BRAIN TUMOR DETECTION

Medical Image Processing Using Deep Learning, 2024

OUR TEAM

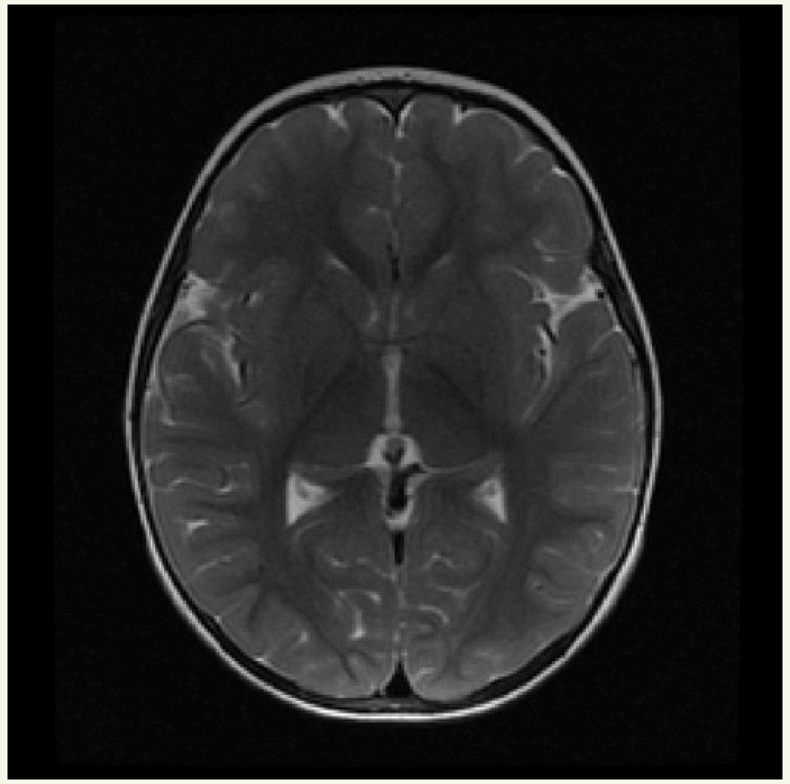
Eden Boaron

Natan Izhak Poor

Lior Mamos

INTRODUCTION

- Brain tumor classification & Grad-CAM to identify areas of interest.
- **Goal:** Help doctors classify between different kinds of brain tumors
- Using brain MRI images dataset from Kaggle
- Input: MRI scan of a brain
- Output: The kind of brain tumor (or no tumor, hopefully)

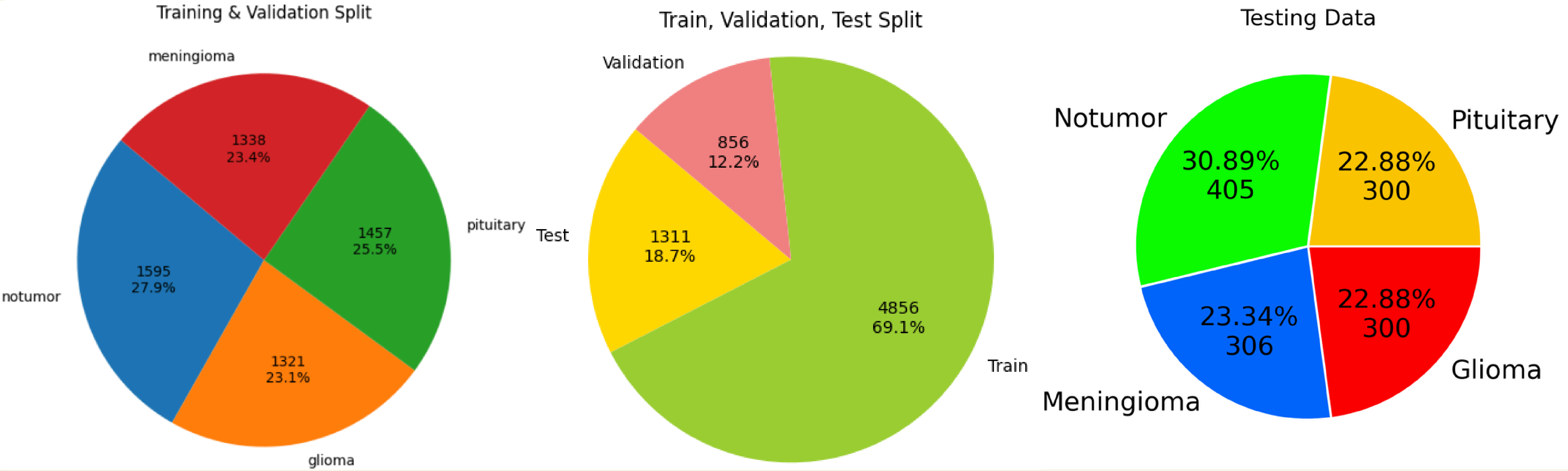


FACT

Spotting brain tumors early is key for best treatment and better patient outcomes.

DATA-SET

- Number of images: 7023
- 4856 are used for training, 855 for validation, and 1310 for testing (70-12-18)
- Number of classes: 4 (glioma - meningioma - pituitary and no tumor)
- We’ve uploaded the Images to the Cloud

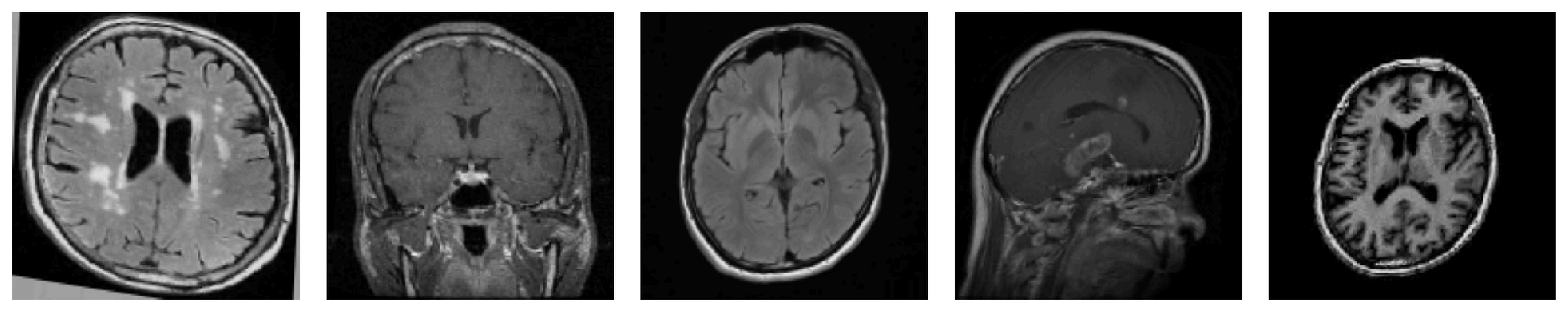


METADATA

- Our dataset is a combination of 3 other datasets
- All images were converted to JPG format by the provider and use a range of [0-255] for pixels with 3 channels
- The images contain MRI imaging of three different types: T1, T2 and FLAIR
- Images include pateint ID which can be used for patient tracking and seeing changes over time
- All images are tagged with their type of tumor

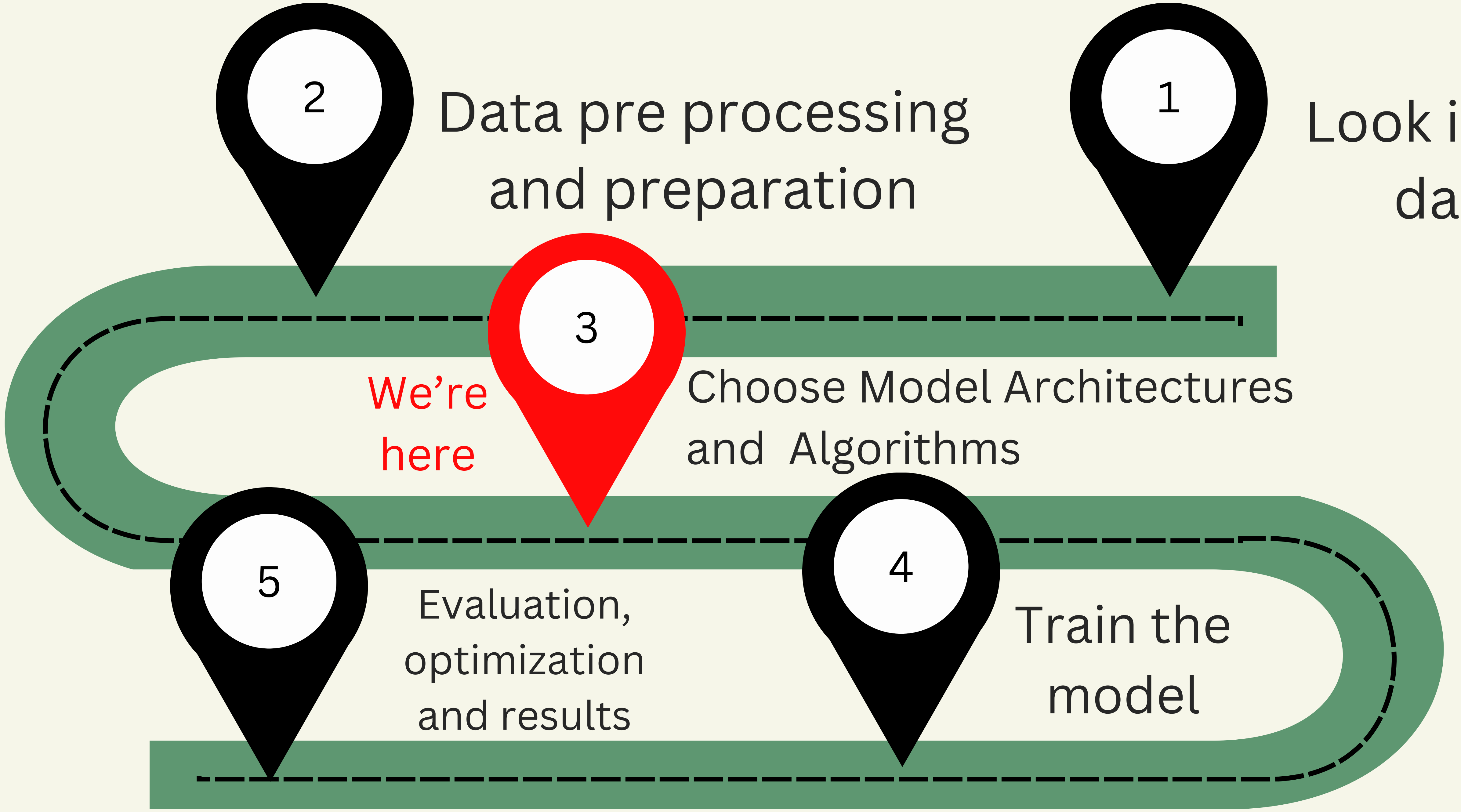
PRE-PROCESSING

- Resize the images (or use padding) to have the same resolution (150x150)
- Normalizing the images to use pixel values of [0-1]
- We use augmentation to increase the diversity of the training, validation datasets, testing dataset isn’t augmented



Example - Some images after augmentation

SOLUTION APPROACH



1 Look into the datset

2 Data pre processing and preparation

3 Choose Model Architectures and Algorithms

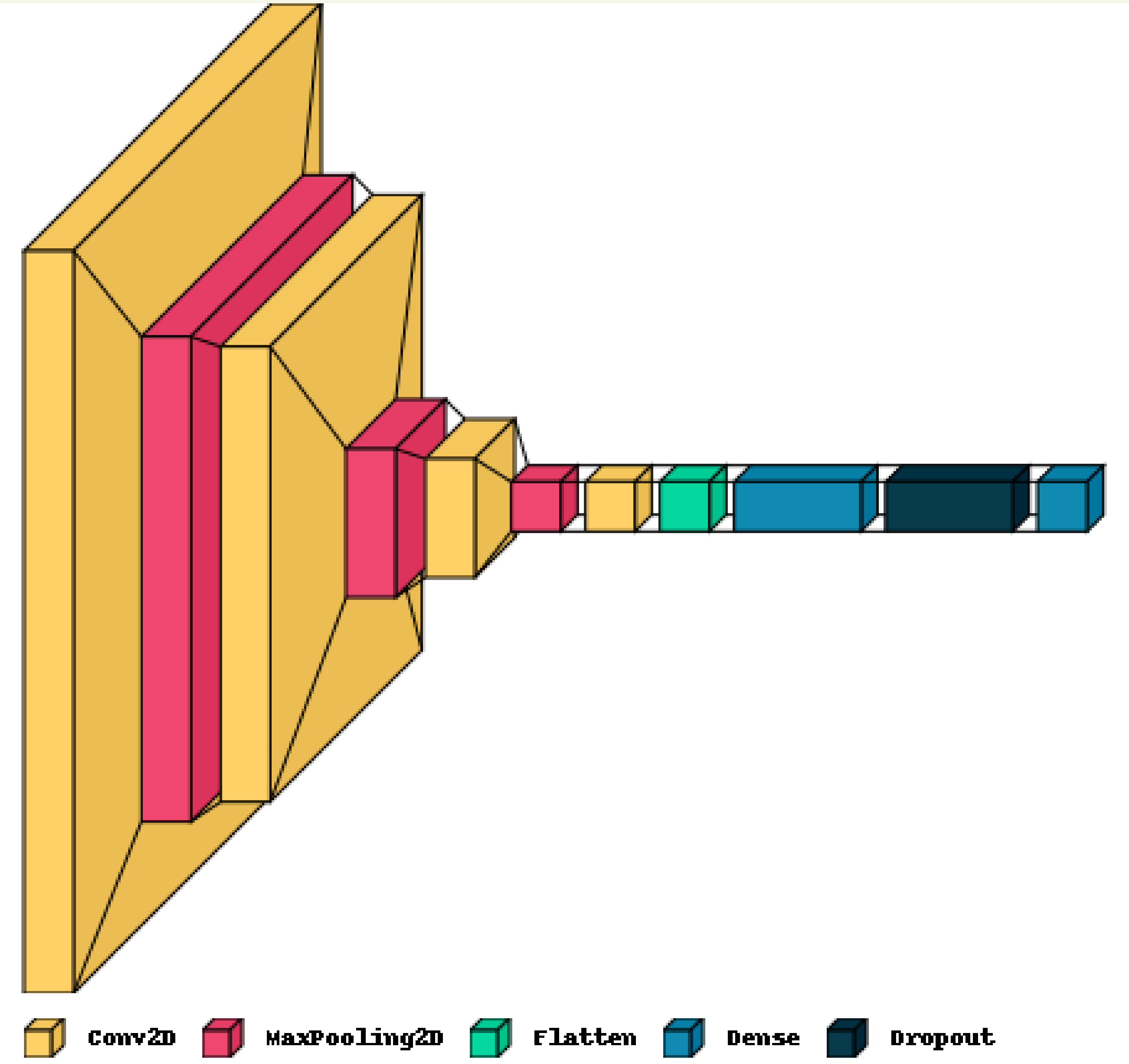
We’re here

4 Train the model

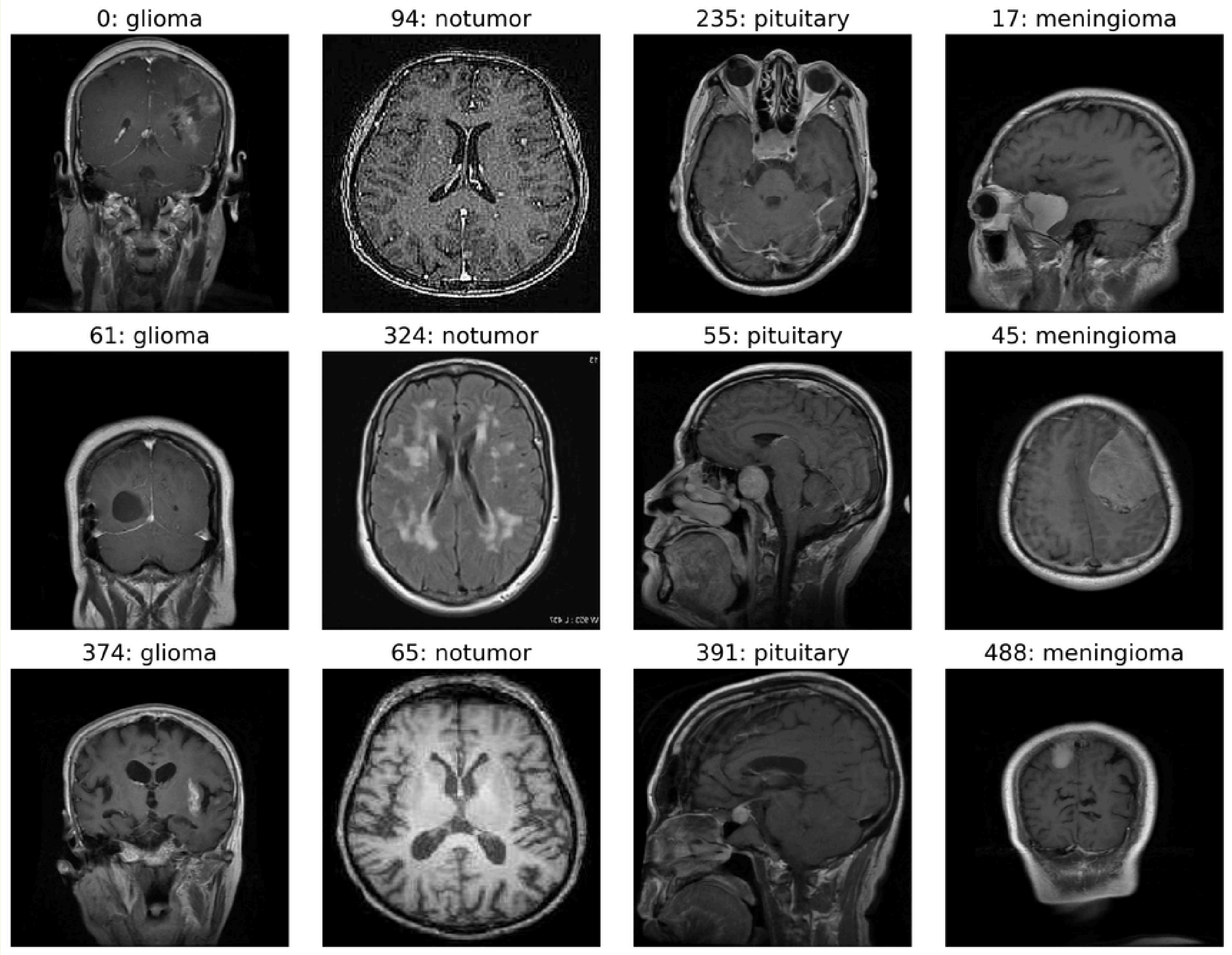
5 Evaluation, optimization and results

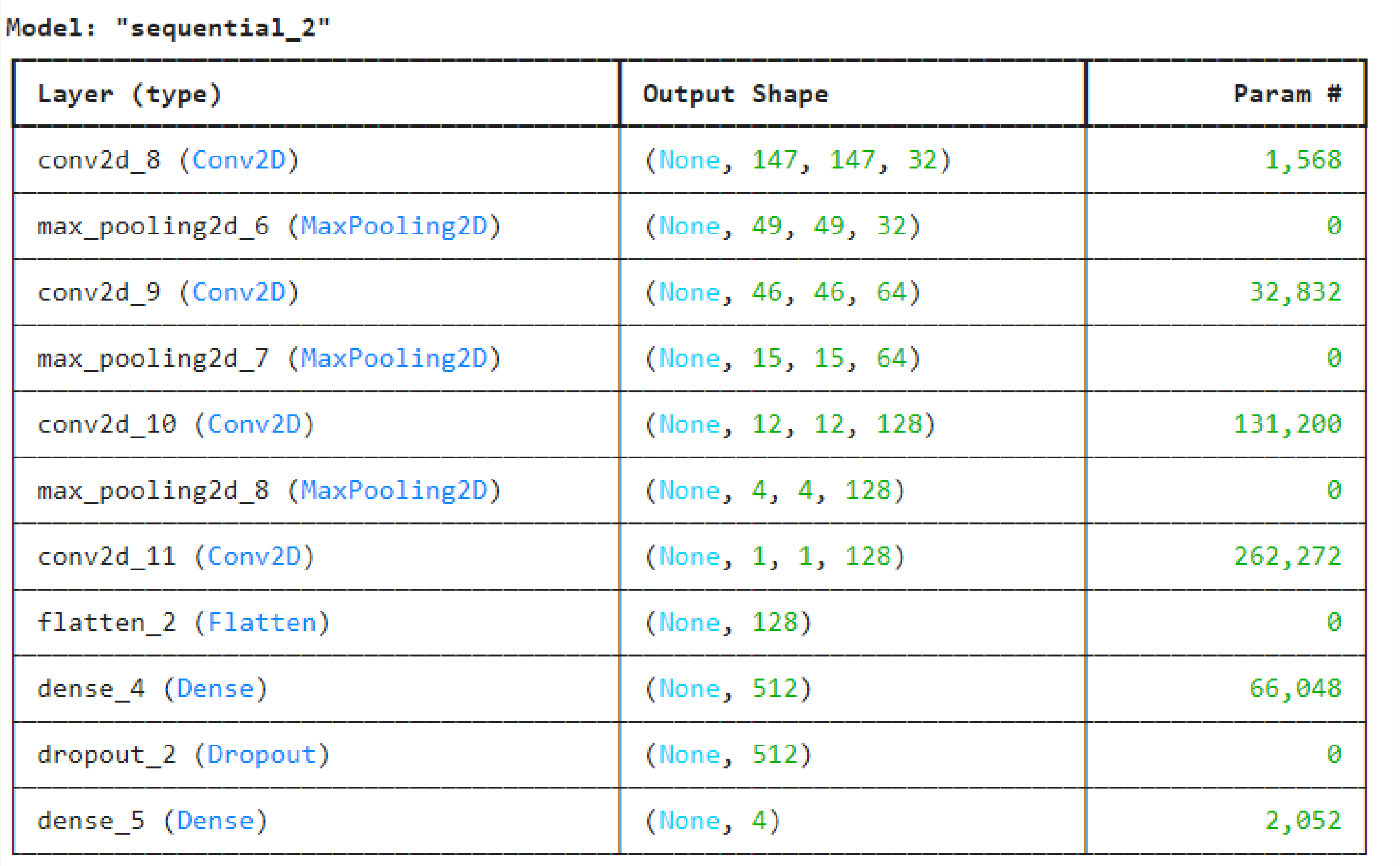
OUR MODEL

- Using sequential CNN to classify images and detect brain tumors
- Probably going to add another CNN-based model to segment the brain and identify the **location** of the tumor
- Different kinds of layers (Convolutional, Max Pooling, etc.)
- Using the categorical crossentropy loss function to optimaize results

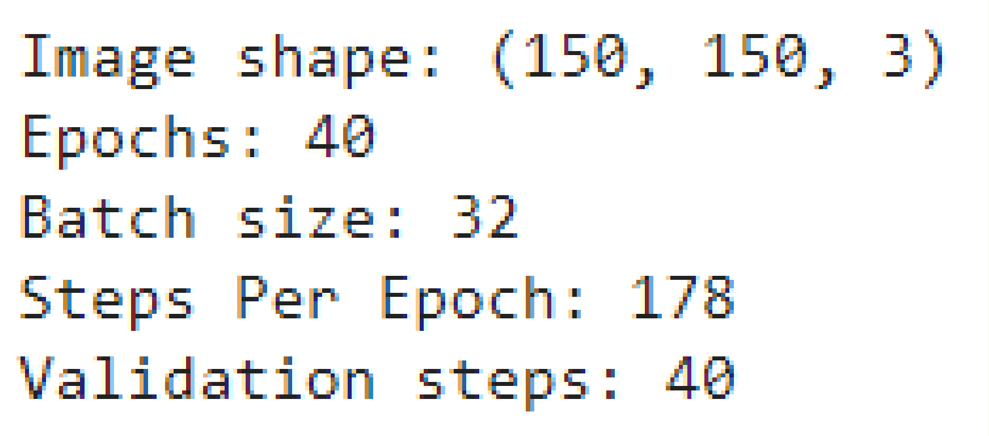


A SMALL TASTE





```
Model: "sequential_2"
┌──────────────────────────────────┬──────────────────────────────┬──────────────┐
│ Layer (type)                     │ Output Shape                 │      Param # │
├──────────────────────────────────┼──────────────────────────────┼──────────────┤
│ conv2d_8 (Conv2D)                │ (None, 147, 147, 32)         │        1,568 │
│ max_pooling2d_6 (MaxPooling2D)   │ (None, 49, 49, 32)           │            0 │
│ conv2d_9 (Conv2D)                │ (None, 46, 46, 64)           │       32,832 │
│ max_pooling2d_7 (MaxPooling2D)   │ (None, 15, 15, 64)           │            0 │
│ conv2d_10 (Conv2D)               │ (None, 12, 12, 128)          │      131,200 │
│ max_pooling2d_8 (MaxPooling2D)   │ (None, 4, 4, 128)            │            0 │
│ conv2d_11 (Conv2D)               │ (None, 1, 1, 128)            │      262,272 │
│ flatten_2 (Flatten)              │ (None, 128)                  │            0 │
│ dense_4 (Dense)                  │ (None, 512)                  │       66,048 │
│ dropout_2 (Dropout)              │ (None, 512)                  │            0 │
│ dense_5 (Dense)                  │ (None, 4)                    │        2,052 │
└──────────────────────────────────┴──────────────────────────────┴──────────────┘
```



```
Image shape: (150, 150, 3)
Epochs: 40
Batch size: 32
Steps Per Epoch: 178
Validation steps: 40
```

THANK YOU FOR LISTENING