National College of Ireland

Configuration Manual for Advancing Biomedical Image Segmentation of Lower-Grade Gliomas using Transfer Learning

MSc Research Project
Msc in Data analytics

Yaswanth Vanapalli
Student ID: X23196718

School of Computing
National College of Ireland

Supervisor: Musfira Jilani

National College of Ireland

MSc Project Submission Sheet

School of Computing

Student Name: ……. ……Yaswanth Vanapalli……

Student ID: ……X23196718……

Programme: ……Msc in Data Analytics…… **Year:** …2024-2025….

Module: ……Msc Research Project……

Lecturer: ……Musfira Jilani……

Submission Due Date: ……29 january 2025……

Project Title: ……Advacing Biomedical Image Segmentation Of Lower Grade Gliomas Using Transfer…

Word Count: ……850… **Page Count:** ……8…

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.
ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature: ……Yaswanth vanapalli……

Date: ……29 january2025……

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	□
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	□
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	□

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual for Advancing Biomedical Image Segmentation of Lower-Grade Gliomas using Transfer Learning

Yaswanth Vanapalli
Student ID: X23196718

1. Introduction

This configuration manual provides detailed instructions for setting up and running the model used in the research titled "Advancing Biomedical Image Segmentation of Lower-Grade Gliomas using Transfer Learning" using the Swin Transformer model for tumor segmentation in MRI images of lower-grade gliomas (LGGs).

The manual includes software requirements, hardware configurations, and step-by-step guidance on how to prepare the environment, dataset, and model for training, validation, and testing.

2. Prerequisites

Before you start, make sure you have the following:

2.1. Hardware Requirements

- GPU: A machine with at least one NVIDIA GPU (e.g., GTX 1080 or better) is highly recommended for training deep learning models.
- RAM: Minimum 16 GB RAM.
- Storage: SSD with at least 50 GB of free space for storing datasets and models.

2.2. Software Requirements

- Operating System: Linux (Ubuntu 18.04 or later), Windows (with WSL2), or macOS.
- Python: Version 3.8 or later.
- CUDA and cuDNN: For GPU acceleration (Optional, but recommended for faster training).
- Libraries and Frameworks:
  - TensorFlow or PyTorch (depending on which framework is used for implementation)
  - transformers library (from Hugging Face, for using transformer models)
  - scikit-learn
  - matplotlib, seaborn (for plotting and visualization)
  - opencv-python, Pillow (for image processing)
  - SimpleITK, nibabel (for handling medical imaging formats like MRI scans)

3. Dataset Configuration

3.1. Dataset Overview

The dataset used for training and evaluation in this project is the LGG-MRI dataset, which consists of MRI scans of patients with lower-grade gliomas along with their corresponding tumor masks.

- Format: TIFF images and binary segmentation masks.
- Size: 256x256 pixel images.

3.2. Dataset Setup

Download the Dataset:

- The dataset can be downloaded from Kaggle or other public repositories.

```
# Collects paths of all images in the specified directory
all_image_paths = []

for dirname in os.listdir(dataset_path):
    if os.path.isdir(os.path.join(dataset_path, dirname)):

        for filename in os.listdir(os.path.join(dataset_path, dirname)):
            # Only the files with ',tif' format should be added to the 'paths' list
            if filename.endswith('.tif'):
                all_image_paths.append(dataset_path+'/'+dirname+'/'+filename)

len(all_image_paths), all_image_paths[:25:1]
```

- Make sure the images and masks are organized in directories as follows:

```
(7858,
 ['dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_8_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_9_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_2_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_3_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_20_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_9.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_15_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_14_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_8.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_16.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_18_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_19_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_17.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_15.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_14.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_12_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_13_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_10.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_11.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_13.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_5_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_4_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_12.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_16_mask.tif',
  'dataset/lgg-mri-segmentation/kaggle_3m/TCGA_CS_6667_20011105/TCGA_CS_6667_20011105_17_mask.tif'])
```

Data Preprocessing:

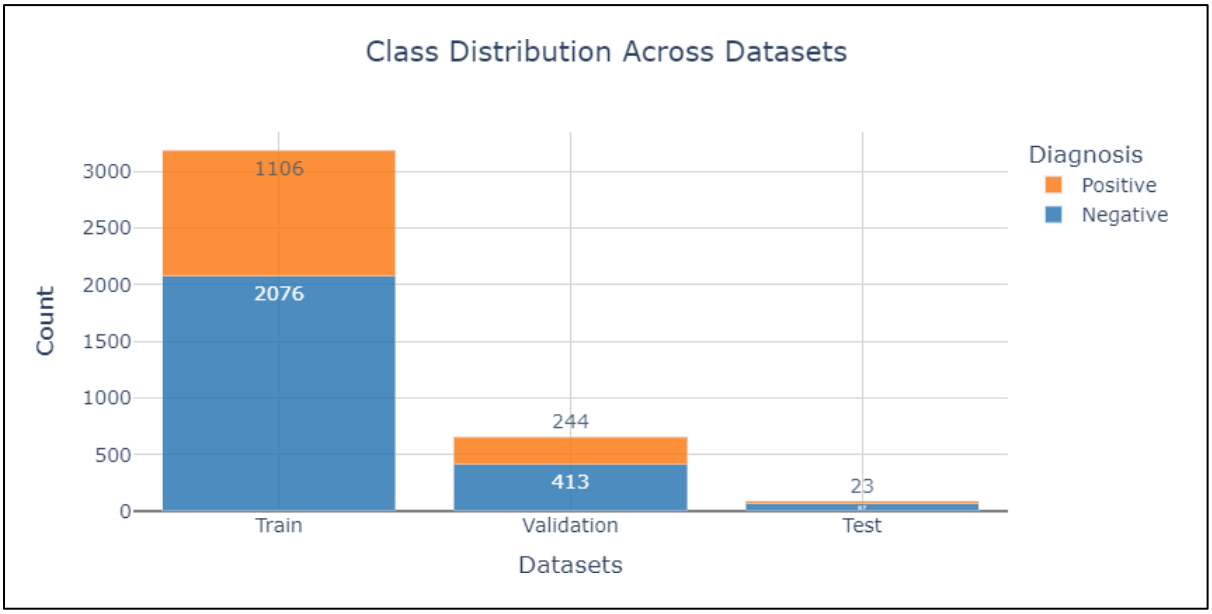
- Resize images to a consistent size (e.g., 256x256 pixels).

```python
# Processes the image
def decode_and_resize_image(img_path):
    # Reading '.tiff' format image
    img = tf.io.read_file(img_path)
    with tf.io.gfile.GFile(img_path, 'rb') as f:
        img = Image.open(f)
        img = np.array(img)
    img = tf.convert_to_tensor(img, dtype=tf.float32)
    img = tf.image.resize(img, IMAGE_SIZE, preserve_aspect_ratio=True)

    # Normalizing the image to the range [0, 1]
    img = img / 255.0

    return img
```

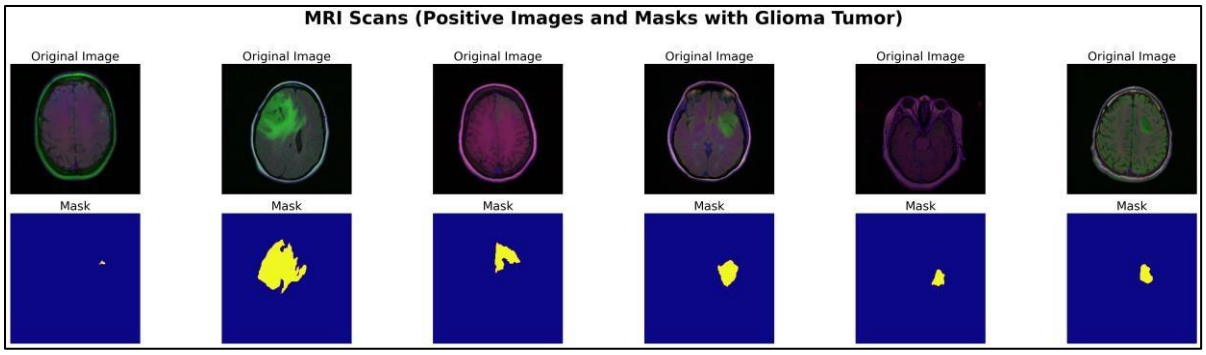
- Normalize pixel values (e.g., scale between 0 and 1).

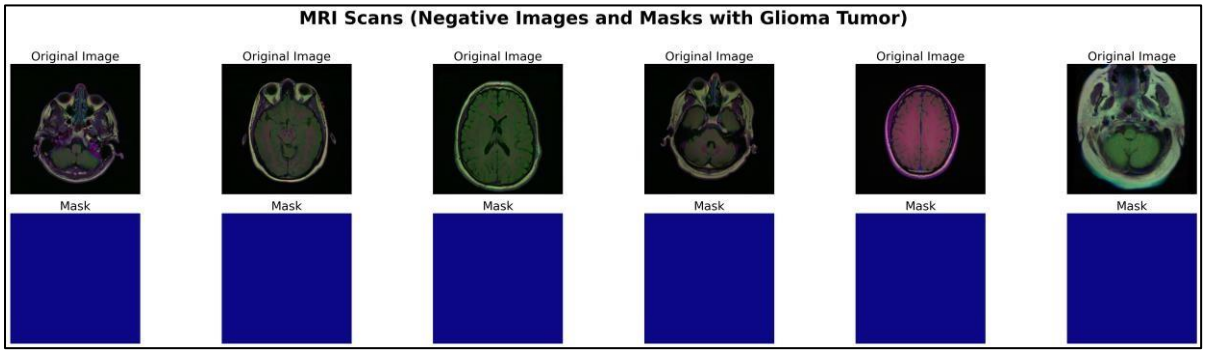
```python
# Processes the mask
def decode_and_resize_mask(mask_path):
    # Reading '.tiff' format masks
    mask = tf.io.read_file(mask_path)
    with tf.io.gfile.GFile(mask_path, 'rb') as f:
        mask = Image.open(f)
        mask = np.array(mask)
    mask = tf.convert_to_tensor(mask, dtype=tf.float32)
    mask = np.expand_dims(mask, axis=-1)
    mask = tf.image.resize(mask, IMAGE_SIZE, method='nearest', preserve_aspect_ratio=True)
    grayscale_mask = tf.reduce_mean(mask, axis=-1, keepdims=True)

    # Normalizing the mask to the range [0, 1]
    grayscale_mask = grayscale_mask / 255.0

    return grayscale_mask
```

- Split the dataset into training (70%), validation (15%), and test (15%) sets.



- Apply data augmentation (e.g., random rotations, zooming, flipping) to increase the diversity of the training set.






```
# Display the some value attributes of training dataset
train_data.head()
```

Python

	ID	Image	Mask	Diagnosis
812	TCGA_DU_5853_19950823_8	dataset/lgg-mri-segmentation/kaggle_3m/TCGA_DU...	dataset/lgg-mri-segmentation/kaggle_3m/TCGA_DU...	0
1550	TCGA_DU_5871_19941206_15	dataset/lgg-mri-segmentation/kaggle_3m/TCGA_DU...	dataset/lgg-mri-segmentation/kaggle_3m/TCGA_DU...	0
2228	TCGA_DU_8163_19961119_21	dataset/lgg-mri-segmentation/kaggle_3m/TCGA_DU...	dataset/lgg-mri-segmentation/kaggle_3m/TCGA_DU...	0
1108	TCGA_DU_7304_19930325_11	dataset/lgg-mri-segmentation/kaggle_3m/TCGA_DU...	dataset/lgg-mri-segmentation/kaggle_3m/TCGA_DU...	0
3728	TCGA_HT_7881_19981015_52	dataset/lgg-mri-segmentation/kaggle_3m/TCGA_HT...	dataset/lgg-mri-segmentation/kaggle_3m/TCGA_HT...	0

4. Defining the Segmentation Model for Lower-Grade Glioma Segmentation through Transfer Learning

4.1. Load Pretrained Swin Transformer

The Swin Transformer model is available in the transformers library by Hugging Face. You can use a pretrained Swin Transformer model fine-tuned for image segmentation tasks.

4.2. Model Customization

To use the pretrained Swin Transformer for segmentation, modify the model architecture to include a final convolutional layer for pixel-wise classification:

4.3. Freezing Layers for Transfer Learning
Fine-tune only the later layers to adapt the model for medical imaging tasks:

```
# Building the Segmentation model
def build_segmentation_model(input_shape=(256, 256, 3)):
    inputs = layers.Input(shape=input_shape)

    # Use pre-trained Swin Transformer model layer (classification task)
    layer = SwinForImageClassification.from_pretrained("microsoft/swin-base-patch4-window7-224")

    # Convolutional layers
    x = layers.Conv2D(32, (3, 3), padding='same', activation='relu')(inputs)
    x = layers.MaxPooling2D(pool_size=(2, 2))(x)

    x = layers.Conv2D(64, (3, 3), padding='same', activation='relu')(x)
    x = layers.MaxPooling2D(pool_size=(2, 2))(x)

    x = layers.Conv2D(128, (3, 3), padding='same', activation='relu')(x)
    x = layers.MaxPooling2D(pool_size=(2, 2))(x)

    # Flatten the output for dense layers
    x = layers.Flatten()(x)
    x = layers.Dense(256, activation='relu')(x)
    x = layers.Dense(128, activation='relu')(x)

    # Output layer with a dense layer
    outputs = layers.Dense(input_shape[0] * input_shape[1], activation='sigmoid')(x)

    # Reshape output to match the input dimensions
    outputs = layers.Reshape((input_shape[0], input_shape[1], 1))(outputs)

    # Creating the model
    model = models.Model(inputs, outputs)

    return model

# Instantiate the model
model = build_segmentation_model()

# Compile the model
model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

# Summary of the model
model.summary()
```

5. Training Configuration for Segmentation Model

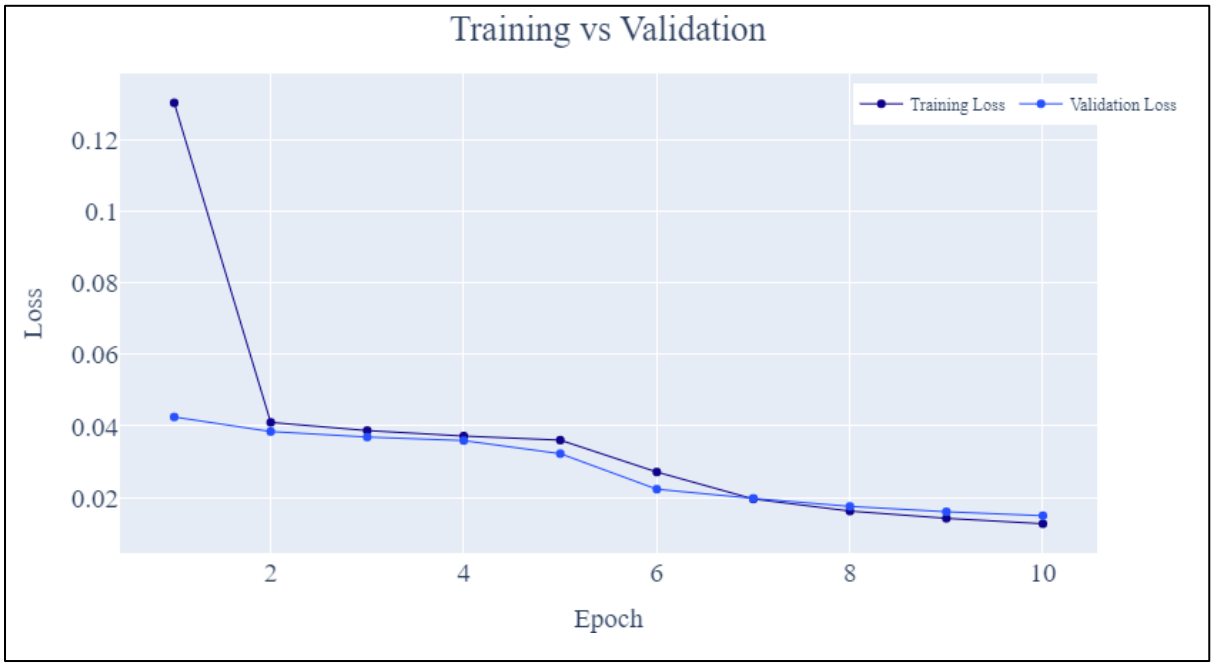
5.1. Hyperparameters
Adjust the following hyperparameters based on your system and dataset:

- Learning Rate: 0.001 (for Adam optimizer)
- Epochs: 10 (adjust based on convergence)
- Optimizer: Adam

- Loss Function: Binary Cross-Entropy for segmentation

```
# Training the model
history = model.fit(train_dataset, validation_data=validation_dataset,
                    epochs=10)
```

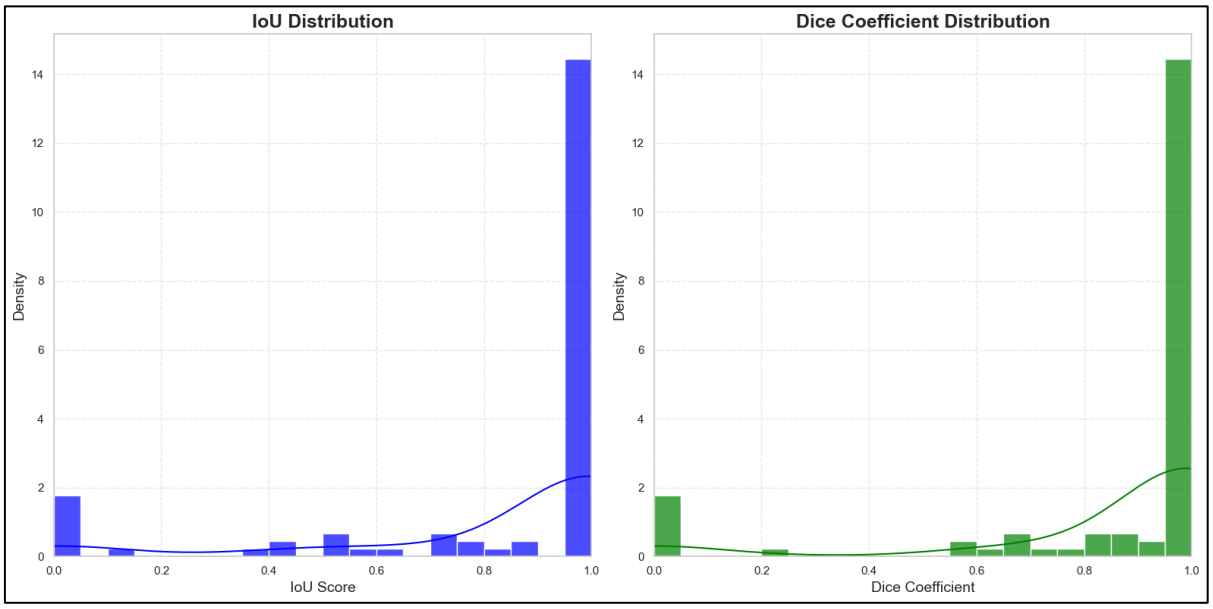
```
Epoch 1/10
71/71 ━━━━━━━━━━━━━━━━━━━━ 81s 1s/step - accuracy: 0.9049 - loss: 0.2625 - val_accuracy: 0.9897 - val_loss: 0.0425
Epoch 2/10
71/71 ━━━━━━━━━━━━━━━━━━━━ 99s 1s/step - accuracy: 0.9898 - loss: 0.0417 - val_accuracy: 0.9897 - val_loss: 0.0384
Epoch 3/10
71/71 ━━━━━━━━━━━━━━━━━━━━ 98s 1s/step - accuracy: 0.9898 - loss: 0.0387 - val_accuracy: 0.9897 - val_loss: 0.0369
Epoch 4/10
71/71 ━━━━━━━━━━━━━━━━━━━━ 114s 2s/step - accuracy: 0.9900 - loss: 0.0366 - val_accuracy: 0.9897 - val_loss: 0.0359
Epoch 5/10
71/71 ━━━━━━━━━━━━━━━━━━━━ 101s 1s/step - accuracy: 0.9897 - loss: 0.0361 - val_accuracy: 0.9900 - val_loss: 0.0322
Epoch 6/10
71/71 ━━━━━━━━━━━━━━━━━━━━ 116s 2s/step - accuracy: 0.9910 - loss: 0.0294 - val_accuracy: 0.9920 - val_loss: 0.0223
Epoch 7/10
71/71 ━━━━━━━━━━━━━━━━━━━━ 108s 2s/step - accuracy: 0.9928 - loss: 0.0204 - val_accuracy: 0.9928 - val_loss: 0.0197
Epoch 8/10
71/71 ━━━━━━━━━━━━━━━━━━━━ 110s 2s/step - accuracy: 0.9939 - loss: 0.0169 - val_accuracy: 0.9935 - val_loss: 0.0175
Epoch 9/10
71/71 ━━━━━━━━━━━━━━━━━━━━ 107s 2s/step - accuracy: 0.9945 - loss: 0.0150 - val_accuracy: 0.9940 - val_loss: 0.0160
Epoch 10/10
71/71 ━━━━━━━━━━━━━━━━━━━━ 121s 2s/step - accuracy: 0.9949 - loss: 0.0133 - val_accuracy: 0.9943 - val_loss: 0.0149
```

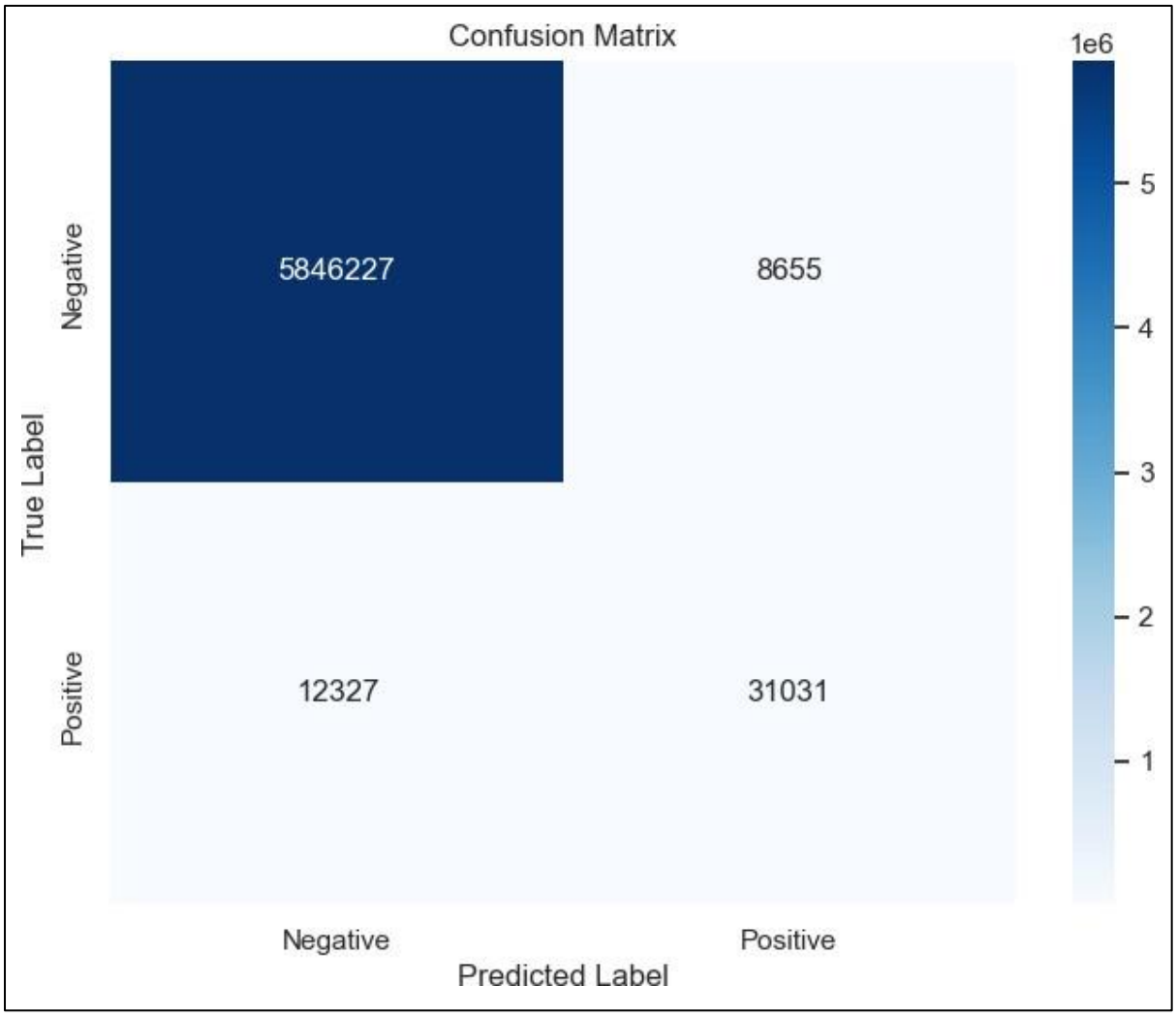


6. Evaluation Configuration

6.1. Segmentation Metrics

Evaluate the model using metrics like IoU, Dice Similarity Coefficient (DSC), and Confusion Matrix:


IoU Distribution
Density
IoU Score
0.0
0.2
0.4
0.6
0.8
1.0
0
2
4
6
8
10
12
14
Dice Coefficient Distribution
Density
Dice Coefficient


Confusion Matrix
1e6
5
4
3
2
1
True Label
Negative
Positive
5846227
8655
12327
31031
Negative
Positive
Predicted Label

```
Classification Report:
              precision    recall  f1-score   support

    Negative       1.00      1.00      1.00   5854882
    Positive       0.78      0.72      0.75     43358

    accuracy                           1.00   5898240
   macro avg       0.89      0.86      0.87   5898240
weighted avg       1.00      1.00      1.00   5898240
```

Conclusion

This manual provides the necessary steps to configure and run the segmentation model for lower-grade gliomas using a Swin Transformer-based architecture with transfer learning. By following this guide, users can replicate the results of this study or adapt the approach for their own medical imaging tasks.

References

Python: https://www.python.org
Dataset Source: https://www.kaggle.com/datasets/mateuszbuda/lgg-mri-segmentation