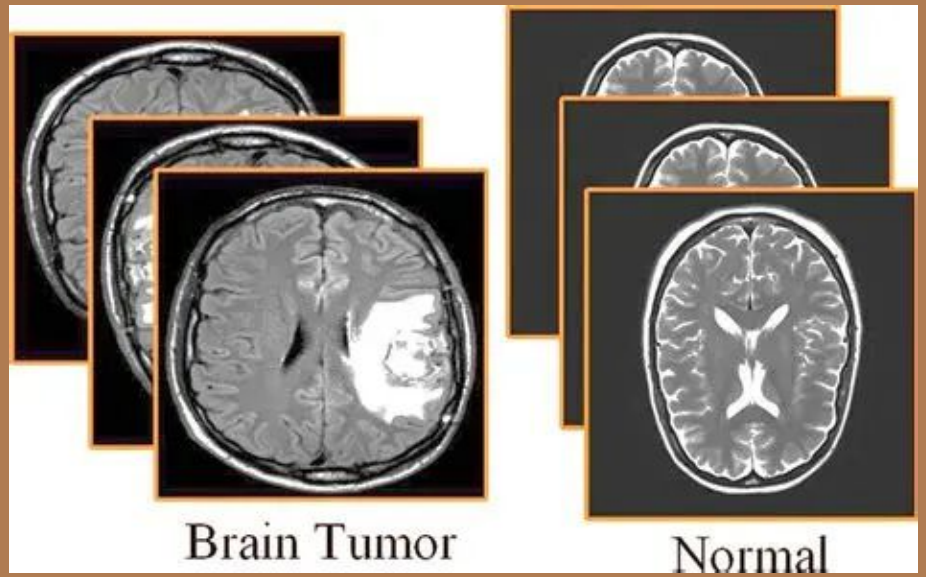



# Brain Tumor Classification with Deep Learning

By: Angeles Olvera


## Problem

NeuroScan Diagnostics has been contracted by a San Diego based hospital to develop a deep learning model for brain tumor classification using MRI data to help their radiology department make faster and more accurate diagnistics. They are focusing on classifying glioma tumors, meningioma tumors, pituitary tumors, or no tumor.

## Context

NeuroScan Diagnostics, is based in San Diego, CA, and is a leader in AI powered medical imaging and diagnostics. San Diego is renowned for its cutting edge research in neurology and biomedical sciences, making it the perfect environment for innovating advanced tumor detection models.

## Success Metrics

Accurate and timely classification of tumors based on MRI scans. The model must have high evaluation metric scores for precision, recall, F1- score, and accuracy.

## Constraints

MRI scan quality, patient positioning, and imaging equipment may affect the reliability of the model's predictions.

## Stakeholders

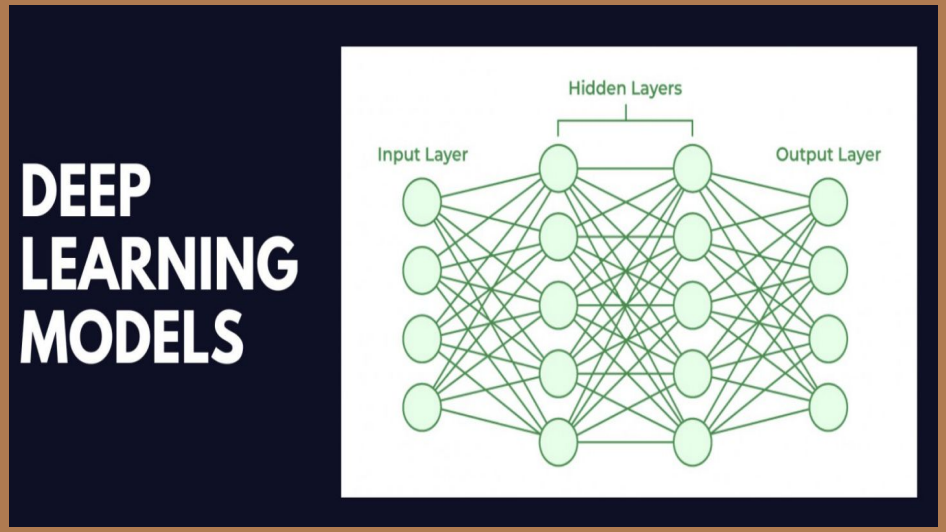
Dr. Olivia Matthews will provide clinical expertise to ensure the model aligns with real world radiology practices.

## Type of Problem

This is a supervised learning image classification task focused on detecting brain tumor types from MRI scans. Will be solved by using deep learning models.

## Why Deep Learning

Deep learning models are machine learning models that use layers of interconnected nodes (neurons) to learn from data. The “deep” part comes from having many layers, each layer extracts more abstract features.




## Data Description

- Downloaded from Kaggle
- The training set contains 3,310 images, while the test set includes 394 images.
- Each training and testing folder had four folder inside with images of the tumors. Each folder was named after the tumor represented in the images.
- Each image is a 2D grayscale scan with a resolution of 240x240 pixels.

## Project Workflow

**Step 1**: Data Wrangling and Pre-Processing

**Step 2**: Exploratory Data Analysis (EDA)

**Step 3**: Model Selection & Training

**Step 4**: Model Predictions and Evaluations

**Step 5**: Model Optimization

**Step 6**: Testing the Final Model

# Data Wrangling and Pre-Processing

**Building on the previously described dataset, the following steps were taken for deep learning model training.**

## Data Extraction

- Used Python zipfile library to extract contents
- Organized into folders per class
- Paths mapped using label driven for loops

## Image Preprocessing

- Grayscale conversion for consistent visualization (cv2.IMREAD_GRAYSCALE)
- Resized to **224×224** for model compatibility
- Applied **CLAHE** for localized contrast enhancement
- Normalized pixel values to [0,1] scale

## Data Preparation

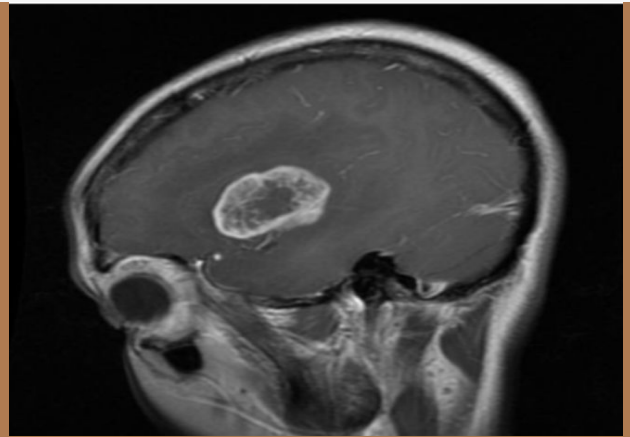
- Processed images stored in X_train, X_test
- Corresponding labels stored in y_train, y_test
- tqdm progress bar used for efficiency tracking

```
labels = ['glioma_tumor', 'meningioma_tumor', 'no_tumor', 'pituitary_tumor']
```

```
X_train = []
y_train = []

image_size = 224
for i in labels:
    folderPath = os.path.join('dataset_extracted/Tumor Data', 'Training', i)
    for j in tqdm(os.listdir(folderPath)):
        img = cv2.imread(os.path.join(folderPath,j),cv2.IMREAD_GRAYSCALE)
        img = cv2.resize(img,(image_size, image_size))
        clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(8,8))
        img = clahe.apply(img)  # Apply CLAHE for controlled contrast enhancement
        img = img / 255.0  # Normalize pixel values to range [0, 1]
        X_train.append(img)
        y_train.append(i)
```

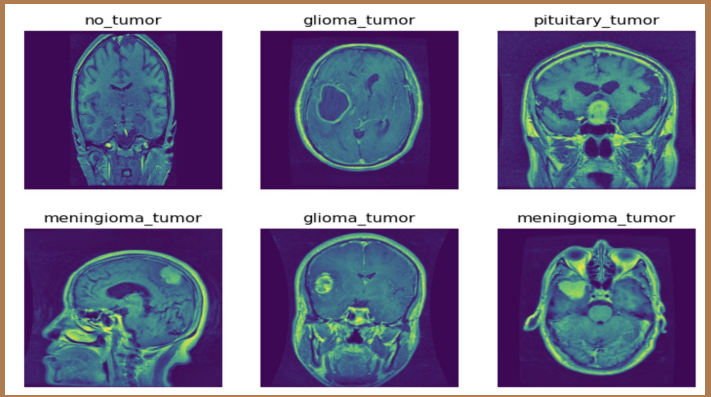
Image Before transformations



# Exploratory Data Analysis

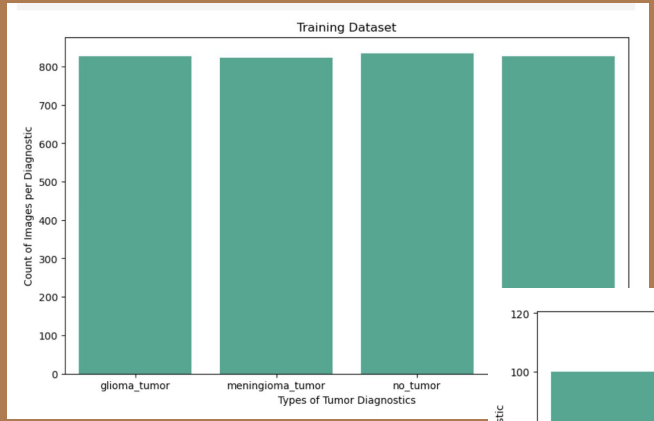
## Visual Checks After Transformations

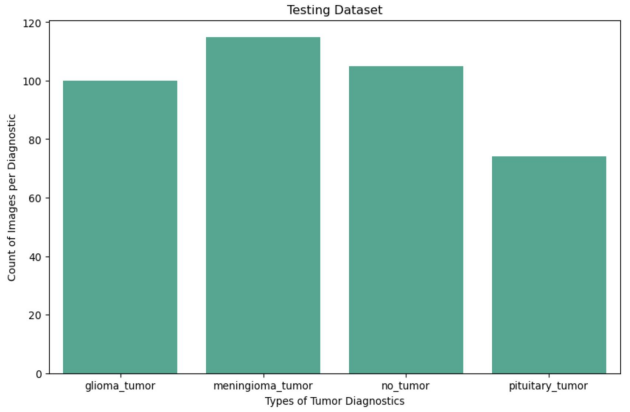
- Random MRI samples printed from training/testing sets
- CLAHE preprocessing improved contrast in low-detail regions




## Class Balances

- **Training data**: Balanced across 4 categories
- **Testing data**: Imbalanced  relied on precision, recall, F1-score for fair evaluation



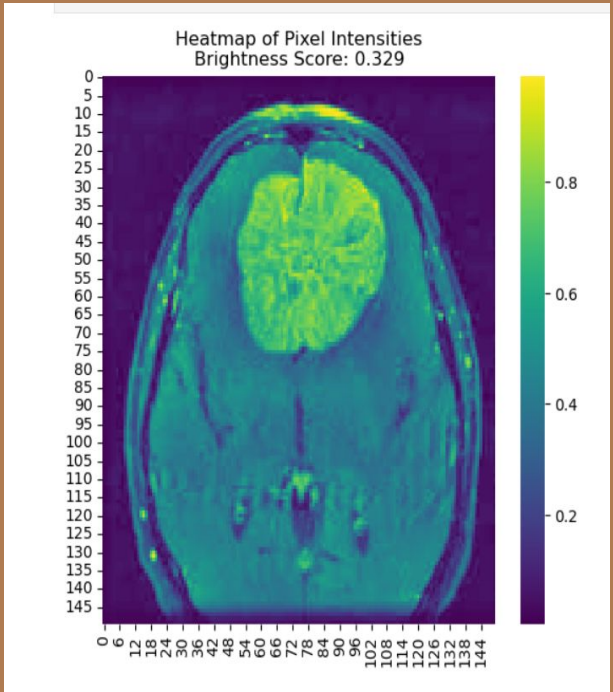




## Dataset Shape

- **Training set**: 3,310 images at 224×224 pixels
- **Test set**: 394 images at 224×224 pixels

# Pixel Intensity Heatmaps

- Visualized brightness across scans
- Bright regions linked to potential tumor presence
- Aids interpretation and supports quality assurance before model training




## RGB Conversion

- Converted grayscale images to 3-channel RGB
- Image arrays renamed: X_train_rgb, X_test_rgb
- Ensures compatibility with ImageNet-pretrained CNN models

## Label Preparation

- Used LabelEncoder() for numeric conversion of categories
- Created both encoded (y_train_encoded, y_test_encoded) and one-hot versions (y_train_cat, y_test_cat)

# Exploratory Data Analysis

# Model Selection & Training

## ResNet50 Custom Model Setup

**Purpose**: To evaluate a model with low layers and a complex classifier would perform on a small medical imaging dataset.

- Pretrained on **ImageNet**
- Used include_top=False to remove default classifier
- res_net_model.trainable = **False** keep all the weights unchanged during training

---

- Added **GlobalAveragePooling2D** to compress features
- Dropout(0.5) for regularization
- Dense(128) layer with **ReLU** activation
- Dropout(0.25) for light regularization
- Dense(4) with **Softmax** activation for 4 class prediction
- Compiled with **Adam**, **categorical cross-entropy**, and **accuracy**

## DenseNet121 Custom Model Setup

**Purpose**: To evaluate a model with deeper architecture (more layers ) and a less complex classifier.

- Pretrained on **ImageNet**
- Used include_top=False to insert custom classifier
- dense_model.trainable = **False** keep all the weights unchanged during training

---

- Dense(64) layer with **ReLU** activation
- Dropout(0.5) for regularization
- Dense(4) with **Softmax** activation for 4 class prediction
- Compiled with **Adam**, **categorical cross-entropy**, and **accuracy**

## Xception Custom Model Setup

**Purpose**: Use a model that is not defined by its depth but rather feature extraction and learning techniques.

- Pretrained on **ImageNet**
- Used include_top=False to insert custom classifier
- xception_model.trainable = **False** keep all the weights unchanged during training

---

- Added **GlobalAveragePooling2D** to compress features
- Dense(64) layer with **ReLU** activation
- Dropout(0.3) for light regularization
- Dense(4) with **Softmax** activation for 4 class prediction
- Compiled with **Adam**, **categorical cross-entropy**, and **accuracy**

# Predictions and Evaluations Strategies

## Prediction Strategy

- Created predictions() function to create predictions per model.

```
def predictions(model, train_data):
    probs = model.predict(train_data)
    return np.argmax(probs, axis=1)
```

## Evaluation Strategy

- Created `evaluate()` function for model evaluation.
  - **Precision**, **Recall**, **F1-Score**
  - Recall here represents false negatives on actual cases.

```
def evaluate(model, y_train, y_pred):
    report = classification_report(y_train, y_pred, output_dict=True)

    precision_score = report['macro avg']['precision'] * 100
    recall_score = report['macro avg']['recall'] * 100
    f1_score_ = report['macro avg']['f1-score'] * 100

    print(f"Precision: Out of all the predictions (y_pred), the model was correct {precision_score:.3f} % of the time")
    print(f"Recall: Out of all the actual cases(y_train/y_test), the model successfully identified {recall_score:.3f} % of th
    print(f"F1-score: Balance between precision and recall is {f1_score_:.3f}")
```

- Added separate `evaluate_accuracy()` function.
  - How many predictions the model got correct.

```
def evaluate_accuracy(y_train, y_pred):
    accuracy = accuracy_score(y_train,y_pred) * 100

    print(f"Accuracy: The models overal accuracy is {accuracy:.3}")
```

# Model Evaluations

## ResNet50 Evaluation Summary

- **Precision**: 64%
- **Recall**: 59%
- **F1-Score**: 60%
- **Accuracy**: 59%
- Struggled to classify tumors reliably
- Fewer layers with more neurons led to overfitting
- **Conclusion**: Underperformed with MRI scans.

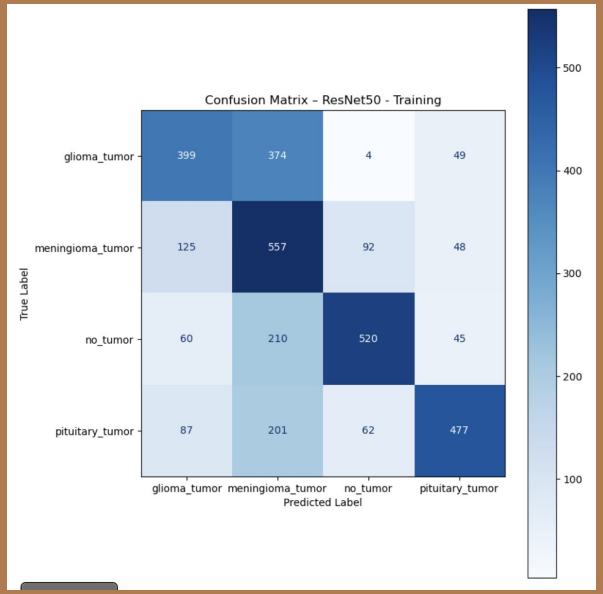
## DenseNet121 Evaluation Summary

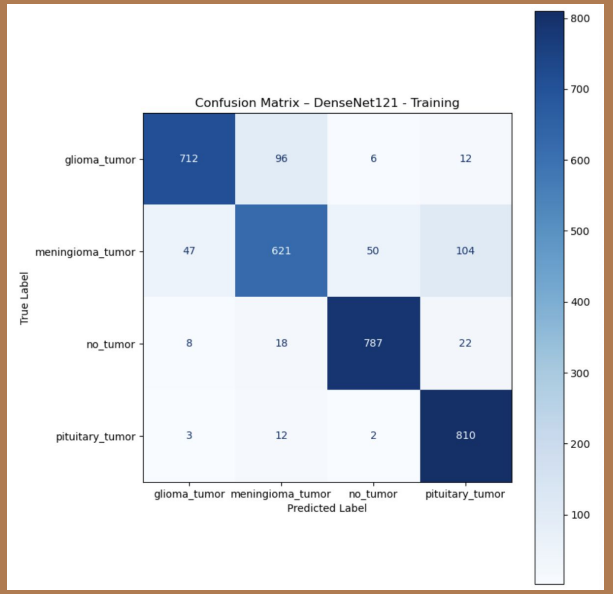
- **Precision**: 88.54%
- **Recall**: 88%
- **F1-Score**: 88%
- **Accuracy**: 88%
- Deep architecture captured complex tumor patterns
- Balanced generalization across all classes
- **Conclusion**: Strong performance with MRI scans.

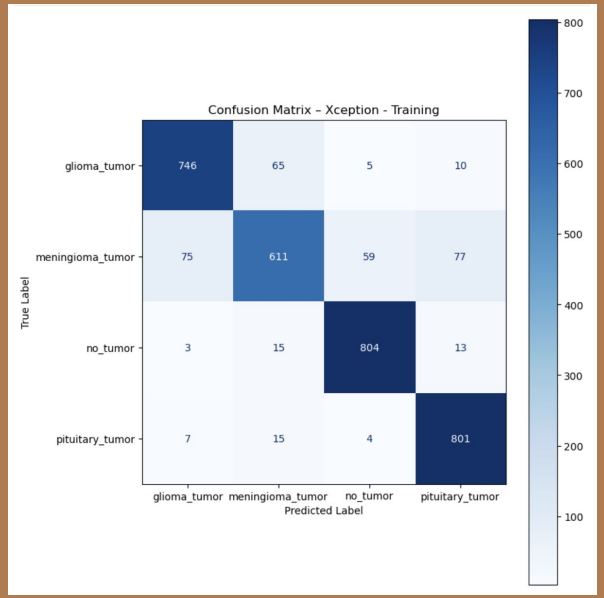
## Xception Evaluation Summary

- **Precision**: 89.35%
- **Recall**: 89%
- **F1-Score**: 89%
- **Accuracy**: 89%
- Slight edge over DenseNet121 in all metrics
- **Conclusion**: Strong performance with MRI scans. Top performing model.

# Confusion Matrix



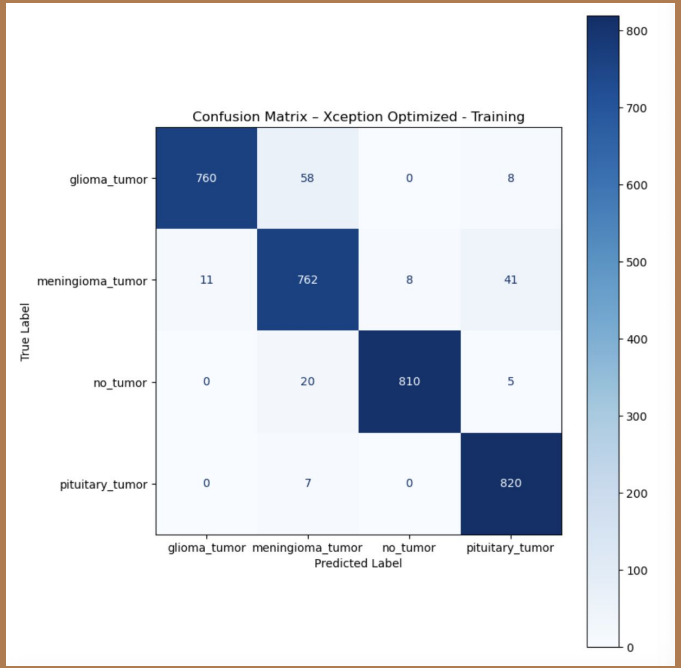




- **DenseNet121** slightly outperformed Xception on *meningioma* and *pituitary* tumor detection
- Correctly classified 9–10 more images in those categories
- Sparked interest in fine-tuning **Xception** to match or exceed that performance

# Model Optimization

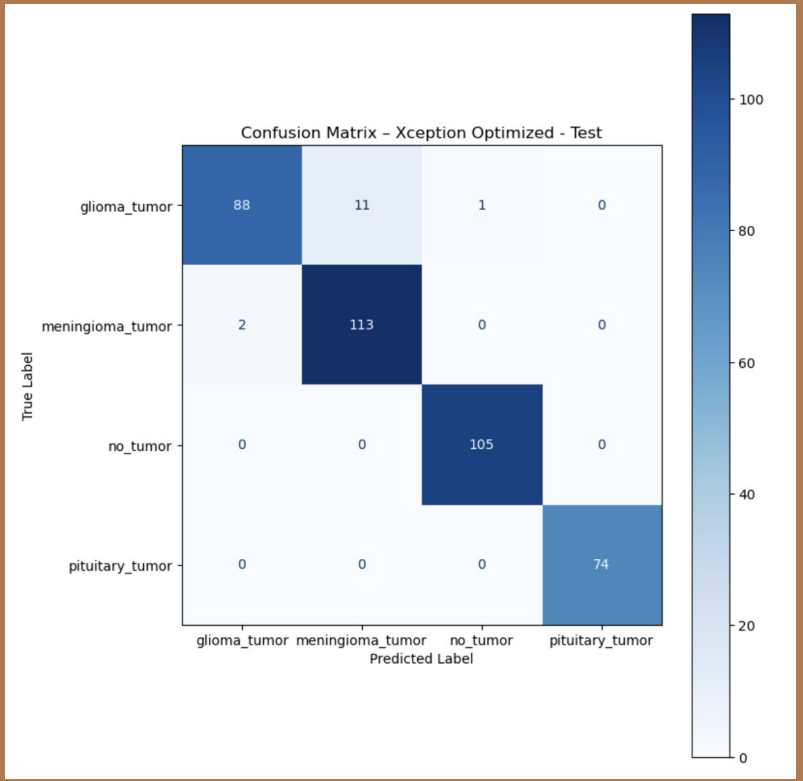
## Manual Optimization: Xception Model

- Increased Dense layer size from **64  to 256 neurons**
- Reduced dropout rate from **30% to 10%**
- Model architecture unchanged beyond hyperparameters
- Achieved significant performance boost:
    - **Precision**: 95%
    - **Recall**: 95%
    - **F1-Score**: 95%
    - **Accuracy**: 95%
- Confusion matrix shows improved predictions across all 4 tumor classes



# Testing the Final Model

- Evaluated on **394 test images**, preprocessed identically to training data
- Achieved strong generalization on unseen data:
  - **Precision**: 97%
  - **Recall**: 96%
  - **F1-Score**: 97%
  - **Accuracy**: 96%
- Consistent correct classifications across all four classes
- Minimal misclassifications suggest **no overfitting or underfitting**
- Confirms the model's reliability for real world application.



# Recommendations

- **Streamline Workflow**: Automate MRI scan diagnostics to free up clinicians for personalized treatment planning.
- **Support Research**: Analyze tumor patterns and demographics to uncover trends in patients to prevent or early detect brain tumors.
- **Enable Early Detection**: Integrate as a screening tool for identifying tumors in high risk individuals.

## Suggestions For Improvement

### Class-Specific Fine-Tuning

- Create a nested model in the final model that retains only on glioma and meningioma images.
- Helps the model learn subtle differences it may be glossing over in full multiclass training.