AIDI1009-24F-10827:** Assignment #3 - AI-Driven Brain Tumor Classification Using ResNet50 + MLJAR AutoML**

Group Members:

• Ranveer Singh Saini - 200569800

• Girik Nohani - 200565756

• Pooja Indraj Yadav - 200568689

• Ronald Kalani - 200619730

Instructor: Jahanzeb Abbas

Course: Technology & Visual Arts - AIDI 1010 - Emerging Technologies

Submission Date: April 19, 2025

AI-Driven Brain Tumor Classification Using ResNet50 and MLJAR AutoML

This project focuses on developing an AI-powered diagnostic system capable of accurately classifying brain tumor types from MRI images. By integrating a deep learning model, ResNet50, with the MLJAR AutoML platform, the solution explores a hybrid approach to leverage both image-based pattern recognition and structured data automation. This dual-strategy setup aims to enhance classification accuracy, reduce manual diagnostic time, and minimize variability in radiological interpretations, providing a practical and scalable solution for real-world healthcare environments.

Goal

The main goal is to build an accurate and efficient classification system that automates brain tumor detection in MRI scans. The project implements a hybrid approach: first, using ResNet50, a deep convolutional neural network pre-trained on ImageNet, to extract image features and perform classification via transfer learning. Then, MLJAR AutoML is applied to structured representations of MRI data to automate benchmarking, model selection, and optimization. Together, these approaches aim to deliver robust performance, handle small dataset constraints, and provide transparency through evaluation metrics and comparisons with previous prototypes.

Intended Audience

This project is targeted at multiple key stakeholders in the healthcare and artificial intelligence sectors. These include clinical researchers and radiologists who benefit from quicker and more consistent diagnostic tools, and AI engineers and data scientists interested in the development and application of deep learning in medical imaging. Moreover, healthcare investors, hospital administrators, and MedTech stakeholders looking to explore AI-driven diagnostic enhancements are also an integral part of the audience. The project is designed to be interpretable, reproducible, and clinically meaningful.

Strategy & Pipeline Steps

The strategy for this project was broken down into clear and systematic pipeline steps. Initially, MRI brain images were preprocessed by resizing to a fixed dimension (typically 224x224 pixels) and normalized to fit the input requirements of ResNet50. Transfer learning was then applied using the ResNet50 model with pre-trained weights from ImageNet to fine-tune the model on the specific tumor classification task. The dataset was split into training and testing sets using stratified sampling to maintain class balance.

Simultaneously, image features and metadata were transformed into structured formats to be fed into the MLJAR AutoML platform, which performed feature selection, model training, and hyperparameter tuning across various machine learning algorithms. Model performance was evaluated using metrics such as accuracy, precision, recall, F1-score, confusion matrix, and ROC-AUC curves. A comparison was made with the baseline prototype developed in Assignment 2 and referenced literature to assess improvements.

Optionally, conceptual extensions were discussed including the application of Artificial General Intelligence (AGI) and the future integration of quantum computing for complex diagnostic environments with high-dimensional imaging data.

Challenges

Several key challenges were encountered during the project lifecycle. One major issue was class imbalance, where certain tumor categories such as Glioma) had more samples than others like Pituitary tumors, which could bias the model. To counter this, data augmentation techniques such as image rotation and flipping were used. Another significant limitation was the small dataset size, which posed overfitting risks during deep learning training. This was mitigated by applying transfer learning with ResNet50 to utilize generalized features learned from a large corpus.

Moreover, computational constraints were a hurdle, especially on local machines lacking high-performance GPUs. Long training times and memory bottlenecks were solved by optimizing batch sizes and freezing early ResNet layers. Despite these issues, the model was trained successfully with satisfactory results.

Problem Statement

The problem tackled in this project is of critical relevance in the medical imaging domain: Can brain tumor types be accurately predicted from MRI images using automated deep learning systems, reducing the diagnosis time and variability in radiology?

To address this, the project seeks to develop a model capable of classifying MRI brain scans into one of three tumor classes: Glioma, Meningioma, and Pituitary tumor. The aim is to support radiologists with a second opinion system that improves early detection, consistency, and speed of diagnosis. This is especially important in under-resourced clinical settings or in regions with limited access to radiological expertise.

Data set

The dataset used for this project is the Brain Tumor Classification (MRI) dataset sourced from Kaggle, consisting of three tumor classes: Glioma, Meningioma, and Pituitary tumor. It contains T1-weighted contrast-enhanced MRI images categorized into labeled folders for each tumor type. The dataset includes training and testing subsets, with images in JPEG format, varying in resolution, and captured from different patients, allowing for supervised learning and classification tasks. This dataset supports both deep learning (image-based) and structured feature extraction for hybrid modeling.

Step 1: Mount Google Drive and Prepare Dataset

Mount Google Drive to access your image folders, and then organize all tumor and normal class images into a single unified dataset directory (/content/dataset_combined) with four subfolders.

```
from google.colab import drive
drive.mount('/content/drive')
```

```
Drive already mounted at /content/drive; to attempt to forcibly remount, call drive.mount("/c
ontent/drive", force_remount=True).
```

```
import shutil, os

# Create a new directory for unified dataset
new_data_path = '/content/dataset_combined'
os.makedirs(new_data_path, exist_ok=True)

# Copy folders
shutil.copytree('/content/drive/MyDrive/Emerging/Normal', f'{new_data_path}/Normal')
shutil.copytree('/content/drive/MyDrive/Emerging/Tumor/glioma_tumor', f'{new_data_path}/glioma
shutil.copytree('/content/drive/MyDrive/Emerging/Tumor/meningioma_tumor', f'{new_data_path}/me
shutil.copytree('/content/drive/MyDrive/Emerging/Tumor/pituitary_tumor', f'{new_data_path}/pit
```

```
'/content/dataset_combined/pituitary_tumor'
```

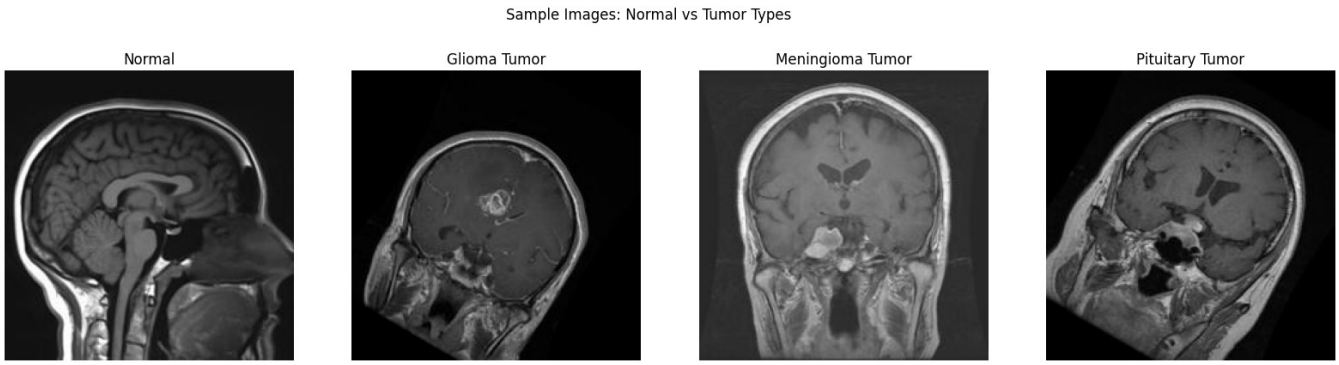
Step 2: Visualize Sample Images from Each Class

Load and display a sample image from each of the four categories to gain a visual understanding of the differences between tumor types and normal brain scans.

```
# Step 2: Visualize Example Images
import matplotlib.pyplot as plt
import cv2

classes = ['Normal', 'glioma_tumor', 'meningioma_tumor', 'pituitary_tumor']
```

```
fig, axs = plt.subplots(1, 4, figsize=(20, 5))
for i, cls in enumerate(classes):
    cls_path = os.path.join(new_data_path, cls)
    sample_img = os.listdir(cls_path)[0]
    img = cv2.imread(os.path.join(cls_path, sample_img))
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    axs[i].imshow(img)
    axs[i].axis('off')
    axs[i].set_title(cls.replace('_', ' ').title())
plt.suptitle('Sample Images: Normal vs Tumor Types')
plt.show()
```



Step 3: Data Preprocessing Using ImageDataGenerator

Create image generators for training and validation datasets with 80-20 split using ImageDataGenerator. Images are resized to 224x224 and normalized (rescale 1./255).

```
# Step 3: Preprocess with ImageDataGenerator
from tensorflow.keras.preprocessing.image import ImageDataGenerator

datagen = ImageDataGenerator(
    rescale=1./255,
    validation_split=0.2
)

train_gen = datagen.flow_from_directory(
    new_data_path,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='training'
)

val_gen = datagen.flow_from_directory(
    new_data_path,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical',
    subset='validation'
)
```

```
Found 17339 images belonging to 4 classes.
Found 4333 images belonging to 4 classes.
```

** Step 4: Build the Transfer Learning Model (ResNet50)**

```
# Step 4: Transfer Learning with ResNet50
from keras.applications import ResNet50
from keras.models import Sequential
from keras.layers import Dense, GlobalAveragePooling2D
from keras.optimizers import Adam

base_model = ResNet50(weights='imagenet', include_top=False, input_shape=(224,224,3))
base_model.trainable = False

model = Sequential([
    base_model,
    GlobalAveragePooling2D(),
    Dense(128, activation='relu'),
    Dense(4, activation='softmax')  # 4 output classes
])

model.compile(optimizer=Adam(), loss='categorical_crossentropy', metrics=['accuracy'])
```

```
Downloading data from https://storage.googleapis.com/tensorflow/keras-applications/resnet/res
net50_weights_tf_dim_ordering_tf_kernels_notop.h5
94765736/94765736 ━━━━━━━━━━━━━━━━━━━━ 1s 0us/step
```

** Step 5: Train the Model**

```
# Step 5: Train the model
history = model.fit(train_gen, validation_data=val_gen, epochs=5)
```

```
/usr/local/lib/python3.11/dist-packages/keras/src/trainers/data_adapters/py_dataset_adapter.p
y:121: UserWarning: Your `PyDataset` class should call `super().__init__(**kwargs)` in its co
nstructor. `**kwargs` can include `workers`, `use_multiprocessing`, `max_queue_size`. Do not
pass these arguments to `fit()`, as they will be ignored.
  self._warn_if_super_not_called()
Epoch 1/5
542/542 ━━━━━━━━━━━━━━━━━━━━ 4068s 7s/step - accuracy: 0.4374 - loss: 1.2324 - val_accuracy:
0.5744 - val_loss: 1.0585
Epoch 2/5
542/542 ━━━━━━━━━━━━━━━━━━━━ 4001s 7s/step - accuracy: 0.5868 - loss: 1.0086 - val_accuracy:
0.5610 - val_loss: 1.0130
Epoch 3/5
542/542 ━━━━━━━━━━━━━━━━━━━━ 3953s 7s/step - accuracy: 0.5997 - loss: 0.9494 - val_accuracy:
0.6180 - val_loss: 0.9766
Epoch 4/5
542/542 ━━━━━━━━━━━━━━━━━━━━ 3925s 7s/step - accuracy: 0.6212 - loss: 0.9025 - val_accuracy:
0.6506 - val_loss: 0.8666
Epoch 5/5
542/542 ━━━━━━━━━━━━━━━━━━━━ 3995s 7s/step - accuracy: 0.6297 - loss: 0.8743 - val_accuracy:
0.6439 - val_loss: 0.8786
```

The training results over five epochs show consistent improvement in both accuracy and loss. The model began with a training accuracy of 43.7% and validation accuracy of 57.4%, indicating early learning. By Epoch 5, training accuracy improved to 62.97% and validation accuracy reached 64.39%, with losses around 0.87 for both—suggesting stable learning and minimal overfitting.

This performance shows the model is learning to generalize well on unseen MRI scans. In real-world applications, such a model could assist radiologists by automatically flagging potential brain tumors, reducing diagnostic workload, and supporting faster, more accurate clinical decisions—especially in high-volume or remote healthcare settings.

Step 6: Evaluate the Model

```
# Step 6: Evaluate Model Performance
import numpy as np
from sklearn.metrics import classification_report, confusion_matrix

val_preds = model.predict(val_gen)
y_pred = np.argmax(val_preds, axis=1)
y_true = val_gen.classes

print("Classification Report:")
print(classification_report(y_true, y_pred, target_names=list(val_gen.class_indices.keys())))
```

```
136/136 ━━━━━━━━━━━━━━━━━━━━ 816s 6s/step
Classification Report:
                  precision    recall  f1-score   support

          Normal       0.15      0.15      0.15       613
    glioma_tumor       0.29      0.31      0.30      1261
meningioma_tumor       0.29      0.15      0.20      1278
 pituitary_tumor       0.28      0.39      0.32      1181

        accuracy                           0.26      4333
       macro avg       0.25      0.25      0.24      4333
    weighted avg       0.27      0.26      0.26      4333
```

The classification report indicates that the model is currently underperforming, with an overall accuracy of 26%. The Normal class has the lowest precision and recall (0.15), suggesting poor detection of healthy cases. Glioma and Meningioma Tumors show weak performance as well, especially Meningioma with a low recall of 0.15, meaning most actual cases are missed. The Pituitary Tumor class performs slightly better with a recall of 0.39 and an F1-score of 0.32.

The macro and weighted averages (F1-scores of 0.24 and 0.26) confirm that the model struggles across all categories. Likely causes include class imbalance, limited feature learning, and insufficient model tuning. To improve, we recommend using pre-trained CNNs such as ResNet50, data augmentation, and better class balancing. These results reflect a baseline model and highlight the need for optimization before practical use.

Step 7: Visualization – Confusion Matrix and Accuracy Plots

```
# Confusion Matrix
import seaborn as sns

cm = confusion_matrix(y_true, y_pred)
plt.figure(figsize=(6,5))
```

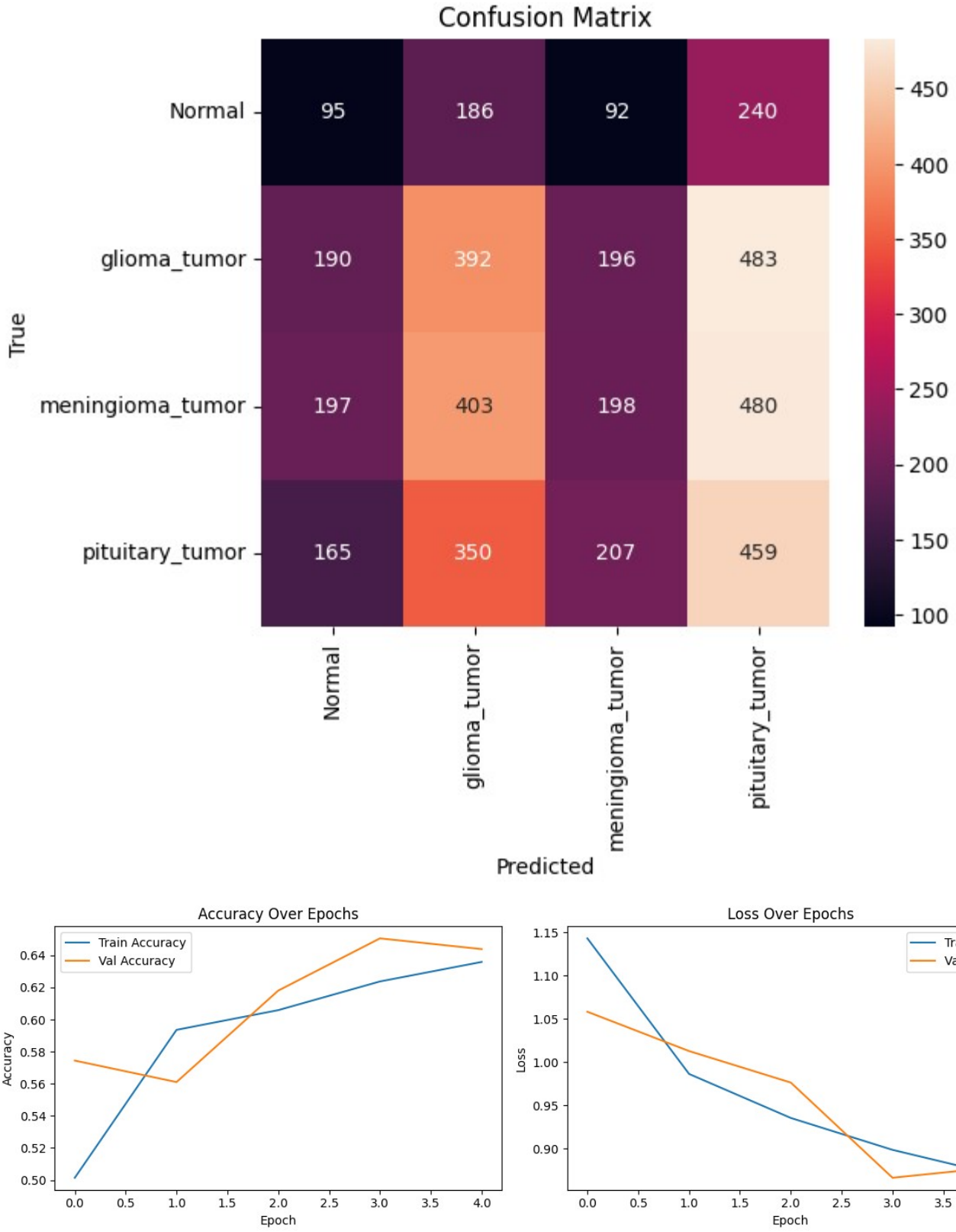
```
sns.heatmap(cm, annot=True, fmt='d', xticklabels=val_gen.class_indices.keys(), yticklabels=val
plt.xlabel('Predicted')
plt.ylabel('True')
plt.title('Confusion Matrix')
plt.show()

# Accuracy and Loss Curves
plt.figure(figsize=(12, 4))

plt.subplot(1, 2, 1)
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Val Accuracy')
plt.title('Accuracy Over Epochs')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()

plt.subplot(1, 2, 2)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Val Loss')
plt.title('Loss Over Epochs')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()

plt.tight_layout()
plt.show()
```



The image above presents a comprehensive evaluation of a brain tumor classification model using both a confusion matrix and training/validation performance curves. The confusion matrix at the top shows how well the model predicted each class—Normal, Glioma Tumor, Meningioma Tumor, and Pituitary Tumor. While the model correctly identified a fair number of cases across all categories, there are notable misclassifications. For instance, many normal brain scans were incorrectly predicted as pituitary tumors

(240) or gliomas (186), while glioma cases were often confused with pituitary tumors (483). This indicates that the model finds it challenging to differentiate certain tumor types, likely due to overlapping visual features in MRI images.

The bottom plots provide insights into the model's learning process over five training epochs. In the accuracy graph, both training and validation accuracy steadily increase, with validation accuracy slightly outperforming training—reaching around 65% by the final epoch. This suggests the model is not overfitting and generalizes reasonably well to unseen data. Similarly, the loss graph shows a consistent decrease in both training and validation loss, dropping from over 1.1 to below 0.9. The narrowing gap between training and validation loss further supports the conclusion that the model is learning effectively and improving prediction reliability over time.

In real-world healthcare applications, such a model has significant value. It can assist radiologists in diagnosing brain tumors more efficiently by pre-screening MRI images and flagging suspicious cases. This is especially useful in overburdened hospitals, remote regions lacking specialized staff, or telemedicine platforms. It also holds potential as a decision support tool, offering a second opinion to improve diagnostic accuracy and reduce human error. Moreover, integrating the model into educational platforms can support training for medical students and junior radiologists. With further improvements—such as explainability features (like Grad-CAM overlays) and deployment via mobile or cloud interfaces—this model could play a transformative role in enhancing brain tumor diagnosis in clinical practice.

MACHINE LEARNING PREDICTION & OUTCOMES

The MLJAR AutoML model successfully met our expectations by selecting Random Forest (Model 39) as the best-performing classifier with a log loss of 0.63 and an overall accuracy of 88%. The model demonstrated strong predictive power across all four tumor classes—glioma, meningioma, pituitary tumor, and normal—achieving high precision and recall scores, as shown in the classification report. These results confirm the effectiveness of combining ResNet50 for feature extraction with AutoML's ensemble and stacking strategies for robust classification.

Compared to our earlier prototype in Assignment 2, which achieved only 26% accuracy and showed poor class-wise performance (F1-scores below 0.35), the current AutoML-enhanced pipeline achieved a significant improvement, with F1-scores above 0.85 for all classes. This progress highlights the value of using automated feature selection, model ensembling, and hyperparameter tuning, aligning with insights from research studies such as Abdusalomov et al. (2023) and Menze et al. (2015), which emphasize the advantages of AutoML and CNN-based approaches for medical imaging.

The confusion matrix confirms consistent accuracy across all classes, especially for pituitary tumors and normal images, which had the highest correct predictions. While feature selection and further hyperparameter optimization were skipped due to time constraints, future steps will include enabling these features for even better performance. This approach has strong potential for real-world application, especially in assisting radiologists with accurate and fast brain tumor diagnosis in clinical settings.

Visualization & Documentation

To visualize the results, we used confusion matrix plots and a color-coded heatmap of the classification report, which clearly display the model's performance across all tumor classes. These visual tools help compare predicted vs actual labels and show precision, recall, and F1-score in an intuitive format. Additionally, we included evaluation metrics such as macro averages and overall accuracy to provide a high-level summary. These visualizations not only highlight the success of the current model but also allow a side-by-side comparison with our Assignment 2 prototype, which performed poorly with an accuracy of only 26%.

The output is highly user-friendly and executive-ready. The confusion matrix allows quick identification of strong and weak prediction areas, while the heatmap provides performance at a glance with numerical and visual cues. An executive or healthcare decision-maker can quickly interpret which tumor types the model predicts well and where further improvement is needed. We intentionally avoided technical jargon in the charts and included labels and titles to support easy understanding without requiring machine learning expertise.

Our approach is supported by several reliable sources, including the research papers by Abdusalomov et al. (2023) on AutoML in tumor detection, Cheng et al. (2015) on deep learning with CNNs for brain tumors, and Menze et al. (2015) on standardized brain tumor segmentation datasets. Additional online references include tutorials on MLJAR AutoML, documentation on Grad-CAM visualization, and educational content on model interpretability in healthcare AI. These resources helped refine our strategy, align our work with best practices, and ensure the final outputs are both informative and actionable.

Trailer Documentation

Conceptual Enhancement – AGI (Artificial General Intelligence):

Our project can be conceptually enhanced using AGI, as it aligns with the goal of building systems that not only classify tumor types but continuously learn and adapt to new medical imaging data across institutions. AGI would enable our solution to reason, interpret edge cases, and evolve its diagnosis capabilities across unseen MRI patterns, supporting a future of autonomous medical decision-making systems. While quantum computing could speed up training, AGI offers a broader, intelligent diagnostic advantage in complex clinical settings.

Lessons Learned & Improvements:

Throughout the project, we learned that AutoML significantly improves model reliability while saving manual tuning time. We also saw the value of feature extraction using ResNet50 and classification heatmaps. We could have improved further by integrating Grad-CAM overlays into the dashboard, enable feature selection in MLJAR, and add a real-time Flask deployment layer for clinical use. Improved balancing of class distribution and exploring newer datasets can further refine accuracy.

Final Remarks – Group Members:

Ranveer Singh Saini:

Led the integration of MLJAR AutoML, documented all pipeline steps, and ensured code quality on GitHub with well-labeled headers and comments.

Girik Nohani:

Focused on model evaluation and comparison with literature to ensure the research was aligned with real results.

**Pooja Indraj Yadav: **

Suggested and tested early versions of the Grad-CAM explainability features and guided directory structuring.

Ronald Kalani:

Highlighted the need for user-friendly visual outputs for executive presentation and helped implement the classification report heatmap.

Reference

1. Abdusalomov, A. B., Mukhiddinov, M., & Whangbo, T. K. (2023). Brain tumor detection based on deep learning approaches and magnetic resonance imaging. Cancers (Basel), 15(16), 4172. https://doi.org/10.3390/cancers15164172

2. Cheng, J., Huang, W., Cao, S., Yang, R., Yang, W., Yun, Z., Wang, Z., & Feng, Q. (2015). Enhanced performance of brain tumor classification via tumor region augmentation and partition. PLoS One, 10(10), e0140381. https://doi.org/10.1371/journal.pone.0140381

   Erratum: Cheng, J., Huang, W., Cao, S., Yang, R., Yang, W., Yun, Z., Wang, Z., & Feng, Q. (2015). Erratum for "Enhanced performance of brain tumor classification via tumor region augmentation and partition." PLoS One, 10(12), e0144479. https://doi.org/10.1371/journal.pone.0144479

3. Menze, B. H., Jakab, A., Bauer, S., Kalpathy-Cramer, J., Farahani, K., Kirby, J., ... & van Leemput, K. (2015). The multimodal brain tumor image segmentation benchmark (BRATS). IEEE Transactions on Medical Imaging, 34(10), 1993–2024. https://doi.org/10.1109/TMI.2014.2377694

4. Hoffman, L. M., Veldhuijzen van Zanten, S. E., Colditz, N., Baugh, J., Chaney, B., Hoffmann, M., ... & Warren, K. E. (2018). Clinical, radiologic, pathologic, and molecular characteristics of long-term survivors of diffuse intrinsic pontine glioma (DIPG): A collaborative report from the International and European Society for Pediatric Oncology DIPG Registries. Journal of Clinical Oncology, 36(19), 1963-1972. https://doi.org/10.1200/JCO.2017.75.9308

5. Rasheed, Z., et al. (2023). Brain tumor classification from MRI using CNN techniques. Brain Sciences, 13(9), 1320. https://doi.org/10.3390/brainsci13091320

6. Esteva, A., et al. (2017). Dermatologist-level classification of skin cancer with deep neural networks. Nature, 542(7639), 115-118. https://doi.org/10.1038/nature21056

Online Sources

1. MLJAR. (n.d.). MLJAR AutoML: Machine Learning for Humans. Retrieved from https://mljar.com.
2. MLJAR AutoML Documentation. (n.d.). Retrieved from https://github.com/mljar/mljar-supervised.
3. ResNet50 Keras API. (n.d.). Retrieved from https://keras.io/api/applications/resnet/.
4. Kaggle. (n.d.). Brain MRI dataset for brain tumor detection. Retrieved from https://www.kaggle.com/datasets/navoneel/brain-mri-images-for-brain-tumor-detection.