# BRAIN TUMOR TISSUE SEGMENTATION AND SURVIVABILITY PREDICTION IN BRATS 2020 MRI SCANS

*Divya Sathiyamoorthy*

Columbia University
ds4221@columbia.edu

*Smriti Vaidyanathan*

Columbia University
sv2785@columbia.edu

## ABSTRACT

Brain tumors, characterized by abnormal tissue growth, present significant diagnostic and treatment challenges. Early detection and precise diagnosis, primarily through Magnetic Resonance Imaging (MRI), are critical for improving patient outcomes. However, conventional methods relying on manual segmentation and expert interpretation are time-intensive and prone to variability. This project explores a deep learning-based approach leveraging the BraTS 2020 dataset to address two key tasks: tumor segmentation and survival prediction.

The BraTS 2020 dataset, comprising multimodal MRI scans with detailed tumor sub-region annotations, was used to develop two models: a UNet-based architecture for segmentation and a ResNet50-based model for survival classification. The UNet model achieved an accuracy of 89.86% for segmentation, with the highest accuracy observed in identifying the enhancing tumor core (ET) region. The ResNet50 model, trained on ground truth masks to classify into survival day bins, achieved an accuracy of 65.15%. When combined, the UNet and ResNet50 models achieved an impressive overall accuracy of 86.92% after filtering empty masks and 98.31% after taking the majority vote across a volume.

This work highlights the potential of deep learning to automate MRI analysis, providing a consistent, scalable solution for brain tumor diagnosis and survival estimation.

## 1. INTRODUCTION

Brain tumors, characterized by abnormal growth of tissue within the brain, pose a significant threat when left untreated, as they can lead to impaired brain function through swelling and increased intracranial pressure [1], [2]. Early detection of brain tumors greatly improves survival outcomes, yet accurate diagnosis remains challenging due to variability in brain imaging quality and reliance on expert interpretation. Additionally, brain tumors have the potential to metastasize to other regions of the body, further complicating treatment.

Magnetic Resonance Imaging (MRI) plays a crucial role in brain tumor diagnosis, offering a non-invasive and radiation-free means of visualization. MRIs provide high-resolution images from multiple perspectives, often creating detailed 3D representations of the brain [1]. However, the accuracy of diagnosis based on MRI images is influenced by factors such as the resolution of the images and the expertise of radiologists interpreting them. Consequently, diagnosis can be time-consuming and inconsistent, which may delay effective treatment. While tissue biopsies provide more definitive diagnostic information, they are invasive, painful, and subject to sampling errors [1].

Given these limitations, improving the accuracy of tumor detection from MRI images through automated techniques holds great promise. Computer vision, particularly using machine learning (ML) and deep learning (DL), offers an opportunity to enhance MRI-based diagnosis by addressing three primary tasks: tumor detection, tumor segmentation, and tumor classification. Detection involves determining the presence of a tumor, segmentation entails defining its size, shape, and composition, and classification identifies the specific type of tumor. By automating these processes, computer vision can facilitate more consistent, timely, and accurate diagnostic outcomes, leading to better patient care [2].

In addition to these applications, understanding the severity and survivability of brain tumors is crucial. Predicting tumor survivability from MRI scans alone could expedite the treatment process and provide insights into the factors that influence brain cancer outcomes.

To address these concerns, this project aims to use MRI brain scans to effectively segment brain tissue for easier analysis and then predict survivability on the resulting segmentation using DL techniques. Manually segmenting, as shown in Fig. 2, varies greatly depending on a person-to-person basis. A deep learning approach would help to standardize the outcome. In addition, by segmenting relevant parts of the brain tissue, this approach may help focus survivability predictions on the regions most likely to impact them. Moreover, it can provide insights into the specific types of tissue that contribute to survivability outcomes.

## 2. RELATED WORK

The use of ML and DL in clinical diagnostics has shown significant potential. For instance, the Biomind system demon-

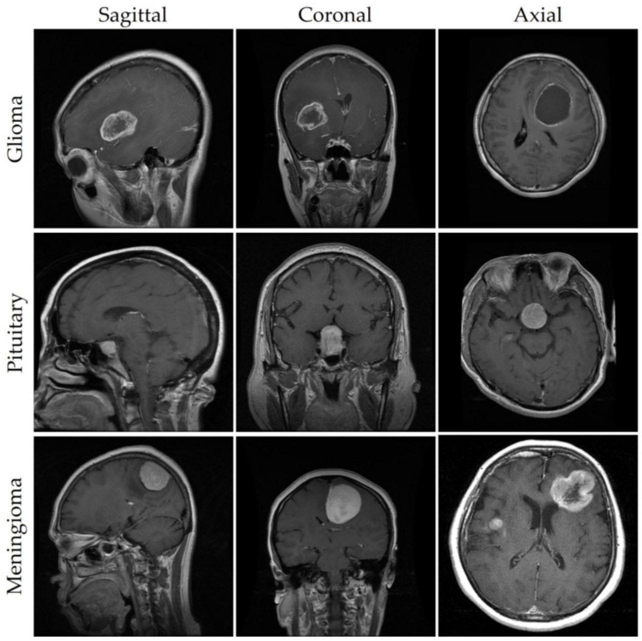

**Fig. 1**: Sagittal, Coronal, and Axial views of a sample of MRI images of three different types of brain tumor [1]

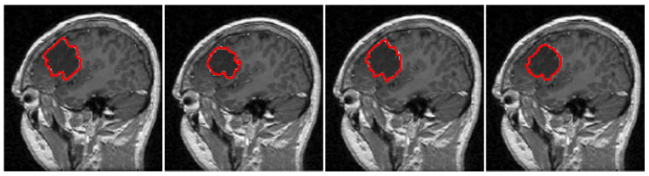

**Fig. 2**: Manual segmentation of glioma on the same MRI performed by four different experts [1]

strated higher accuracy (11% improvement) and faster analysis (15 minutes quicker) compared to manual diagnosis by 15 radiologists during a brain tumor diagnosis competition [2]. Furthermore, numerous publicly available MRI datasets, such as AANLIB, ADNI1, the BraTS challenges, and the Allen Brain Atlas, have contributed to advancing computer vision research in brain tumor detection and other medical applications [2].

Traditional ML methods for tumor detection, such as Support Vector Machines (SVMs), Artificial Neural Networks (ANNs), and k-Nearest Neighbors (KNNs), rely on extracting features from MRI images and employing dimensionality reduction techniques, such as Principal Component Analysis (PCA), to identify abnormalities. Tumor segmentation, on the other hand, often requires image intensity analysis combined with edge detection methods, and may incorporate both traditional and DL techniques, including Convolutional Neural Networks (CNNs), to achieve optimal accuracy and efficiency [2].

Recent advancements have demonstrated the effectiveness of fine-tuned deep learning models for tumor diagnosis. Abdusalomov et al. employed the YOLOv7 model to detect brain tumors with high accuracy (99.5%) by applying a multi-stage image enhancement approach and using curated MRI datasets. The success of this study exemplifies the potential of deep learning-based computer vision systems in delivering highly accurate and efficient brain tumor detection, which is critical for prompt and effective treatment.

## 3. DATASET

This project utilized the BraTS 2020 dataset hosted on Kaggle, originating from the BraTS 2020 challenge [3],[4] . The BraTS 2020 dataset provides multimodal pre-operative MRI scans collected from multiple institutions. These scans include native (T1), post-contrast T1-weighted (T1Gd), T2-weighted (T2), and T2-FLAIR sequences. All scans are pre-processed, including co-registration, interpolation to a resolution of 1 mm³, and skull-stripping [5], [6], [7], [8], [9].

Manually segmented tumor sub-regions are included, encompassing the enhancing tumor core (ET, label 4), peritumoral edema (ED, label 2), and necrotic/non-enhancing tumor core (NCR/NET, label 1). These annotations, verified by neuro-radiologists, serve as ground truth data for training and validating segmentation models.

The Kaggle-hosted dataset provides both the segmentation masks and the corresponding images in `.h5` file format within a `.zip` file. The dataset contains volumes (which correspond to patients) and Axial slice images within each volume. In total, the dataset contains 57,195 `.h5` files, with each file containing a `numpy ndarray` for images of size $(240, 240, 4)$ and a corresponding mask of size $(240, 240, 3)$. Additionally, survival data, including overall survival (OS) information, patient age, and resection status, is provided in a `.csv` file.

BraTS 2020 introduced four main tasks: (1) tumor subregion segmentation, (2) overall survival (OS) prediction based on clinical data, (3) distinguishing between pseudoprogression and true recurrence, and (4) uncertainty estimation in segmentation. This project focuses on combining tasks (1) and (2) by generating segmentation masks from the image channels and subsequently utilizing the predicted segmentation to improve survival prediction performance.

## 4. METHODOLOGY

### 4.1. Segmentation Model

#### *4.1.1. Data Pre-Processing*

The data preprocessing pipeline involved loading images and their corresponding masks from `.h5` files and converting

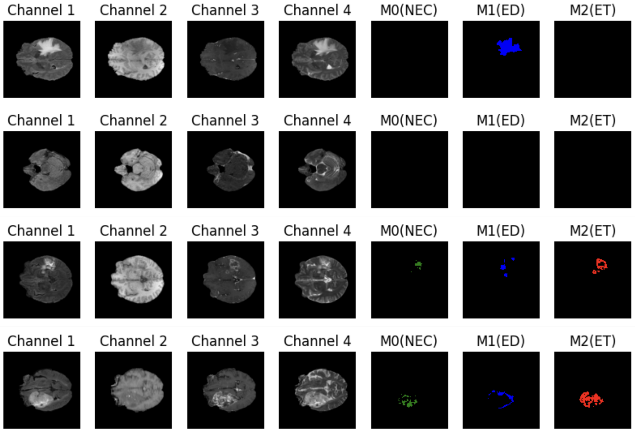

**Fig. 3**: Example Image Data from BraTS 2020 with Corresponding Masks

them into tensors, resulting in a total of 57,195 samples for each. A fourth (0th) channel was added to the masks to represent the background, which was activated whenever none of the other three channels were. Both the images and masks were resized to $256 \times 256$ pixels using bilinear interpolation to match the input dimension of ResNet18. The four image channels and the four (three + background) mask channels were then stacked into tensors respectively, meaning a single image had a size of $(4, 256, 256)$ and a single mask also had a size of $(4, 256, 256)$. Subsequently, the data was split into training, validation, and testing subsets using an 80-10-10 ratio. This resulted in 45,760 samples for training, 5,760 for validation, and 5,760 for testing. The data subsets were then organized into PyTorch dataloaders to facilitate efficient batch processing during model training and evaluation.

### *4.1.2. Model Architecture*

For segmentation, this project uses the U-Net architecture, a convolutional neural network specifically designed for biomedical image segmentation tasks [10]. U-Net combines an encoder, decoder, and segmentation head for its architecture. The encoder, based on a ResNet-18 backbone, processes the input through a series of layers that reduce spatial dimensions while increasing feature depth. Specifically, the encoder includes four stages, each with multiple convolutional layers, batch normalization, and ReLU activations, followed by downsampling via max-pooling or strided convolutions. The decoder mirrors this structure with four stages, gradually restoring spatial resolution using upsampling layers and combining features from the encoder at corresponding levels through skip connections [10].

The segmentation head of the U-Net was customized to better suit the specific task. Instead of the original head, a new series of layers was added: a $3 \times 3$ convolutional layer reduces the feature depth from 16 to 8, followed by a ReLU activation; another $3 \times 3$ layer reduces the depth from 8 to 4, followed by another ReLU; and finally, a $1 \times 1$ convolution maps these features to the four output classes.

To train the model, the original authors relied on extensive data augmentation and used stochastic gradient descent with cross-entropy loss to optimize the network. The UNet model, for this project, utilized pretrained weights for the ResNet-18 encoder, which were trained on ImageNet, to leverage its ability to recognize general features from large-scale image datasets. [10].

### *4.1.3. Loss Function*

The *Dice Loss Coefficient* is a loss function commonly used in image segmentation tasks to evaluate the similarity between predicted and ground truth masks. It is derived from the *Dice Similarity Coefficient*, which measures the overlap between two sets and scales it to a range of 0 to 1, where 1 indicates perfect overlap [11]. For segmentation, this concept is extended to multi-channel or multi-class masks.

In this implementation, the predicted mask $\hat{M}$ (of shape $[B, C, H, W]$, where $B$ is the batch size, $C$ is the number of classes, and $H$ and $W$ are the spatial dimensions) is passed through a softmax function along the channel dimension to obtain class probabilities. The ground truth mask $M$ is cast to a float type for computation. Both $\hat{M}$ and $M$ are then flattened into vectors of shape $[B, C, HW]$ for each batch.

The Dice coefficient for each class $c$ is computed as:

$$D_c = \frac{2 \sum_i \hat{M}_{i,c} M_{i,c} + \epsilon}{\sum_i \hat{M}_{i,c} + \sum_i M_{i,c} + \epsilon}$$

where $\hat{M}_{i,c}$ and $M_{i,c}$ represent the predicted and true values for class $c$ at pixel $i$, and $\epsilon$ (denoted as the `smooth_coef`) is a small smoothing constant to prevent division by zero.

To account for the class imbalance in the dataset, where most masks are dominated by the background (black pixels), class weights of $[0, 1, 1, 1]$ were applied. This ensures that an all-black mask does not contribute to a score that would artificially rewards the model for predicting the majority class. Specifically, the background class (class 0) is excluded from the Dice score computation by setting its weight to zero, while all other classes are equally weighted. The adjusted Dice coefficient is then computed as:

$$D_c^{\text{weighted}} = w_c \cdot \frac{2 \sum_i \hat{M}_{i,c} M_{i,c} + \epsilon}{\sum_i \hat{M}_{i,c} + \sum_i M_{i,c} + \epsilon}$$

where $w_c$ is the weight for class $c$. The mean Dice coefficient across all classes is then calculated and the Dice Loss is defined as:

$$\text{Dice Loss} = 1 - \frac{1}{\sum_{c=1}^{C} w_c} \sum_{c=1}^{C} D_c^{\text{weighted}}$$

The Dice coefficient is a good metric for segmentation tasks because it balances precision and recall, providing a robust measure of overlap between predicted and ground-truth masks [11]. As noted in [11], it is especially effective for tasks that involve imbalanced data sets or small object segmentation, because it focuses on the regions of overlap rather than penalizing all misclassifications equally.

This loss function encourages the model to maximize the overlap between the predicted and true masks for each class, penalizing mismatches proportionally to their size and weight.

#### *4.1.4. Training*

The model was trained in two stages to optimize segmentation performance. Initially, the original UNet architecture with its standard segmentation head was used. Training employed a batch size of 120, a learning rate of $1 \times 10^{-4}$, and the Adam optimizer. The loss function was DiceLoss, incorporating class weights $[0, 1, 1, 1]$ to address class imbalance, and a StepLR scheduler was applied to reduce the learning rate by a factor of 0.1 every 10 epochs. Training continued for 33 epochs until no further improvements were observed, as determined by an early-stopping criterion with a patience of 3 (this step had to be done in two segments since the runtime timed out). Subsequently, the segmentation head was replaced as detailed in the methodology. In this second stage, all layers except the new segmentation head were frozen, and training was conducted with a learning rate of $1 \times 10^{-3}$, maintaining the same batch size, optimizer, loss function, and scheduler settings. After 17 epochs, the model was fine-tuned with a reduced learning rate of $1 \times 10^{-5}$, still freezing all layers except the segmentation head. This final stage yielded a validation loss of 0.0694 and training loss of 0.0684.

### 4.2. Survival Classification Model

#### *4.2.1. Data Pre-Processing*

To prepare the data for the survival classification model, several preprocessing steps were applied to both the survival information and the MRI scan masks. The following outlines the process used for preprocessing:

- Dataset Creation: The BraTS 2020 metadata, which includes MRI scan information, was combined with the survival data based on matching patient IDs. This integration allowed for a comprehensive dataset containing both image and clinical data.
- Processing Survival Data: The 'Survival_days' field, representing the number of days a patient survived post-diagnosis, was processed to ensure all values were floats. Non-numeric entries were removed to maintain data consistency.
- Grouping Survival Data into Bins: To categorize patients based on their survival time, the 'Survival_days' were divided into three bins, ensuring equal distribution across each bin. This was done using quantile-based binning, resulting in the following distribution:
  - Short: 5 to 265 days
  - Medium: 268 to 476 days
  - Long: 486 to 1767 days
- Preprocessing Masks: This process involved loading masks from `.h5` files stored in the data directory and stacking the three mask channels into a format suitable for processing. The mask images were then resized to 224x224 pixels and normalized using PyTorch transforms to align with the input requirements of the pre-trained ResNet50 model. Additionally, patient age was scaled to a range between 0 and 1 to standardize the input feature across the dataset, based on the minimum and maximum age values.
- Data Splitting: The data was split into training, validation, and test sets using an 80-10-10 ratio. A stratified split based on the 'Survival_bins' was performed to maintain equal representation of each survival category in all sets. This resulted in 25,606 training samples, 5,487 validation samples, 5,487 test samples.

The data subsets were then organized into PyTorch dataloaders to facilitate efficient batch processing during model training and evaluation.

#### *4.2.2. Model Architecture*

The custom model for survival classification leverages a modified ResNet50 architecture, which is designed to process and classify MRI brain scan images in combination with patient age data. The architecture consists of three main components:

- Pre-trained ResNet50 Backbone: The backbone of the model is a pre-trained ResNet50 network, initialized with the default weights from ImageNet. The final fully connected (fc) layer of ResNet50 is replaced with a custom sequence, which consists of:
  - A linear layer that reduces the feature dimension from the ResNet output to 128 units.
  - A ReLU activation function.
  - A dropout layer with a dropout rate of 0.3 to help mitigate overfitting.

- Age Processing Layer: This component processes the patient age as an additional input feature. The age input is passed through:
  - A linear layer that projects the age feature into 10 units.
  - A ReLU activation function.
  - Another linear layer that maintains the 10 dimensional feature space.
- Combined Layer: The features extracted from the ResNet50 backbone and the age processing layer are concatenated and fed into the combined layer, which performs further processing to produce the final output:
  - A linear layer that takes the concatenated 138-dimensional input (128 from the image features and 10 from the age features) and projects it to 256 units.
  - A ReLU activation function.
  - A dropout layer with a rate of 0.3.
  - A second linear layer that reduces the feature space to 128 units with ReLU activation.
  - Another dropout layer with a rate of 0.3.
  - A final linear layer that outputs predictions for the specified number of survival bins.

#### *4.2.3. Training*

The custom ResNet50 model was trained using a total of 50 epochs, with an early stopping criterion to prevent overfitting. The training was conducted with a learning rate of 0.0001, utilizing the Adam optimizer, and the learning rate was adjusted periodically using a step scheduler, which reduced the learning rate by a factor of 0.1 every 10 epochs. The loss function used for optimization was CrossEntropyLoss, which is suitable for the multi-class classification task.

During training, the model was evaluated on both the training and validation sets after each epoch. The training accuracy gradually improved from 39.78% in the first epoch to 67.44% by the 11th epoch. Validation accuracy showed an upward trend as well, initially starting at 46.87% and peaking at 65.53% by epoch 15.

Despite the continuous improvement in training accuracy, the validation loss did not significantly decrease after epoch 15, which triggered the early stopping mechanism due to a lack of improvement. The best recorded training loss was 0.5908, and the best validation loss was 0.6687.

### 4.3. Combination of Models

The combined model leverages the outputs of a U-Net model for mask segmentation, which identifies relevant regions within the input images, and a ResNet model for survival bin classification. The process starts by passing the input images through the U-Net, which generates a mask with four channels, each representing different regions (including the background). The output mask is then processed to exclude the background channel, leaving the last three channels representing the segmented regions of interest. These transformed masks are resized and preprocessed to match the input requirements of the ResNet model. The processed masks, along with the age information, are fed into the ResNet model to output a prediction for the survival category.

## 5. RESULTS

### 5.1. Accuracy of Trained UNet

Testing the model with a batch size of 50 yielded a test accuracy of 0.8986, defined as the Dice Coefficient (1 - calculated loss). We can see the break down of test accuracies by mask channels—1 (NEC), 2 (ED), and 3 (ET)—produced values in Table 1. The highest accuracy was observed for channel 3, while channel 2 had the lowest, which aligns with visual inspections showing that the tumor (ET) mask class is the most prevalent.

Fig. 4 presents examples of ground truth and predicted masks, demonstrating a high degree of similarity overall, though with some discrepancies in finer details. Fig. 5 illustrates a confusion matrix of the log-transformed pixel counts for true and predicted labels across all four classes, including the background layer. A log of the counts was used instead to address the overwhelming dominance of background pixels. The confusion matrix shows frequent misclassifications between necrotic tissue (NEC) and edema (ED), likely due to the smaller areas they occupy. Despite these challenges, the results indicate that the model captures the overall structure of the tumor effectively and maintains reasonable performance across all classes.

| Bin | Classification Acc (%) |
|---|---|
| NEC | 0.8728 |
| ED | 0.8275 |
| ET | 0.8942 |

**Table 1**: Classification Accuracy for Different Bins

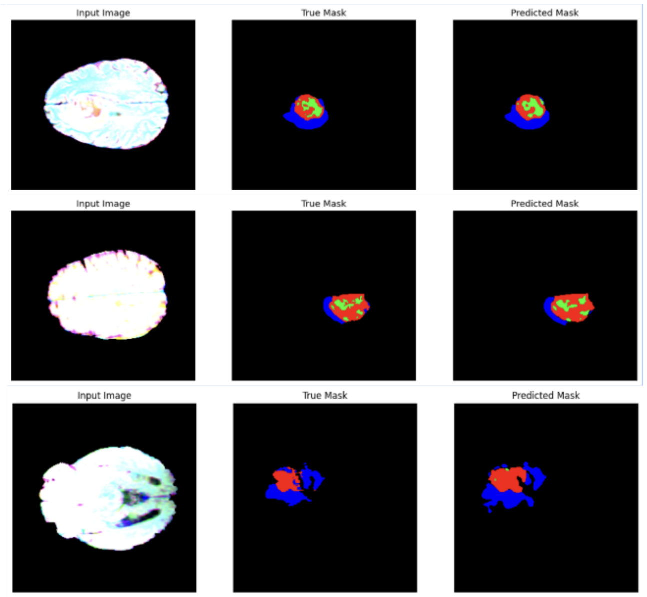

**Fig. 4**: Examples of Ground Truth and Predicted Masks from Trained UNet [1]

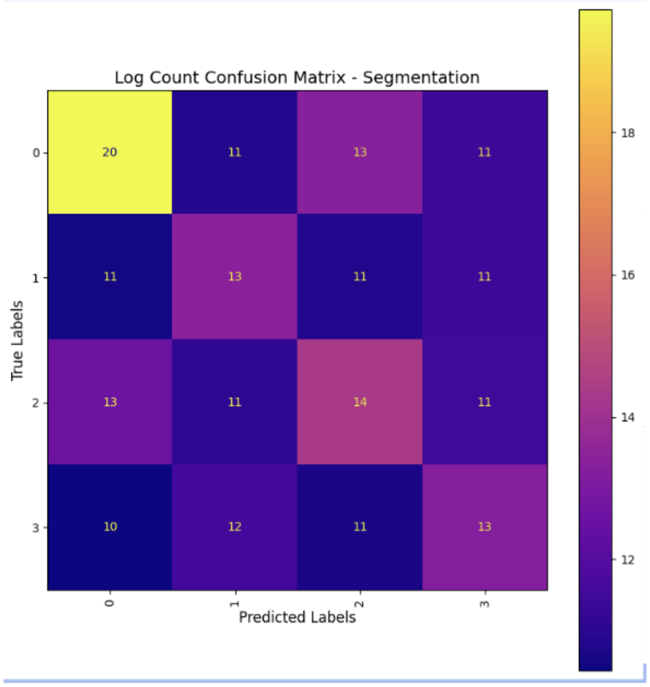

**Fig. 5**: Log Count Confusion Matrix from Trained UNet Model [1]

### 5.2. Accuracy of Trained ResNet50 on Ground Truth Masks

The ResNet50 model was evaluated on the test set to assess its performance in predicting survival categories based on ground truth masks and age data. The overall test accuracy was 65.15%, with a test loss of 0.6613. Accuracy was also analyzed for each survival bin to better understand the model's performance across different survival ranges as shown in Table 2. For bin 0, corresponding to survival times between 5 and 265 days, the accuracy was 70.06%. Bin 1, covering survival times from 268 to 476 days, exhibited the lowest accuracy at 51.58%, indicating potential difficulty in distinguishing this intermediate survival range. In contrast, bin 2, representing the longest survival times (486 to 1767 days), achieved the highest accuracy at 73.93%.

| Bin | Classification Acc (%) |
|---|---|
| 0 | 70.06 |
| 1 | 51.58 |
| 2 | 73.93 |

**Table 2**: Classification Accuracy for Different Bins

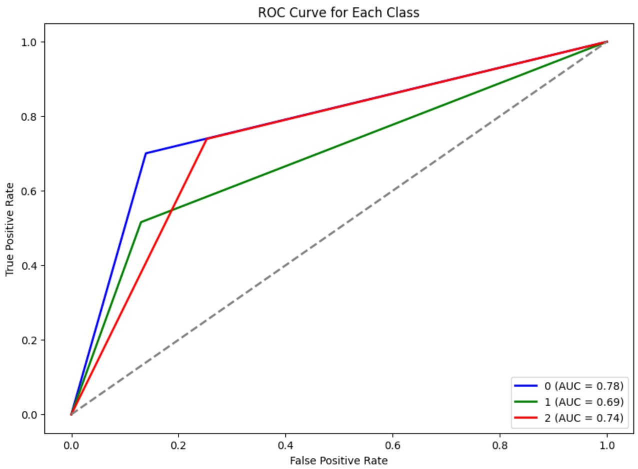

**Fig. 6**: ROC Curve and AUC for Survival Bins

To further understand the model's performance, the ROC (Receiver Operating Characteristic) curve is shown in Fig. 6. The ROC curve evaluates the discriminative ability of the model for each survival bin: short, medium, and long. The axes of the ROC curve represent:

- False Positive Rate (FPR) on the x-axis, which measures the proportion of negative instances misclassified as positive.
- True Positive Rate (TPR) on the y-axis, which mea-

sures the proportion of positive instances correctly classified.

Each curve corresponds to one of the three survival bins, and the area under the curve (AUC) quantifies the model's discriminative performance:

- Bin 0 (Short survival): Achieved the highest AUC of 0.78, indicating strong performance in classifying short survival times.

- Bin 1 (Medium survival): Exhibited the lowest AUC of 0.69, suggesting that the model struggles to distinguish this category.

- Bin 2 (Long survival): Achieved an AUC of 0.74, reflecting reasonably good performance in predicting longer survival times.

The dashed diagonal line represents random classification (AUC = 0.5). All the curves lie above this baseline, demonstrating that the model consistently outperforms random guessing. Notably, the model performs particularly well for the short and long survival bins, with room for improvement in the medium survival bin.

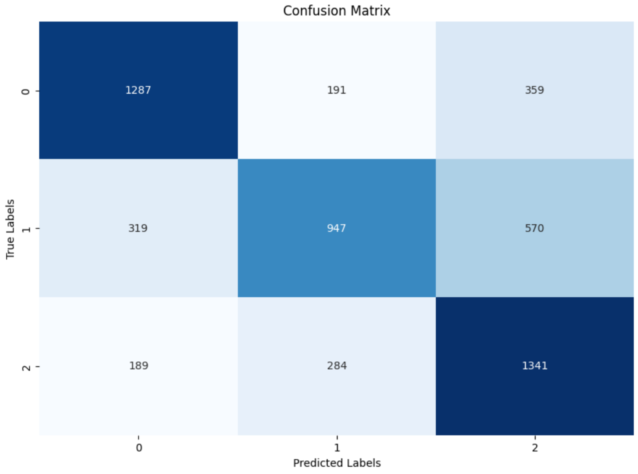

**Fig. 7**: Confusion Matrix for Trained Resnet Model

The confusion matrix, presented in Fig. 7, offers a detailed breakdown of the model's predictions for the survival classification task. The rows indicate the true labels (actual survival bins), while the columns represent the predicted labels. The diagonal elements indicate correctly classified instances, while off-diagonal elements represent misclassifications.

- Bin 0 (Short survival) achieved the highest accuracy, with the most correctly classified instances (1,287). However, a notable number of instances were misclassified into Bin 2 (359), possibly due to overlapping features between short and long survival cases.

- Bin 1 (Medium survival) exhibited the greatest challenge, with a higher number of misclassifications (319 as Bin 0 and 570 as Bin 2) compared to correct classifications. This aligns with the lower AUC observed in the ROC curve analysis, suggesting difficulty in distinguishing medium survival cases.

- Bin 2 (Long survival) performed well, with 1,341 correct classifications. However, some instances were misclassified into Bin 1 (284), indicating potential overlap in features between medium and long survival groups.

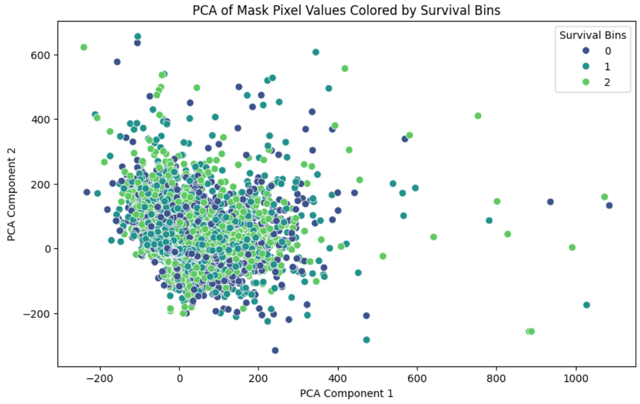

**Fig. 8**: Distribution of Survival Bins in PCA-Reduced Mask Feature Space

The PCA visualization in Fig. 8 provides an additional perspective, showing significant overlap among the survival bins in the reduced 2D feature space. This suggests poor separability among the classes, likely due to insufficient discriminative power of the features extracted from the ground truth masks. The lack of clear separation is most evident in Bin 1, which exhibited the lowest accuracy, suggesting that additional feature refinement may be required to improve the model's performance.

### 5.3. Accuracy of ResNet50 on UNet Generated Masks

When combining the two models, testing involved first passing the 4-channel images through the trained custom UNet and then feeding its output into the trained custom ResNet50 model. This approach yielded a low overall testing accuracy of 0.6014, with a loss of 0.8126. The per-bin accuracy for each survival bin was 0.6473 (Bin 0), 0.4586 (Bin 1), and 0.6996 (Bin 2). However, when images with no mask present (i.e., all-black masks) were excluded, accuracy increased significantly to 86.92%. Furthermore, since originally slices from the same volume, corresponding to a single patient, were treated as separate images, a majority-vote bin was calculated across all slices for each volume. This yielded an accuracy of 98.31%. Table 3 summarizes the per-bin accuracy for filtered masks, both at the image level and the majority-vote volume level.

| True Bin | Image Wise Acc(%) | Maj Vote Acc(%) |
|---|---|---|
| 0.0 | 85.488127 | 97.625330 |
| 1.0 | 87.688098 | 99.316005 |
| 2.0 | 87.755102 | 97.959184 |

**Table 3**: Match percentages for image wise predicted bins and volume wise majority vote bins compared to true bins.

In addition to accuracy per bin, the correlation between the percentage of each mask channel in an image and its image-wise survival bin assignment was calculated using Seaborn, as illustrated in Fig. 9.

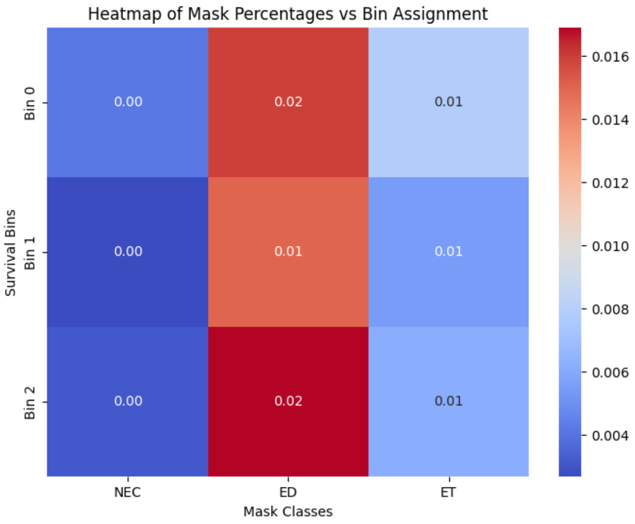

**Fig. 9**: Heatmap Showing Correlation between Mask Channel Class Percentage vs Survival Bin Assignment

## 6. DISCUSSION

### 6.1. Significance

The outcomes of this project highlight the transformative potential of deep learning in addressing critical challenges in brain tumor diagnosis and prognosis. The segmentation model, based on the U-Net architecture, demonstrated its ability to accurately identify and delineate tumor sub-regions from MRI scans. This capability is essential for precise diagnosis and effective treatment planning, as it provides radiologists with a consistent and detailed understanding of tumor characteristics. By incorporating Dice Loss with class weights, the model effectively handled class imbalances, focusing on relevant tumor regions while minimizing the influence of non-tumor background areas.

Building on this, the ResNet50-based survival classification model demonstrated its capacity to predict patient survival categories by combining segmentation outputs with additional features such as age. By linking segmentation results to survival predictions, the model offers valuable insights that could support personalized treatment strategies and inform clinical decision-making. Together, these models underscore the potential of deep learning to enhance the accuracy, consistency, and efficiency of brain tumor analysis, ultimately contributing to improved patient outcomes.

### 6.2. Future Work

To further improve the performance and generalizability of the models, several directions can be explored:

- Input Data Expansion: Incorporating additional MRI imaging perspectives, such as axial, sagittal, and coronal images, as input channels for both the segmentation and classification models could provide a more comprehensive understanding of the brain structure and improve model accuracy.

- Data Augmentation: Implementing data augmentation techniques, including rotation, flipping, and brightness adjustments, can help make the models more resilient to variations in MRI scan quality and improve generalization on unseen data.

- Volume-Based Processing: Modifying the ResNet50 model to process all slices of a given MRI volume as input channels, rather than relying solely on individual masks, could leverage the 3D nature of MRI scans for better contextual understanding and more precise survival classification.

- ResNet Variants: Exploring the use of alternative ResNet architectures, such as ResNet34 and ResNet101, could be beneficial for both the segmentation encoder and the survival classification model.

These steps aim to refine and expand the current model to deliver more precise and scalable solutions for brain tumor diagnosis and prognosis prediction.

## 7. AUTHOR CONTRIBUTION STATEMENT

**S.V.** Developed, trained, and tested the segmentation model. Contributed to writing the Introduction, Related Work, Dataset, Methodology of the Segmentation Model, and Results of the Segmentation Model and Combined Model sections of the report. Additionally, contributed to the Introduction, Dataset, and Segmentation Model and Combined Results portions of the presentation slides.

**D.S.** Developed, trained, and evaluated the classification model. Contributed to drafting the Abstract, Methodology of the Classification Model, Results of the Classification Model, and Discussion sections of the report. Also contributed to the Classification Model, Results, and Future Directions sections of the presentation slides.

**D.S.** & **S.V.** collaboratively worked on sourcing datasets, formulating the project design, and gathering references for the project.

## 8. REFERENCES

[1] Akmalbek Bobomirzaevich Abdusalomov et al., "Brain tumor detection based on deep learning approaches and magnetic resonance imaging," *Cancers*, vol. 15, pp. 4172, August 2023.

[2] Mahmoud Khaled Abd-Ellah et al., "A review on brain tumor diagnosis from mri images: Practical implications, key achievements, and lessons learned," *Magnetic Resonance Imaging*, vol. 61, pp. 300–318, 2019.

[3] Center for Biomedical Image Computing and Analytics, "Brats 2020 dataset," 2020.

[4] Awsaf, "Brats 2020 training data," 2020.

[5] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, and J. Kirby et al., "The multimodal brain tumor image segmentation benchmark (brats)," *IEEE Transactions on Medical Imaging*, vol. 34, no. 10, pp. 1993–2024, 2015.

[6] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, and J. S. Kirby et al., "Advancing the cancer genome atlas glioma mri collections with expert segmentation labels and radiomic features," *Nature Scientific Data*, vol. 4, pp. 170117, 2017.

[7] S. Bakas, M. Reyes, A. Jakab, S. Bauer, M. Rempfler, and A. Crimi et al., "Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the brats challenge," *arXiv preprint*, vol. arXiv:1811.02629, 2018.

[8] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, and J. Kirby et al., "Segmentation labels and radiomic features for the pre-operative scans of the tcga-gbm collection," *The Cancer Imaging Archive*, 2017.

[9] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, and J. Kirby et al., "Segmentation labels and radiomic features for the pre-operative scans of the tcga-lgg collection," *The Cancer Imaging Archive*, 2017.

[10] Olaf Ronneberger, Philipp Fischer, and Thomas Brox, "U-net: Convolutional networks for biomedical image segmentation," *arXiv preprint arXiv:1505.04597*, 2015.

[11] Reza Azad, Moein Heidari, Kadir Yilmaz, Michael Hüttemann, Sanaz Karimijafarbigloo, Yuli Wu, Anke Schmeink, and Dorit Merhof, "Loss functions in the era of semantic segmentation: A survey and outlook," *arXiv preprint arXiv:2312.05391*, 2023.