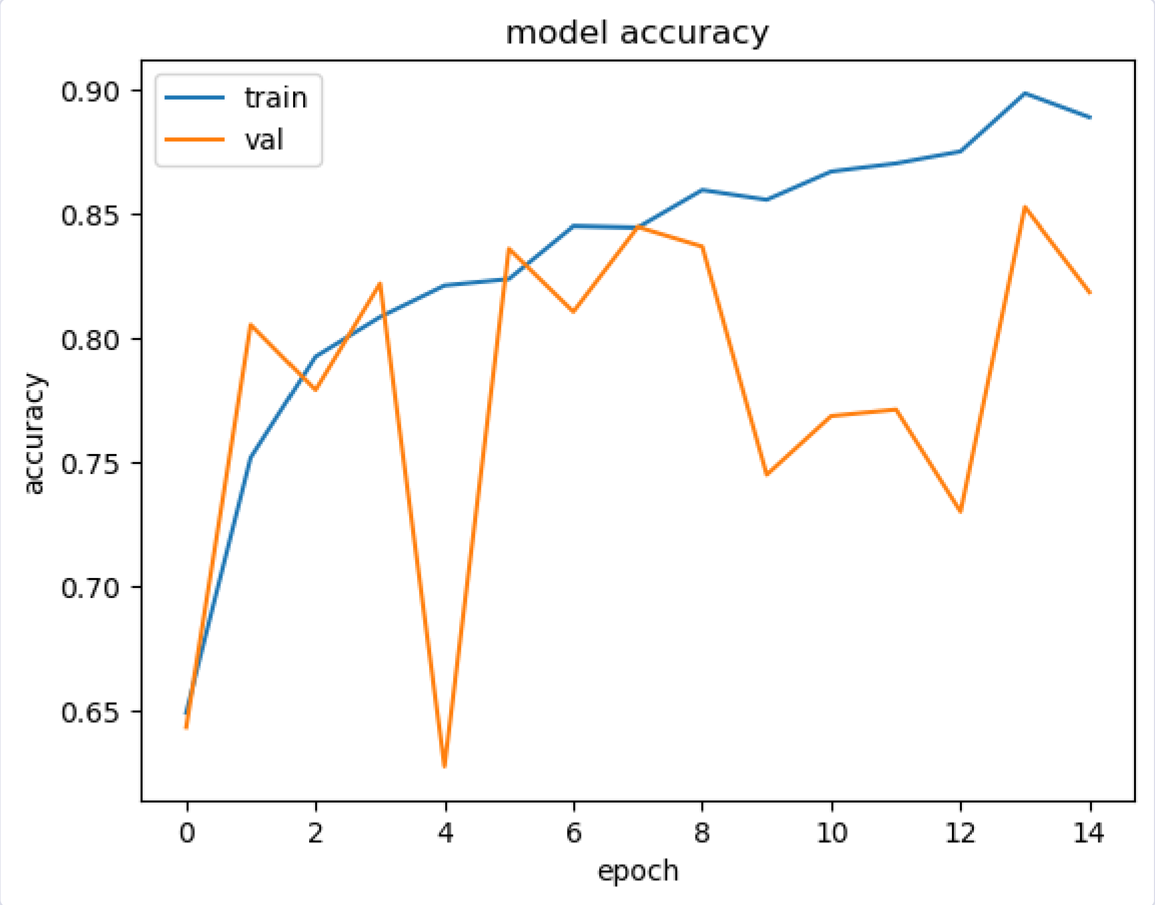


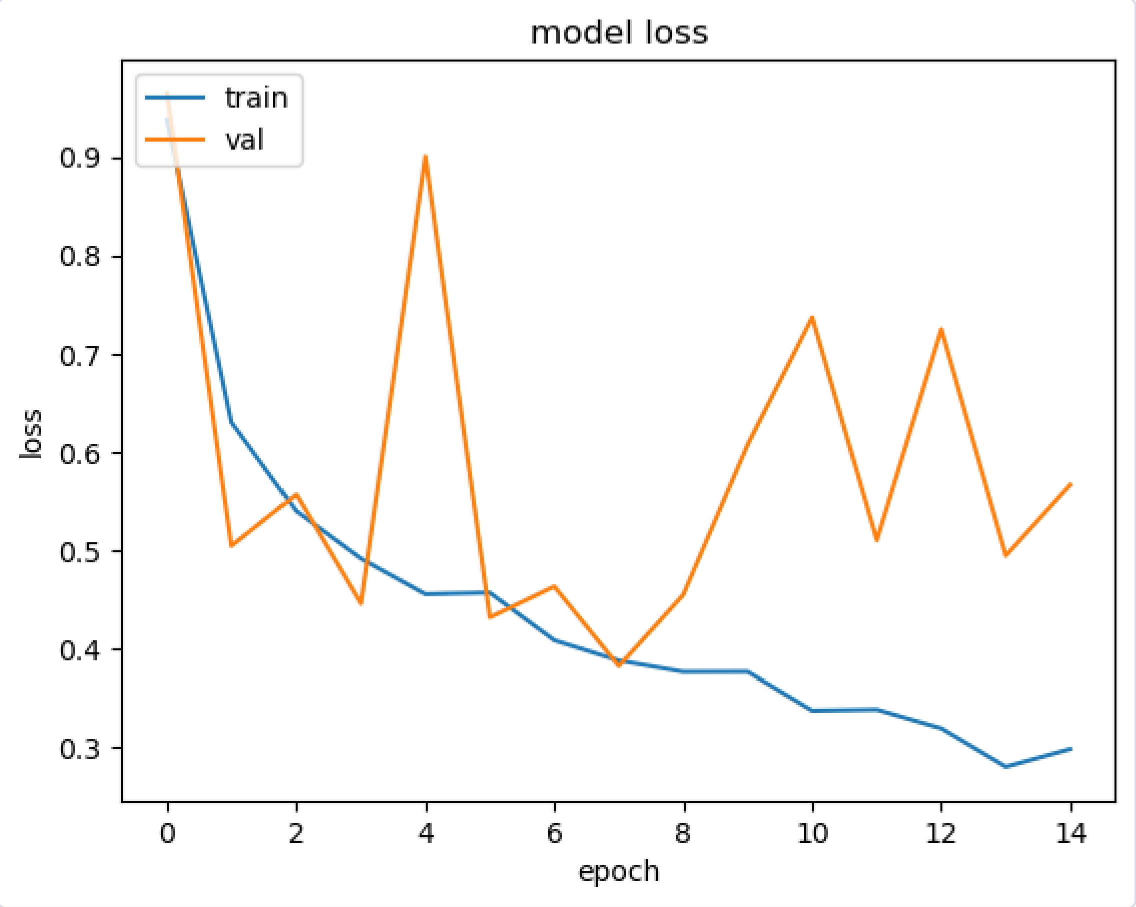
Classification report

	Precision	Recall	F1-Score	Specificity	Support
glioma	0.92	0.35	0.51	0.99	300
meningioma	0.52	0.89	0.66	0.75	306
notumor	0.98	0.81	0.89	0.99	405
pituitary	0.85	0.96	0.90	0.94	300
Accuracy			**0.76**		1311
Macro Avg	0.82	0.75	0.74	0.92	1311
Weighted Avg	0.83	0.76	0.75	0.92	1311

Loss and accuracy

At **15th** epoch
train_loss: 0.2979 - train_accuracy: 0.8891 - val_loss: 0.5666 - val_accuracy: 0.8187



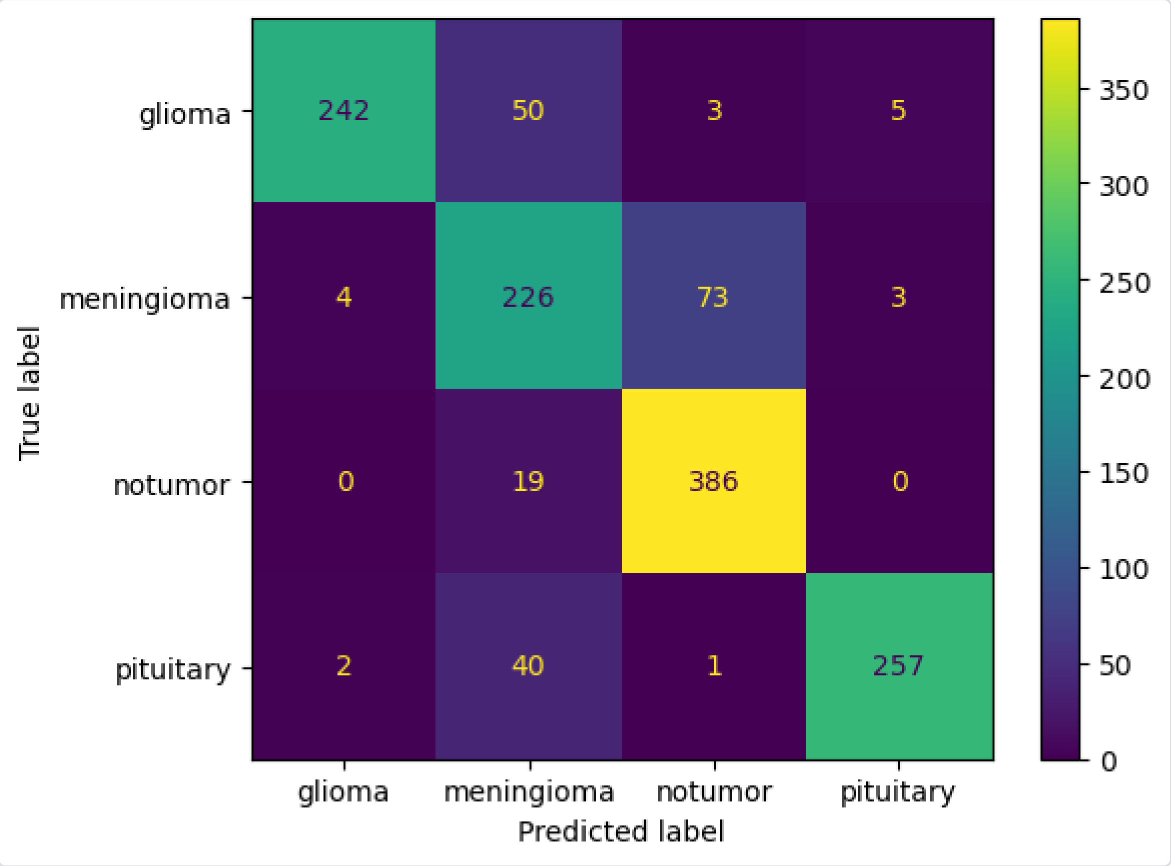


Resnet101 v2

- Used pretrained Resnet101 with *imagenet* dataset
- All the layers are trainable

Performance Metrics

Confusion Matrix

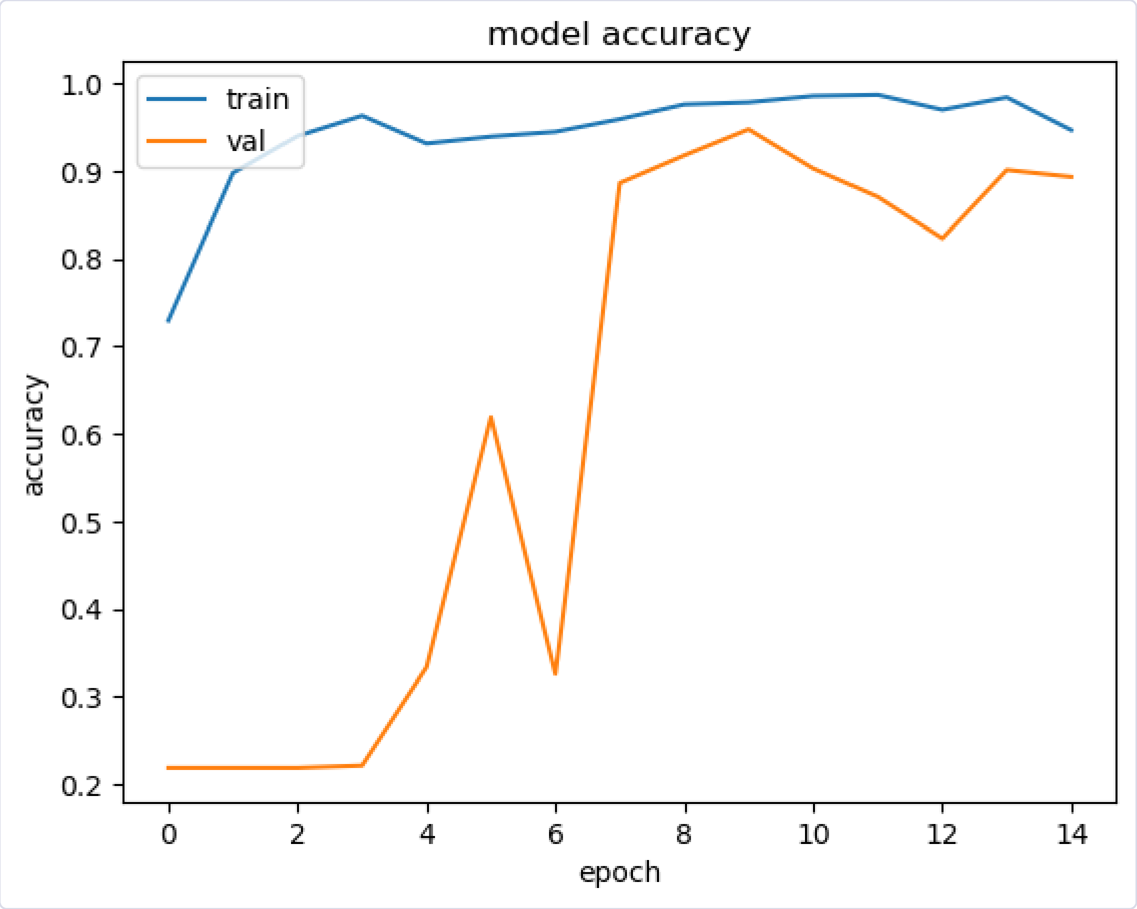


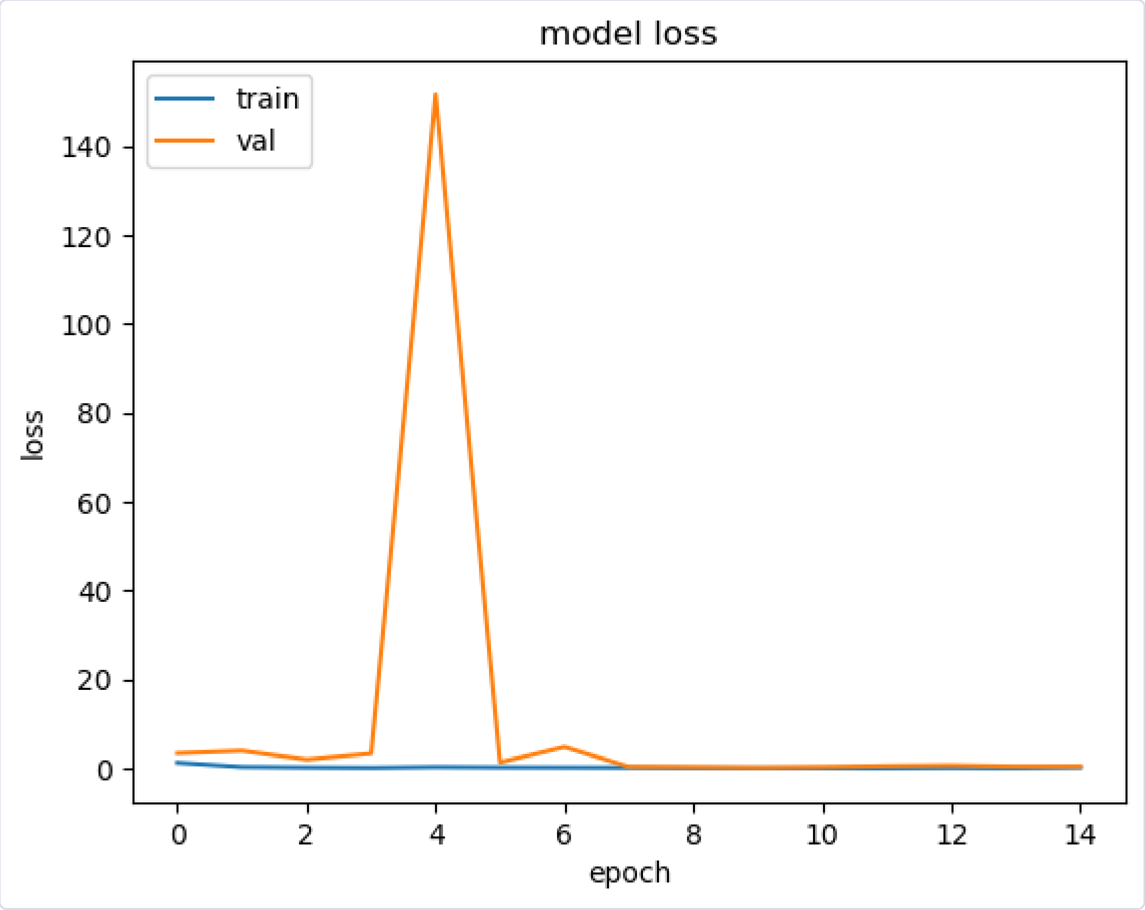
Classification report

	Precision	Recall	F1-Score	Specificity	Support
glioma	0.98	0.81	0.88	0.99	300
meningioma	0.67	0.74	0.71	0.89	306
notumor	0.83	0.95	0.89	0.91	405
pituitary	0.97	0.86	0.91	0.99	300
Accuracy			**0.85**		1311
Macro Avg	0.86	0.84	0.85	0.94	1311
Weighted Avg	0.86	0.85	0.85	0.94	1311

loss and accuracy

at **15th** epoch
train_loss: 0.1957 - train_accuracy: 0.9466 - val_loss: 0.4219 - val_accuracy: 0.8932





Conclusion

In terms of accuracy **model1_v2** (custom model with 0.005 learning rate) is the best with 96% accuracy on test set

Model	**Accuracy**
Model1_v2	0.96
Model1_v1	0.95
VGG19_finetune *transfer learning*	0.95
Resnet50 (all frozen) *transfer learning*	0.94
Resnet50 (all trainable) *transfer learning*	0.94
Resnet101 (all trainable) *transfer learning*	0.85
Resnet101_finetune *transfer learning*	0.76
Resnet50	0.72