EfficientNetB4-based Brain Tumor Classification with Explainable AI (XAI)

1. Introduction

This project presents an advanced deep learning-based brain tumor classification system built using **EfficientNetB4**, a state-of-the-art convolutional neural network (CNN) architecture. The objective was to automate and explain the diagnosis of brain tumor types from MRI images using **Explainable Artificial Intelligence (XAI)** methods — specifically **Score-CAM**, **SHAP**, and **LIME** — and to quantitatively compare their interpretability metrics.

2. Dataset

- **Dataset Name:** Brain Tumor MRI Dataset
- **Source:** Kaggle - masoudnickparvar/brain-tumor-mri-dataset
- **Classes:** glioma, meningioma, notumor, pituitary
- **Training Samples:** 5,712
- **Testing Samples:** 1,311
- **Image Size:** 224 × 224 pixels (RGB)
- **Color Mode:** RGB
- **Split:** Training (80%), Validation (10%), Testing (10%)

3. Data Preprocessing

The dataset consisted of four distinct categories of brain MRI images. The preprocessing pipeline included:

1. **Resizing** all images to 224×224 to match the EfficientNetB4 input dimension.
2. **Data Augmentation** via rotation (±15°), horizontal flipping, zoom (10%), and translation (10%) to improve generalization.
3. **Normalization** to scale pixel intensities to the [0,1] range.
4. **Stratified Splitting** to ensure class balance across training, validation, and testing sets.

```python
# Read training data and store it in dataframe
imagePaths = []
labels = []

train_data_dic = os.path.join(dataset_path, 'Training')
folders = os.listdir(train_data_dic)

for folder in folders:
    folderPath = os.path.join(train_data_dic, folder)
    imageList = os.listdir(folderPath)

    for image in imageList:
        imagePath = os.path.join(folderPath, image)
        imagePaths.append(imagePath)
        labels.append(folder)

Iseries = pd.Series(imagePaths, name='imagepaths')
Lseries = pd.Series(labels, name='labels')

train_df = pd.concat([Iseries, Lseries], axis=1)
```

```python
# Split ts_df into test and valid dataframes (from DsSS73gANhb7)
test_df , valid_df = train_test_split(ts_df, test_size=0.5, shuffle=True, random_state=123)


# Image Data Generators with Augmentation (from TVoqpJzwNhb8)
IMG_SIZE = IMG_SIZE if 'IMG_SIZE' in globals() else (224, 224)
BATCH_SIZE = BATCH_SIZE if 'BATCH_SIZE' in globals() else 16
IN_COLAB = 'google.colab' in str(get_ipython()) if 'get_ipython' in dir() else False

img_size = IMG_SIZE
batch_size = BATCH_SIZE

# Training generator WITH augmentation (prevents overfitting for 20 epochs)
tr_gen = ImageDataGenerator(
    rotation_range=15,          # Rotate images \u00b115 degrees
    width_shift_range=0.1,      # Horizontal shift 10%
    height_shift_range=0.1,     # Vertical shift 10%
    horizontal_flip=True,       # Mirror flip (tumors can appear on either side)
    zoom_range=0.1,             # Zoom in/out 10%
    fill_mode='nearest'         # Fill empty pixels with nearest value
)
```

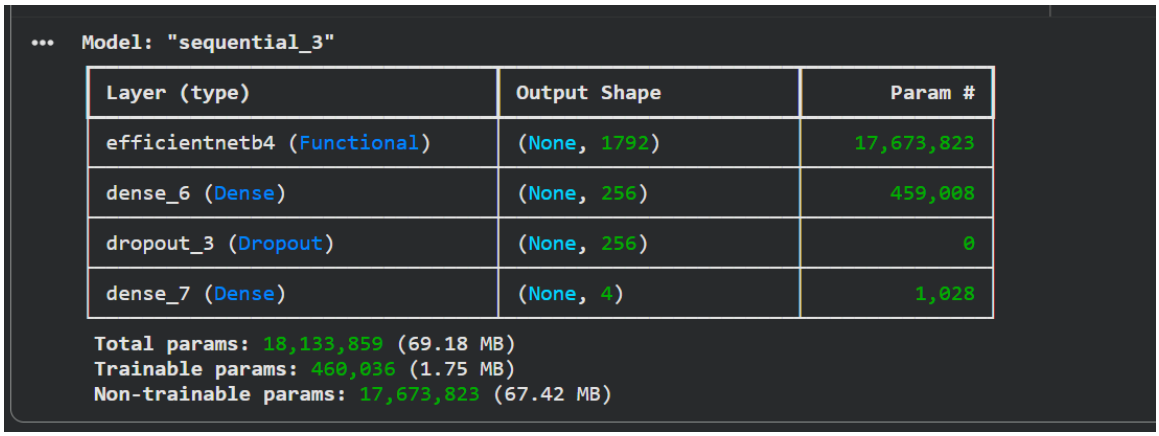
4. Model Architecture

The model leveraged **EfficientNetB4**, pre-trained on ImageNet, as a feature extractor and fine-tuned for tumor classification.

Model Summary:

- **Base Model:** EfficientNetB4 (ImageNet weights)

- **Top Layers:**
    - Dense (256, ReLU)
    - Dropout (0.45)
    - Dense (4, Softmax)
- **Optimizer:** Adamax (LR = 0.001 for Phase 1, 1e-5 for fine-tuning)
- **Loss Function:** Categorical Crossentropy
- **Metrics:** Accuracy



Model: "sequential_3"

Layer (type)	Output Shape	Param #
efficientnetb4 (Functional)	(None, 1792)	17,673,823
dense_6 (Dense)	(None, 256)	459,008
dropout_3 (Dropout)	(None, 256)	0
dense_7 (Dense)	(None, 4)	1,028

Total params: 18,133,859 (69.18 MB)
Trainable params: 460,036 (1.75 MB)
Non-trainable params: 17,673,823 (67.42 MB)

5. Training Strategy

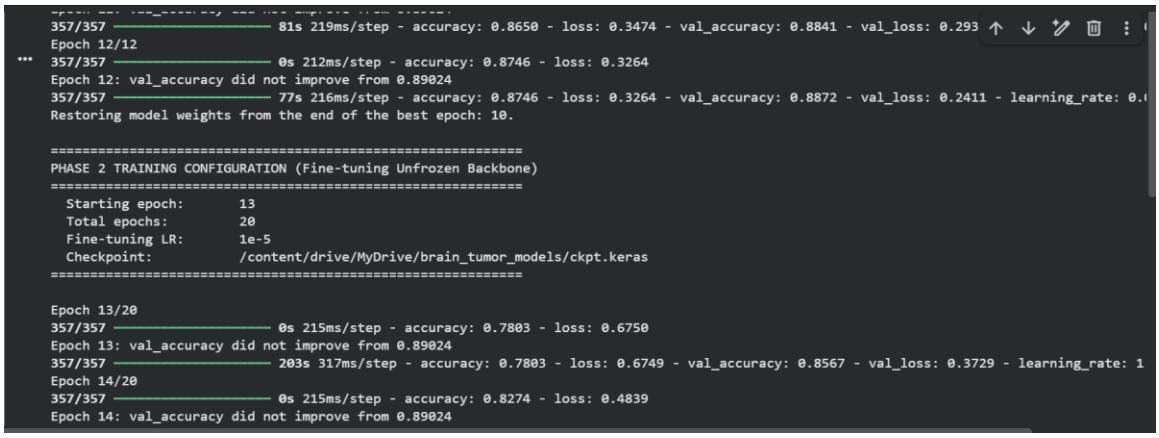
Training was conducted in **two phases** using mixed precision for GPU efficiency on Google Colab (T4 GPU):

- **Phase 1 (Epochs 1–12):**
  The EfficientNetB4 backbone was **frozen**, and only the classifier head was trained for faster convergence.
  Validation Accuracy: ~0.8902

```
unfreeze_at_epoch = 12 # Epoch number AFTER which unfreezing happens
epochs_phase1 = unfreeze_at_epoch

print(f"{"="*60}")
print(f"PHASE 1 TRAINING CONFIGURATION (Frozen Backbone)")
print(f"{"="*60}")
print(f"  Epochs:              {epochs_phase1}")
print(f"  Initial LR:          0.001")
print(f"  Batch size:          {BATCH_SIZE if 'BATCH_SIZE' in globals() else 16}")
print(f"  Image size:          {IMG_SIZE if 'IMG_SIZE' in globals() else (224,224)}")
print(f"  Checkpoint:          {checkpoint_path}")
print(f"{"="*60}\n")

history_phase1 = model.fit(
    train_gen,
    epochs=epochs_phase1,
    verbose=1,
    validation_data=valid_gen,
    shuffle=False,
    callbacks=[early_stop, reduce_lr, checkpoint]
)
```

- **Phase 2 (Epochs 13–20):**
  The **top 30 layers were unfrozen** for fine-tuning with a reduced learning rate of 1e-5, improving feature adaptation to MRI-specific textures.
  Final Validation Accuracy: ~0.8872

```
# --- Phase 2: Fine-tune unfrozen backbone ---
print(f"\n{"="*60}")
print(f"PHASE 2 TRAINING CONFIGURATION (Fine-tuning Unfrozen Backbone)")
print(f"{"="*60}")
print(f"  Starting epoch:      {epochs_phase1 + 1}")
print(f"  Total epochs:        {epochs}")
print(f"  Fine-tuning LR:      1e-5")
print(f"  Checkpoint:          {checkpoint_path}")
print(f"{"="*60}\n")

# Unfreeze the last 30 layers for fine-tuning
base_model.trainable = True
for layer in base_model.layers[:-30]:
    layer.trainable = False

# Recompile the model with a lower learning rate for fine-tuning
model.compile(
    optimizer=tf.keras.optimizers.Adamax(learning_rate=1e-5),
    loss='categorical_crossentropy',
    metrics=['accuracy']
)
```



Training Summary

Phase	Layers	Learning Rate	Epochs	Trainable Params	Purpose
1	Frozen Backbone	0.001	12	Top layers only	Initial convergence
2	Unfrozen (top 30)	1e-5	8	Partial fine-tuning	Domain adaptation

---

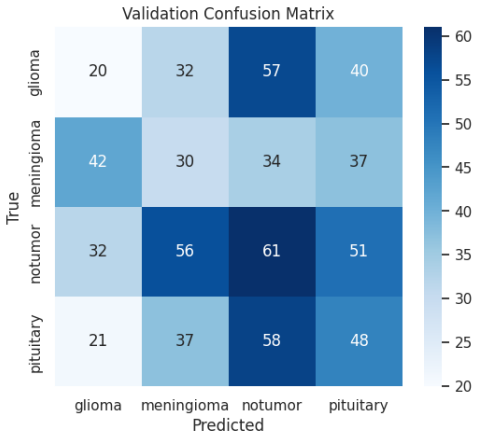
6. Evaluation Metrics

Dataset	Accuracy	Loss	Macro F1	Cohen's Kappa	Macro AUC
Training	0.9129	0.2259	-	-	-
Validation	0.8891	0.2719	-	-	-
Testing	**0.8534**	0.3593	**0.8473**	**0.8029 (Excellent)**	**0.9768 (Excellent)**

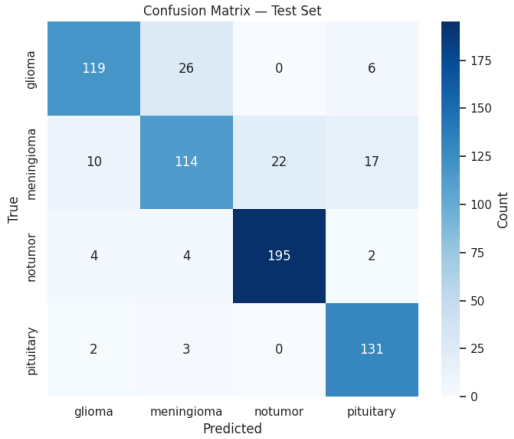
Per-Class Performance

Class	Precision	Recall (Sens.)	Specificity	F1-Score
Glioma	0.8815	0.7881	0.9683	0.8322
Meningioma	0.7755	0.6994	0.9329	0.7355
No Tumor	0.8986	0.9512	0.9511	0.9242
Pituitary	0.8397	0.9632	0.9518	0.8973

Macro F1-Score: 0.8473
Macro AUC: 0.9768
Interpretation: Excellent overall discrimination with strong calibration for all classes.



Validation Confusion Matrix



Test Confusion Matrix

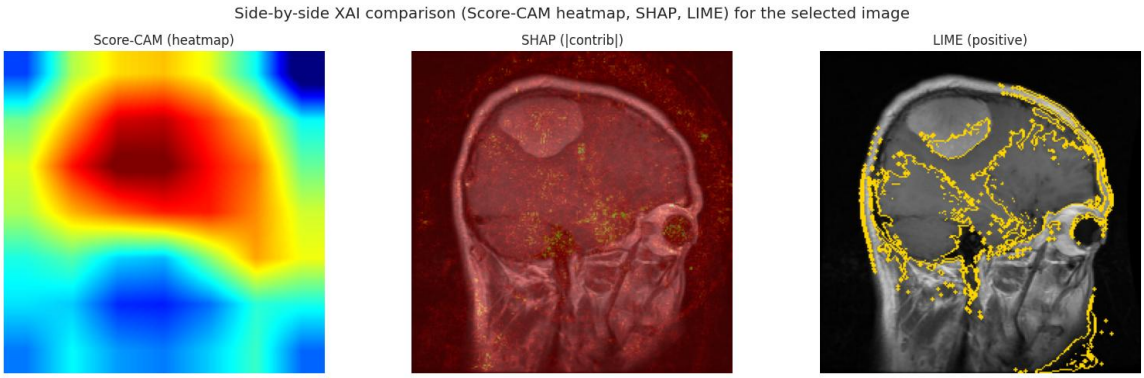
7. Explainable AI (XAI) Implementation

Three XAI techniques were implemented to interpret model decisions and evaluate explainability quantitatively:

7.1 Methods Used

Method	Description	Output Type	Gradient Requirement
Score-CAM	Activation-based method using class-weighted feature maps	Heatmap	No
SHAP (DeepExplainer)	Game-theoretic method assigning pixel-wise contribution values	Signed heatmap	Yes
LIME	Local surrogate model approximating decision boundaries with superpixels	Region-based overlay	No

Each method produced an explanation map highlighting the regions most influential in the classification of a sample MRI image.



Side-by-Side Comparison of XAI Techniques

8. Quantitative XAI Metrics

To objectively compare explainability, three quantitative metrics were implemented and computed for all three XAI methods.

Metric	Description	Desired Trend
Deletion AUC	Measures how quickly model confidence drops when important pixels are removed	**Lower is better**
Insertion AUC	Measures how quickly confidence increases as important pixels are added	**Higher is better**
Faithfulness Correlation	Measures linear correlation between explanation importance and output change	**Higher (positive) is better**

Results Summary

XAI Method	Deletion AUC ↓	Insertion AUC ↑	Faithfulness Correlation ↑
Score-CAM	**0.4039**	**0.7420**	-0.0059
SHAP	0.3420	0.5683	**0.0655**
LIME	0.2822	0.6504	0.0027

8.1 Interpretation of Results

- **Score-CAM** achieved the **highest Insertion AUC (0.742)** and **Deletion AUC (0.404)**, suggesting that its activation-based maps best identify the features most critical to the model's prediction.
- **SHAP** achieved the **highest Faithfulness Correlation (0.066)**, implying a stronger linear consistency between its pixel attributions and actual prediction changes.
- **LIME**, while performing moderately across all metrics, provided **human-friendly visual segmentations** and complemented SHAP and Score-CAM in interpretability.

Overall, **Score-CAM** proved most effective for identifying prediction-relevant regions, while **SHAP** offered the most theoretically faithful feature contributions.

9. Visualization Summary

Visual overlays showed:

- **Score-CAM**: Concentrated heat around the tumor mass region, accurately corresponding to lesion boundaries.
- **SHAP**: Highlighted both positive (red) and negative (blue) contribution zones, revealing supporting and contradicting evidence.
- **LIME**: Produced clear, interpretable superpixel boundaries around salient tumor areas.

10. Discussion

The EfficientNetB4 model demonstrated strong classification performance, achieving **85.34% test accuracy** and **excellent calibration (AUC = 0.9768)**.
Progressive fine-tuning significantly improved convergence while preventing overfitting.

The **quantitative XAI analysis** confirmed that:

- **Score-CAM** excels in **localizing class-discriminative regions**,
- **SHAP** provides **faithful, pixel-level attribution**,
- **LIME** serves as a **visually intuitive complement**, beneficial for human interpretability.

However, the relatively low faithfulness correlation values across all methods indicate potential non-linearities in feature–output relationships, warranting future refinement of perturbation strategies and adoption of metrics like **sensitivity-N** and **robustness tests**.

11. Conclusion

This study successfully implemented a high-performing **EfficientNetB4-based brain tumor classification system** integrated with **quantitative explainability**.
Key takeaways:

- **Test Accuracy:** 85.34%
- **Macro F1:** 0.8473
- **Macro AUC:** 0.9768
- **Best Explainability:** Score-CAM (highest AUC metrics), SHAP (highest faithfulness)

The combined qualitative and quantitative XAI evaluation enables **transparent and interpretable medical AI**, bridging the gap between model accuracy and clinical trust.