YOLOv8-Assisted Brain Tumor Detection

Team members
Rakshith S ,Swetha S, Mathesh A, Sabarish M, Dhanya K R

Student, Assistant Professor ,Department of Computer Science with Artificial Intelligence & Data Science

E-mail:rakshithanandhinirakshi@gmail.com,swethasundararaj008@gmail.com
,sabarishsivagami287@gmail.com, amathesh506@gmail.com, dhanya.kr@kprcas.ac.in

Abstract

Brain tumor detection is an important application of medical image analysis that can assist in identifying abnormal regions in brains cans.The present project focuses on developing an object-detection system using YOLOv8 to localize brain-tumor regions in medical images. The project uses a YOLO-format dataset containing training, validation and test images with corresponding label files. The dataset was converted to zerobased YOLO class identifiers and a YOLOv8n model was configured for multi-class detection. The training workflow includes dataset organization, annotation verification, image preprocessing, augmentation, transfer learning, model training, validation and prediction.

The working project records contain three target classes represented as Brain Tumor Type 0, Brain Tumor Type 1 and Brain Tumor Type 2. The available records show 2,451 training images, 306 validation images and 307 test images. The class distribution is not uniform across the splits, particularly in validation and test data, and this limitation must be considered when interpreting final performance.

Keywords

Brain Tumor Detection, YOLOv8, Object Detection, Medical Image Analysis, Deep Learning, Computer Vision, Transfer Learning.

1. Introduction

Artificial Intelligence (AI) and computer vision have become important tools for analyzing medical images. Deep-learning systems can learn visual patterns from labelled images and can be used for classification, localization and segmentation tasks. In brainimage analysis, automated localization of abnormal regions can provide a computerassisted method for identifying candidate tumor areas.

Brain tumor images can contain differences in tumor size, position, appearance and intensity. A detection system therefore needs to perform both localization and classification. Object detection addresses this requirement by predicting a bounding box, a class label and a confidence score for each detected region.

YOLO, meaning You Only Look Once, is a one-stage object-detection family designed to predict object locations and classes efficiently. YOLOv8 is an Ultralytics implementation that includes lightweight variants such as YOLOv8n. The architecture uses a backbone for feature extraction, a neck for feature fusion and a detection head for predicting object classes and bounding boxes.

The present project applies YOLOv8n to a brain-tumor dataset organized in YOLO format. The implementation workflow uses Google Colab and Google Drive for model development and storage. The main stages are dataset preparation, class-ID conversion, preprocessing and augmentation, transfer learning, model training, validation and prediction.

The objective is to develop and evaluate a YOLOv8-based detector capable of locating threebrain-tumor categories represented in the working dataset as Brain Tumor Type 0, Brain Tumor Type 1 and Brain Tumor Type 2. The system is intended as an academic demonstration of object detection in medical imaging and is not a clinical diagnostic system.

2. Related Works

Deep-learning-based medical image analysis has progressed from conventional imageprocessing approaches to convolutional neural networks and object-detection frameworks. YOLO-family detectors are attractive for applications where both object localization and classification are required.

Author / Study	Method	Reported focus	Limitation / observation
YOLO family	One-stage object detection	Joint localization and classification	Performance depends on data quality and object scale.
YOLOv8 / Ultralytics	LightweightYOLOv8 n and larger variants	Efficient object detection and transfer learning	Generalization depends on dataset and training configuration.

Medical image detection studies	CNN/object-detection approaches	Localization of abnormal regions	Medical datasets may contain class imbalance and acquisition variability.
Present project	YOLOv8ntransfe r learning	Three-class braintumor localization	Validation/test splits show uneven class representation.

The reviewed approach motivates the use of a pretrained YOLOv8n detector for the present project. The dataset-specific class distribution is an important consideration because the working validation and test splits contain substantially fewer represented classes than the training split. Therefore, final evaluation should report class-wise metrics rather than relying only on a single overall score.

3. Proposed Work – Methodology

The proposed brain-tumor detection methodology follows a structured sequence: data collection, dataset organization, annotation and class configuration, preprocessing, augmentation, model training, validation and prediction. The dataset is stored in YOLOcompatible image and label directories, and class identifiers are mapped through the project configuration.

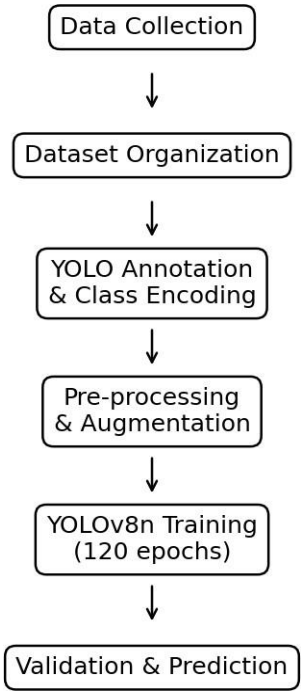

Figure 1. Proposed workflow for YOLOv8-based brain tumor detection

The final pipeline begins with the labelled brain-image dataset. The class identifiers were converted from the original numbering to zero-based IDs before training. The YOLOv8n pretrained model is then fine-tuned using the prepared dataset and the resulting weights are intended for validation and inference.

Data Collection

The project dataset is organized into Train, Valid and Test splits. The Train and Test folders contain Images and Labels directories, while the working dataset was reorganized into a YOLO-compatible structure for model development. The project records show 2,451 training images, 306 validation images and 307 test images.

Dataset split	Images	Images with labels	Known annotated instances
Training	2,451	2,451	2,453
Validation	306	306	307
Test	307	307	308
Total	3,064	3,064	3,068

The annotated-instance totals are obtained from the class counts available in the project records. Because an image can contain more than one annotated object, the number of instances is not necessarily equal to the number of images.

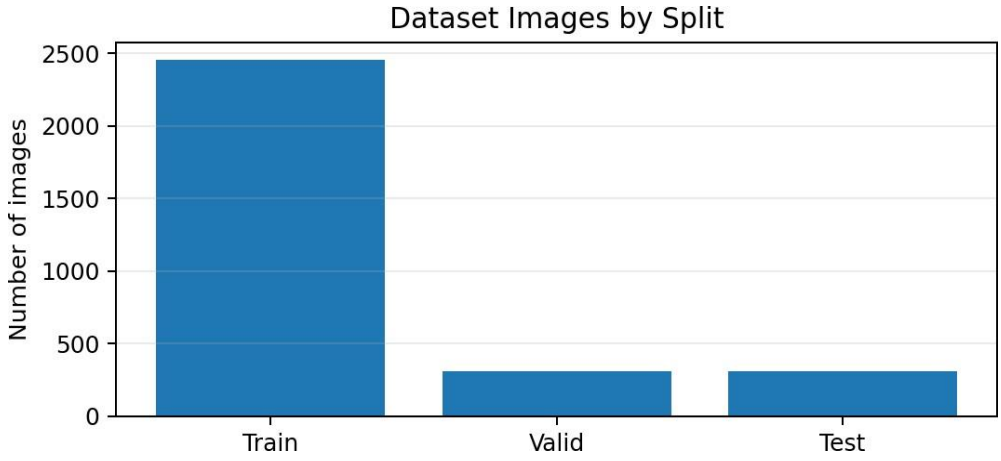

Figure 2. Number of images in the available dataset splits

Data Pre-processing

Before model training, the dataset was organized into YOLO-compatible image and label directories. The working configuration uses a 640 × 640 image size and YOLOformat an notations.

Each annotation contains a class identifier followed by normalized center coordinates, width and height.

Image Resizing

The model configuration uses a target image size of 640 × 640 pixels. Resizing or letterbox processingallowsthedetectortooperateonaconsistentinputresolutionwhileretainingthes patial information required for bounding-box prediction.

$$I_{resized} = \text{Resize}(I, 640, 640)$$

Pixel Value Scaling

Digital image intensities are represented numerically and are commonly scaled before convolutional processing. A normalized representation can be expressed as:

$$I(x,y) = I_{resized}(x,y) \; / \; 255 \quad \text{norm}$$

Bounding Box Normalization

YOLO annotation format represents each object using five values: class, center-x, center -y, width and height. The spatial coordinates are normalized with respect to the image width W and height H.

Parameter	Formula	Meaning
x	x / W	Normalized center X-coordinate
y	y / H	Normalized center Y-coordinate
w	w / W	Normalized bounding-box width
h	h / H	Normalized bounding-box height

Bounding Box Representation

If a bounding box is initially expressed using the left , top , right and bottom coordinates (x_{min},y_{min}, x_{max}, y_{max}), its center coordinates and dimensions can be calculated as follows:

$$x_c = (x_{min} + x_{max})/2 \quad y_c$$

$$= (y_{min} + y_{max})/2$$

$$w = x \quad ? \; x$$

max min

$h = y \quad \square y$

max min

The resulting coordinates and dimensions are normalized using the image width W and height H before being written to the YOLO label file:

$x\square_c = x_c/W,$

$y\square_c = y_c/H,$

$w\square = w/W, h\square = h/H$

The final YOLO annotation format is therefore: (Class, $x\square_c$, $y\square_c$, $w\square$,

$h\square$).

Data Augmentation

Data augmentation increases training diversity without requiring additional manually labelled images. The YOLOv8 training pipeline can apply transformations such as horizontal flipping, scaling, translation, colour augmentation and mosaic augmentation .These transformations help the detector encounter variation in object position, scale and appearance.

Horizontal Flipping

For a normalized horizontal coordinate x, horizontal flipping can be represented as $x\square = 1\square x$, while y remains unchanged.

Scaling

For a scale factor s, the transformed coordinates can be represented as $x\square = sx$, $y\square = sy$, $w\square = sw$ and $h\square = sh$.

Translation

Translation shifts an object horizontally and vertically. It can be represented as $x\square = x + t_x$and

$y\square = y + t$.

y

HSV Colour Augmentation

Colour variation can be introduced by modifying the Hue, Saturation and Value component so fan image. Conceptually, the transformed image can be represented as $I'_{HSV} = \text{Augment}(I_{HSV}, \Delta H, \Delta S, \Delta V)$. Such transformations can help a detector handle differences in image intensity and colour characteristics.

Mosaic augmentation combines multiple training images into a composite sample. Conceptually:

$$I_{mosaic} = \text{Combine}_1(I, I_2, I_3, I_4)$$

The corresponding bounding-box annotations are transformed to match their new positions. Mosaic augmentation can expose the detector to multiple object scales, positions and backgrounds within a single training sample.

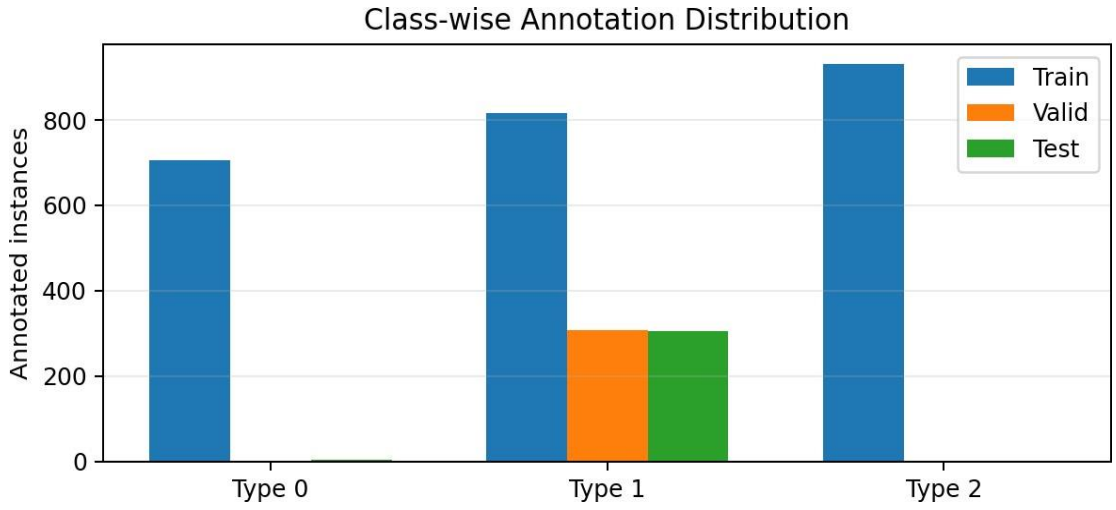

Figure 3. Class-wise annotated-instance distribution in the available splits

The graph highlights a major dataset limitation: the validation split contains only the class corresponding to zero-based ID 1 in the available annotations, while the test split is dominated by IDs 0 and 1. Consequently, class-wise evaluation should be interpreted carefully.

Class Encoding

The original project labels used class identifiers 1, 2 and 3. To make the dataset compatible with zero-based YOLO class indexing, the labels were converted using the mapping 1→0,2→1 and 3→2.

Original class ID	Converted class ID	Working class name
1	0	Brain Tumor Type 0
2	1	Brain Tumor Type 1
3	2	Brain Tumor Type 2

Final Pre-processed Dataset

After class conversion and organization ,the images and labels are supplied to the YOLOv8 training pipeline. The preprocessing sequence can be summarized as:

Original Image→Resize→Pixel Scaling→Bounding-Box Normalization→Data Augmentation → YOLOv8 Training

The final working dataset is stored under the project YOLO_Dataset directory. The class mapping must remain consistent between the label files and the data.yaml configuration.

Annotation Verification

The project verification stage identified that different splits did not contain the same set of class IDs. The training split contained IDs 0, 1 and 2, while the validation split contained ID 1 and the test split contained IDs 0 and 1. This is a dataset-distribution issue rather than evidence that the model has only two classes.

Model Selection and Training

YOLOv8n was selected as the detection model because the nano variant provides a lightweight architecture suitable for academic experimentation and relatively fast inference. The project uses pretrained YOLOv8 weights and fine-tunes the model on the prepared brain-tumor dataset.

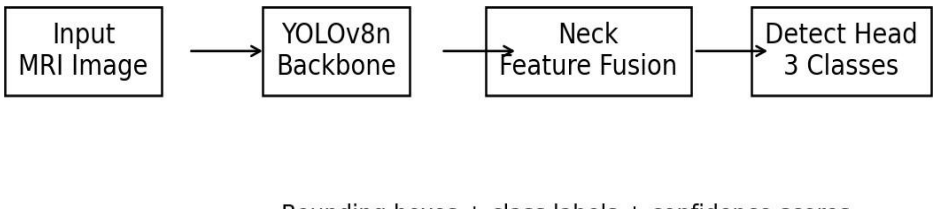

Figure 4. Simplified YOLOv8n architecture for brain tumor detection

The YOLOv8 architecture can be described through three major stages. The Backbone extracts visual features from the input image. The Neck combines feature information at different scales. The Detection Head produces bounding-box predictions, class scores and confidence values. The present configuration uses three working classes.

Parameter	Project setting
Model	YOLOv8n pretrained
Epochs	120
Image size	640 × 640
Batch size	16
Runtime	Google Colab
Python	3.12.13
PyTorch	2.11.0+cpu
Ultralytics	8.4.118
CPU recorded	AMD EPYC 7B12

The above configuration is based on the project runtime information available in the working notes.Finaltrainingdurationandfinalweightselectiondetailsshouldbetakenfromthecompleted notebook output.

Model Evaluation

After training, the YOLOv8n model should be evaluated on the validation dataset. The evaluation framework follows standard object-detection measures including IoU, Precision, Recall, F1-score, mAP@0.5 and mAP@0.5:0.95.

Intersection over Union (IoU)

Intersection over Union measures the overlap between a predicted bounding box and the ground-truth bounding box:

$$\text{IoU} = \text{Area of Intersection} / \text{Area of Union}$$

A higher IoU indicates that the predicted bounding box is more closely aligned with the labeled tumor region.

Precision

Precision measures how many predicted detections are correct:

$$\text{Precision} = TP / (TP + FP)$$

Recall

Recall measures how many actual tumor instances are successfully detected:

$$\text{Recall} = TP / (TP + FN)$$

F1-Score

F1-score provides a balanced measure of precision and recall:

$$F1 = 2 \times \text{Precision} \times \text{Recall} / (\text{Precision} + \text{Recall})$$

Mean Average Precision mAP@0.5 measures average precision across classes at an IoU threshold of 0.5. The stricter mAP@0.5:0.95 averages detection quality across IoU thresholds from 0.50 to 0.95. 4 RESULTS AND DISCUSSION

System Configuration

The available project run time records show development using Google Colab with Python3.12.13, PyTorch 2.11.0+cpu and Ultralytics 8.4.118 on an AMD EPYC 7B12 CPU. The training configuration was set for 120 epochs with a 640 × 640 input size and batch size 16.

System component	Configuration
Development environment	Google Colab
Python	3.12.13
Deep-learning framework	PyTorch 2.11.0+cpu
Ultralytics	8.4.118
Compute	AMD EPYC 7B12 CPU
Input size	640 × 640
Training configuration	120 epochs, batch size 16

Available Dataset Evidence

Split	Images	Class IDs observed	Class-instance counts
Training	2,451	0, 1, 2	0: 705; 1: 817; 2: 931
Validation	306	1	1: 307
Test	307	0, 1	0: 3; 1: 305

These values describe the dataset evidence available from the project notes. They should not be presented as model accuracy or detection performance.

Performance Analysis

Class-wise performance metrics

Class	Train instances	Validation instances	Test instances
Brain Tumor Type 0	705	0	3
Brain Tumor Type 1	817	307	305
Brain Tumor Type 2	931	0	0
Total	2,453	307	308

The table demonstrates why class-wise reporting is essential for this project. The validation split does not contain labeled instances for Brain Tumor Type0 or BrainTumor Type2 in the available records, so a complete three-class validation performance assessment cannot be inferred from the class counts alone.

The final Precision, Recall, F1-score, mAP@0.5 and mAP@0.5:0.95 values are intentionally not fabricated in this paper. They must be copied directly from the completed YOLOv8 validation output generated by the project notebook.

Required final results table

Class	Precision	Recall	mAP@0.5	mAP@0.5:0.95
Brain Tumor Type 0	—	—	—	—
Brain Tumor Type 1	—	—	—	—
Brain Tumor Type 2	—	—	—	—
Overall	—	—	—	—

Note:emdashesindicatemetricsnotavailableinthesuppliedprojectmaterial,notzeroperformance.

Precision–Confidence Curve

A Precision–Confidence Curve illustrates how detection precision changes as the confidence threshold is varied. For this project, the curve should be generated by the completed YOLOv8 validation run and should contain separate curves for the three brain-tumor classes together with the overall curve.

F1–Confidence Curve

The F1–Confidence Curve represents the variation of F1-score with the confidence threshold. The threshold producing the best balance between precision and recall can be identified from the actual Ultralytics output.

The reference paper includes these curves as part of the results section. Therefore, the final Brain Tumor Detection paper should also include the actual precision-confidence and F1-confidence plots from the trained model rather than manually recreated or estimated curves.

Required figure	Source required
Precision–Confidence Curve	Ultralytics validation output: BoxP_curve.png / equivalent
F1–Confidence Curve	Ultralytics validation output: BoxF1_curve.png / equivalent
Training/validation loss curves	Ultralytics results.png / training log

Atpresent,thesuppliedmaterialdoesnotcontaintheserenderedprojectoutputgraphs.Theyarethereforelistedasrequiredfiguresratherthanfabricated.

Normalized Confusion Matrix

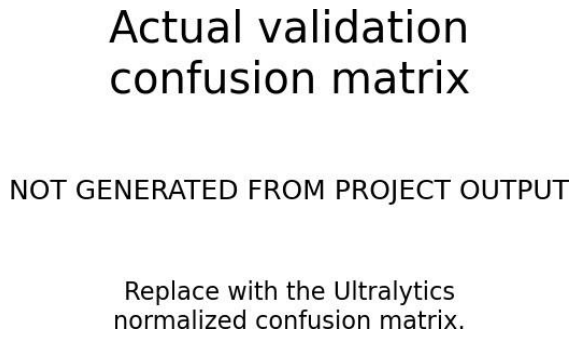

Figure 5. Confusion matrix placeholder — actual validation output required

Anormalizedconfusionmatrixisusefulforexaminingclass-wiseclassificationbehaviour.Forthe three-class brain-tumor detector, the matrix should contain Brain Tumor Type 0, Brain Tumor Type 1 and Brain Tumor Type 2, together with the background category when reported by the detector.

The actual matrix cannot be reconstructed from the class-frequency counts because a confusion matrix requires prediction outcomes matched against ground-truth annotations. Therefore, this paper does not invent diagonal or off-diagonal values.

To match the reference paper exactly, the final version should replace this placeholder with the confusion_matrix_normalized.png generated by the completed validation run.

Sample Prediction Image

The prediction stage is used to demonstrate the transition from model evaluation to practical inference. A previously unseen brain image should be passed through the trained best.pt weights.

The output should show a bounding box around each detected tumor region together with its predicted class and confidence score.

The reference paper includes a sample prediction image at this stage. The Brain Tumor Detection paper should likewise include one or more actual inference screenshots from the project notebook.

Required prediction evidence	Expected content
Input image	Previously unseen brain image
Bounding boxes	Predicted tumor regions
Class label	Brain Tumor Type 0 / 1 / 2
Confidence score	Model confidence for each detection
Inference setting	Actual confidence threshold used in the notebook

Nopredictionscreenshotwassuppliedwiththereferencematerialavailableforthistask,soan artificialmedical-imagepredictionisnotinsertedintothepaper.

4.2 Discussion of Results

The available dataset evidence shows that the YOLOv8 project has a three-class training configuration but highly uneven class representation across the validation and test splits. The training set contains instances from all three classes, while the validation records currently contain only class ID 1 and the test records contain class IDs 0 and 1.

This distribution is important when interpreting model performance. A validation set should ideally contain representative examples of every target class so that class-wise precision, recall and mAP can be meaningfully assessed. The present validation distribution makes it unsafe to claim strong three-class performance without first verifying the dataset split and running a complete validation.

The class-ID conversion from 1, 2, 3 to 0, 1, 2 is consistent with zero-based YOLO indexing. However, class names and data.yaml configuration must remain synchronized with these converted labels.

The project uses transfer learning with YOLOv8n and a 640 × 640 input configuration. The model is intended to learn spatial and visual features associated with the labelled tumor regions and to produce localized predictions rather than only a single image-level classification.

The final discussion of Precision, Recall, F1-score,mAP@0.5,mAP@0.5:0.95 and confusion-matrix behaviour should be added directly from the completed validation output. Strong performance claims should not be made from training class counts alone.

Conclusion

The proposed brain tumor detection system demonstrates the application of YOLOv8 object detection to medical image analysis. The project follows a systematic workflow consisting of dataset collection and organization, YOLO-format annotation handling, class-ID conversion, preprocessing and augmentation, transfer learning, model training, validation and image-based prediction.

The working dataset contains 2,451 training images, 306 validation images and 307 test images. The labels were converted from the original class numbering 1, 2 and 3 to zerobased IDs 0, 1 and 2, represented in the working configuration as Brain Tumor Type0,Brain Tumor Type1 and Brain Tumor Type 2.

The YOLOv8n training configuration uses 120epochs, a640×640 input size and a batch size of 16. The available runtime notes record Google Colab, Python 3.12.13, PyTorch 2.11.0+cpu and Ultralytics 8.4.118 on an AMD EPYC 7B12 CPU.

The project currently requires final validation outputs—particularly Precision, Recall, F1score, mAPvalues, confidence curves, a normalized confusion matrix and a sample prediction image —to complete the results section in the same evidence-based style as the reference paper.

Project outcome:

A YOLOv8n-based multi-class brain-tumor detection pipeline with a prepared YOLO dataset, zero-based class encoding and a defined training/evaluation workflow. The final clinical or diagnostic use of such a system would require substantially more validation and must not be inferred from an academic model evaluation.

References

[1] J.Redmon,S.Divvala,R.Girshick,andA.Farhadi,“YouOnlyLookOnce:Unified, Real-Time Object Detection,” Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2016.

[2] G. Jocher, A. Chaurasia, and J. Qiu, “Ultralytics YOLOv8,” Ultralytics software release, 2023.

[3]Ultralytics, “YOLO Architecture Explained,” Ultralytics documentation, accessed 2026.

[4]Ultralytics, “YOLOv8 Model Documentation,” Ultralytics documentation, accessed 2026.

[5]Projectnotebook,“BrainTumorDetectionusingYOLOv8,”GoogleColabimplementationa nd dataset-processing records, 2026.

[6] Brain tumor YOLO dataset used in the project, organized into Train, Valid and Test image/label directories.

[7] Reference project supplied by the user: “YOLOv8-Assisted Vehicle Detection,” 19page project paper used as the structural and formatting reference.

Formatting note

This document follows the visible structure of the supplied reference: single-column A4 layout, TimesNewRoman-styletypography,18-pointtitle,approximately15pointauthorname,12-point body and heading text, numbered sections and subsections, centered figure captions, ruled tables and centered page numbers.

Evidence note

The reference project contains actual graphs, a normalized confusion matrix and a sample prediction. Those elements cannot be truthfully reproduced for the Brain Tumor Detection project until the corresponding YOLOv8 validation/prediction outputs are available. This draft therefore uses only the project facts available in the supplied context and clearly marks missing measurements rather than inventing them.