Major Project - Review 3

# An improved LinkNet based approach for Retinal Image Segmentation

Guide : -
Dr.Velliangiri S
Assistant Professor, Department of Computational Intelligence, SRM Institute of Science and Technology, Kattankulathur.

Team Members : -
Srijarko Roy (RA1911026010087)
Ankit Mathur (RA1911026010088)

# Abstract

The characteristics of Retinal Blood Vessels helps diagnose various eye ailments. The proper localization, extraction and segmentation of the blood vessels is very essential to the cause of treatment of the eye. Manual segmentation of blood vessels may be error prone and inaccurate, leading to difficulty in further treatment. We present a novel approach of semantic segmentation of Retinal Blood Vessels using Linked Networks to account for lost spatial information during feature extraction. The implementation of the segmentation technique involves using Residual Networks as a feature extractor and Transpose Convolution for image to image translation thereby giving a segmentation mask as an output. The main feature of the architecture is the links between the Feature Extractor and the Decoder networks that enhance the performance of the network by helping in the recovery of lost spatial information. Training and Validation using the Pytorch framework has been performed on the Digital Retinal Images for Vessel Extraction (DRIVE) Dataset to establish quality results.

# Introduction

- The retinal vascular system is the only non-invasive imaging technique that can get visible blood vessels from the human body, and it offers abundant information on the health of the eye.
- For the detection of various disorders, retinal vascular segmentation is of utmost importance. Retinal scans are therefore often utilised to identify early indicators of systemic vascular disease.
- Accurate segmentation of the arteries is necessary to make systemic vascular disorders easier to diagnose. As a result, in the realm of medical imaging, the automated segmentation of retinal blood vessels from fundus pictures has grown in popularity.

# Motivation

Manual segmentation of blood arteries may be incorrect and mistake prone, making subsequent treatment challenging. The retinal blood vessels system provides extensive information on the health of the eye and is the only non-invasive imaging tool that can get visible blood vessels from the human body. Retinal vascular segmentation is crucial for the identification of many diseases. Therefore, retinal scans are frequently used to detect early signs of systemic vascular disease. The diagnosis of systemic vascular diseases requires accurate segmentation of the arteries. The automatic segmentation of retinal blood vessels from fundus images has therefore gained favour in the field of medical imaging.

# Plan of Action

| Events | Time Period |
|---|---|
| Ideation | 30th December 2022 |
| Base Paper Review and Literature Survey | 31st December 2022 - 5th January 2023 |
| Data collection and Preprocessing | 7th January 2023 - 16th February 2023 |
| Model Implementation and Validation | 18th February 2023 - 17th March 2023 |
| Model Testing | 19th March 2023 - 20th April 2023 |
| Deployment | 20th April 2023 - 28th April 2023 |

# Requirements

Software Requirements:

- Google Colab for Training
- VS Code
- Linux Terminal for local Testing
- Python
- PyTorch

Hardware Requirements:

- NVIDIA P100
- cuDNN VERSION 8005

# Literature Survey

| Paper | Author(s) | Dataset | Method | Metric | Year |
|---|---|---|---|---|---|
| Retina Blood Vessel Segmentation Using A U-Net Based Convolutional Neural Network | Wang Xiancheng *et. al.* | DRIVE Dataset | U-Net with specific data augmentations | 0.9790 (AUC ROC) | 2018 |

# Literature Survey

| Paper | Author(s) | Dataset | Method | Metric | Year |
|---|---|---|---|---|---|
| Blood Vessel Segmentation in Retinal Images Using Lattice Neural Networks | Gildardo Sanchez-Ante *et. al.* | STARE Database | Lattice Neural Networks with Dendritic Processing | 0.81 (F1 Score) | 2013 |
| Methods for the detection of blood vessels in retinal fundus images and reduction of false-positive pixels around the optic nerve head | Ashish Kumar Dhara *et. al.* | DRIVE Dataset | Multiscale vesselness measures and Gabor filters | 0.9616 (F1 Score) | 2014 |

# Literature Survey

| Paper | Author(s) | Dataset | Method | Metric | Year |
|---|---|---|---|---|---|
| A Fundus Retinal Vessels Segmentation Scheme Based on the Improved Deep Learning U-Net Model | Xiuqin Pan *et. al.* | DRIVE Dataset | Improved U-Net | 96.50% (Segmentation Accuracy) | 2019 |
| DeepVessel: Retinal Vessel Segmentation via Deep Learning and Conditional Random Field | Huazhu Fu *et. al.* | DRIVE, STARE and CHASE_DB1 | Multi-level CNN with Conditional Random Fields | 0.9523 (Accuracy) | 2016 |

# Literature Survey

| Paper | Author(s) | Dataset | Method | Metric | Year |
|---|---|---|---|---|---|
| Segmentation of retinal blood vessels by combining the detection of centerlines and morphological reconstruction | Mendonca *et. al.* | DRIVE Dataset | Morphological Processing | 0.9452 (Accuracy) | 2006 |
| An effective retinal blood vessel segmentation method using multi-scale line detection | Nguyen *et. al.* | STARE Dataset | Multi-Scale Line Detection | 0.9326 (Accuracy) | 2013 |

# Literature Survey

| Paper | Author(s) | Dataset | Method | Metric | Year |
|---|---|---|---|---|---|
| Retinal blood vessel segmentation based on heuristic image analysis | Maja Braovic *et. al.* | DRIVE and STARE Dataset | Modified SUSAN edge detector and shape analysis | 96.33% and 96.10% (Accuracy) respectively | 2019 |
| Retinal vessels segmentation based on level set and region growing | Yu Qian Zhao *et. al.* | DRIVE and STARE databases | Contrast-limited adaptive histogram equalization (CLAHE), followed by a 2D Gabor wavelet to enhance the contrast of the retinal image. | 94.77% on the DRIVE database and 95.09% on the STARE database (Accuracy) | 2014 |

# Literature Survey

| Paper | Author | Dataset | Method | Metric | Year |
| --- | --- | --- | --- | --- | --- |
| An Ensemble Classification-Based Approach Applied to Retinal Blood Vessel Segmentation | Fraz *et. al.* | DRIVE, STARE, CHASE | Ensemble system of bagged and boosted decision trees and utilizes a feature vector based on the orientation analysis of gradient vector field, morphological transformation, line strength measures, and Gabor filter | 0.9480, 0.9534, 0.9469 respectively (Accuracy) | 2012 |
| A New Supervised Method for Blood Vessel Segmentation in Retinal Images by Using Gray-Level and Moment Invariants-Based Features | Diego Marin *et. al.* | DRIVE | neural network (NN) scheme for pixel classification and moment invariants-based features for pixel representation | 0.9452 (Accuracy) | 2010 |

# Literature Survey

| Paper | Author(s) | Dataset | Method | Metrics | Year |
|---|---|---|---|---|---|
| Trainable COSFIRE filters for vessel delineation with application to retinal images | G Azzopardi *et. al.* | DRIVE | B-COSFIRE filter | 0.9442 (Accuracy) | 2015 |
| Iterative vessel segmentation of fundus images | Roychowdhury *et. al.* | DRIVE | Unsupervised Iterative Segmentation with tophat reconstruction and global thresholding | 0.9494 (Accuracy) | 2015 |

# Literature Survey

| Paper | Author(s) | Dataset | Method | Metrics | Year |
|---|---|---|---|---|---|
| Ridge-Based Vessel Segmentation in Color Images of the Retina | Joes Staal *et. al.* | DRIVE | Ridge-Based | 0.9441 (Accuracy) | 2004 |
| Holistically-Nested Edge Detection | Saining Xie, Zhuowen Tu | DRIVE | HED | 0.9435 (Accuracy) | 2016 |

# Literature Survey

| Paper | Author(s) | Dataset | Method | Metrics | Year |
|---|---|---|---|---|---|
| Comparative Analysis of Vessel Segmentation Techniques in Retinal Images | AZHAR IMRAN *et. al.* | DRIVE, STARE, CHASE_DB1, HRF, MESSIDOR, REVIEW | Vector machine-based methods, neural network-based methods, miscellaneous methods, matched filter methods. | 67.6% On CHASE_DB1 (Accuracy) | 2019 |
| A retinal vessel boundary tracking method based on Bayesian theory and multi-scale line detection | JiaZhang *et. al.* | REVIEW (KPIS, CLRIS, HRIS, VDIS) | A Bayesian method with the Maximum a posterior (MAP) | 100%, 98.23%, 94.2%, 100% (Width Error) | 2014 |

# Literature Survey

| Paper | Author(s) | Dataset | Method | Metrics | Year |
|---|---|---|---|---|---|
| Accurate Retinal Vessel Segmentation via Octave Convolution Neural Network | Zhun Fan *et. al.* | DRIVE, STARE, CHASE DB1, and HRF | Octave UNet | 0.9663 (Accuracy), on DRIVE Dataset | 2019 |
| Retinal Blood Vessel Segmentation by Support Vector Machine Classification | Eva Tuba *et. al.* | DRIVE | Overlapping-block-based supervised machine learning approach for blood vessel segmentation | 0.9538 (Accuracy), 0.9773 (Specificity) | 2017 |

# Literature Survey

| Paper | Author(s) | Dataset | Method | Metrics | Year |
|---|---|---|---|---|---|
| AUTOMATIC BLOOD VESSEL SEGMENTATION IN COLOR IMAGES OF RETINA | A. OSAREH and B. SHADGAR | Custom Dataset | Octave UNet | 95.24% (Accuracy) | 2009 |
| Segmentation of retinal blood vessels using the radial projection and semi-supervised approach | Xinge You *et. al.* | DRIVE and STARE | Vessel segmentation | 0.9434 on DRIVE And 0.9497 on STARE | 2011 |

# Literature Survey

| Paper | Author(s) | Dataset | Method | Metrics | Year |
|---|---|---|---|---|---|
| Morphology Approach for Features Extraction in Retinal Images for Diabetic Retionopathy Diagnosis | Ibrahim Abdurrazaq *et. al.* | DRIVE | Mathematical morphology theory and intensity transformation algorithm | TPF value of 0.8214 (True Positive Fraction) | 2008 |
| Retinal Vessel Segmentation Using Deep Neural Networks | Martina Melinscak, Pavle Prentasic and Sven Loncaric | DRIVE | DNNs or CNNs which instead of subsampling or down-sampling layers have a maxpooling layer (MPCNNs). | 0.9466 with 0.7276 and 0.0215 TPR and FPR | 2015 |

# Literature Survey

| Paper | Author(s) | Dataset | Method | Metrics | Year |
|---|---|---|---|---|---|
| Construction of Retinal Vessel Segmentation Models Based on Convolutional Neural Network | Qiangguo Jin *et. al.* | DRIVE and STARE | Deformable-Conv Net | Accuracy of 0.9628/0.9690 | 2020 |
| Retinal Vessel Segmentation Using the 2-D Morlet Wavelet and Neural Network | R.Ghaderi *et. al.* | DRIVE | Morlet transform | 0.90% to 0.9668% of the area under the ROC curve | 2008 |

# Challenges and limitations in existing system

- Existing systems for Retinal Blood Vessel Segmentation such as UNETs, CNNs, SVMs have a common problem of the loss of important spatial information during the feature extraction stage as a result of multiple downsampling. Hence, the existing models are not able to achieve the accuracy that it could have if the spatial information has been retained.

- Certain networks such as UNETs use Transpose Convolutions for the Upsampling task. The usage of Transpose Convolution often leads to noisy outputs.

# Problem Statement: -

Manual segmentation of Retinal Blood Vessels is often error prone and exhausting for skilled ophthalmologists, leading to improper diagnosis of the eye ailments.

# Objective: -

Implementations of a Semantic Segmentation Network to accurately segment the Blood Vessels present in the retina for proper treatment and reduction of operator fatigue.

# Innovation idea of the project

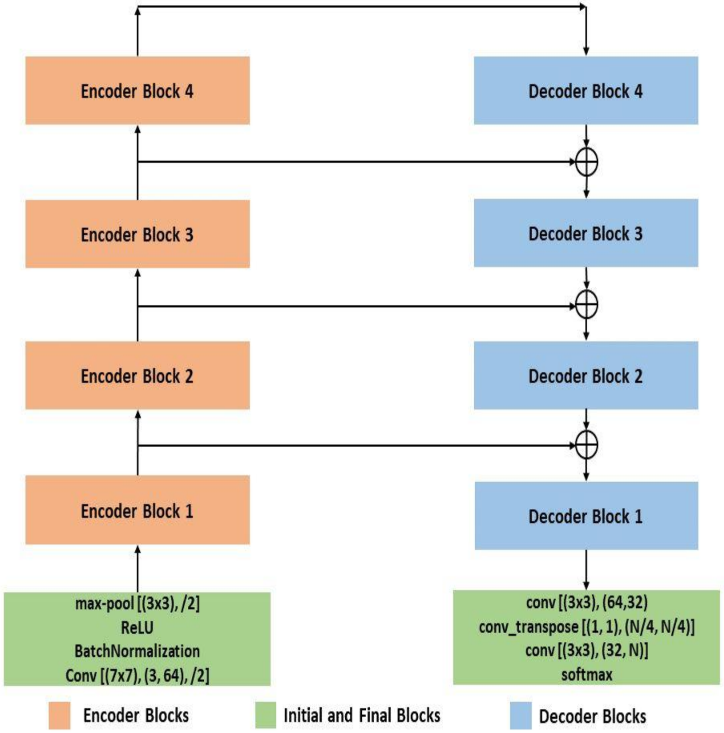
In order to counter the problem of lost spatial information, we propose using an improved LinkNet based architecture for semantic segmentation of Retinal Blood Vessels.

- The network makes use of skip connections to link each Encoder Block to its corresponding Decoder Block, passing on the lost spatial information to be concatenated during Upsampling.
- Using Upsample instead of Transpose Conv counters the problem of noisy output.

# Scope and application of the project

- Medical Science
- Ophthalmology
- Retinal Disease Diagnosis
- Disease Monitoring in Diagnosis Centers
- Ophthalmic Research Institutes
- Educational Institutions offering courses in Ophthalmology

# Architecture




Encoder: ResNet18

Decoder: ConvTranspose Blocks

Linking between each Encoder Block with corresponding Decoder Block

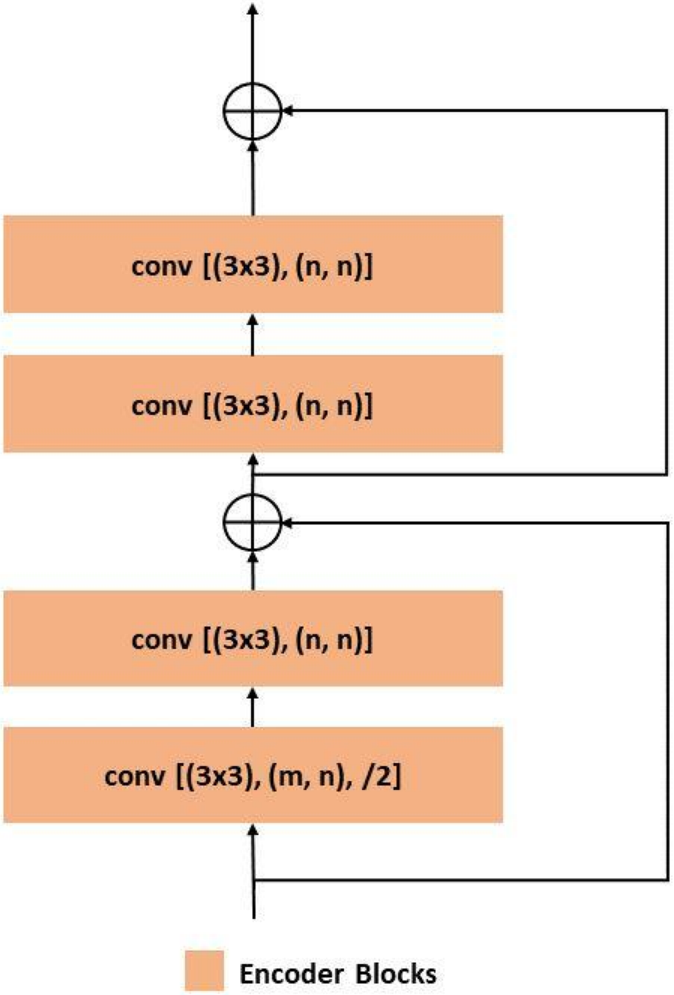
# Modules

- Albumentations - It is used for augmenting the image during preprocessing.
- Imageio - This module is responsible for performing image processing.
- Pytorch - This is acts as the framework for our model's architecture.
- Urllib.request - This module is used for getting an input image into the ui via image address.
- Streamlit: This the platform on which our web application has been developed.

# Modules (Architecture)

Encoder: The encoder network comprises of an initial block and 4 residual blocks for feature extraction purposes.

- Initial Block: The initial block of the encoder performs a convolution operation on the input image, with a (7x7) kernel and a stride of 2. This is followed by Batch Normalization and ReLU activation before a spatial max-pooling operation with a kernel size of (3x3) and a stride of 2.
- Residual Blocks: Following the initial block of the encoder are the residual blocks, used for feature extraction. Each residual block has a strided convolution operation followed by 3 convolutional operations with a (3x3) kernel accompanied by skip connections.


conv [(3x3), (n, n)]
conv [(3x3), (n, n)]
conv [(3x3), (n, n)]
conv [(3x3), (m, n), /2]
Encoder Blocks

# Modules

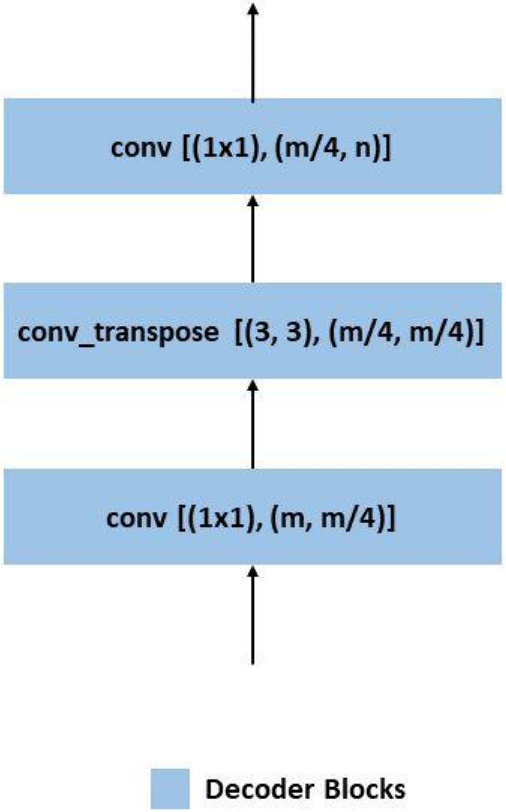
Linked Architecture:

The linking of each encoder to each decoder has been performed exactly as mentioned in, in order to recover the lost spatial information as a result of multiple downsampling operations in its encoder. To enable the linking operation, strided convolutions have been used in the encoder.

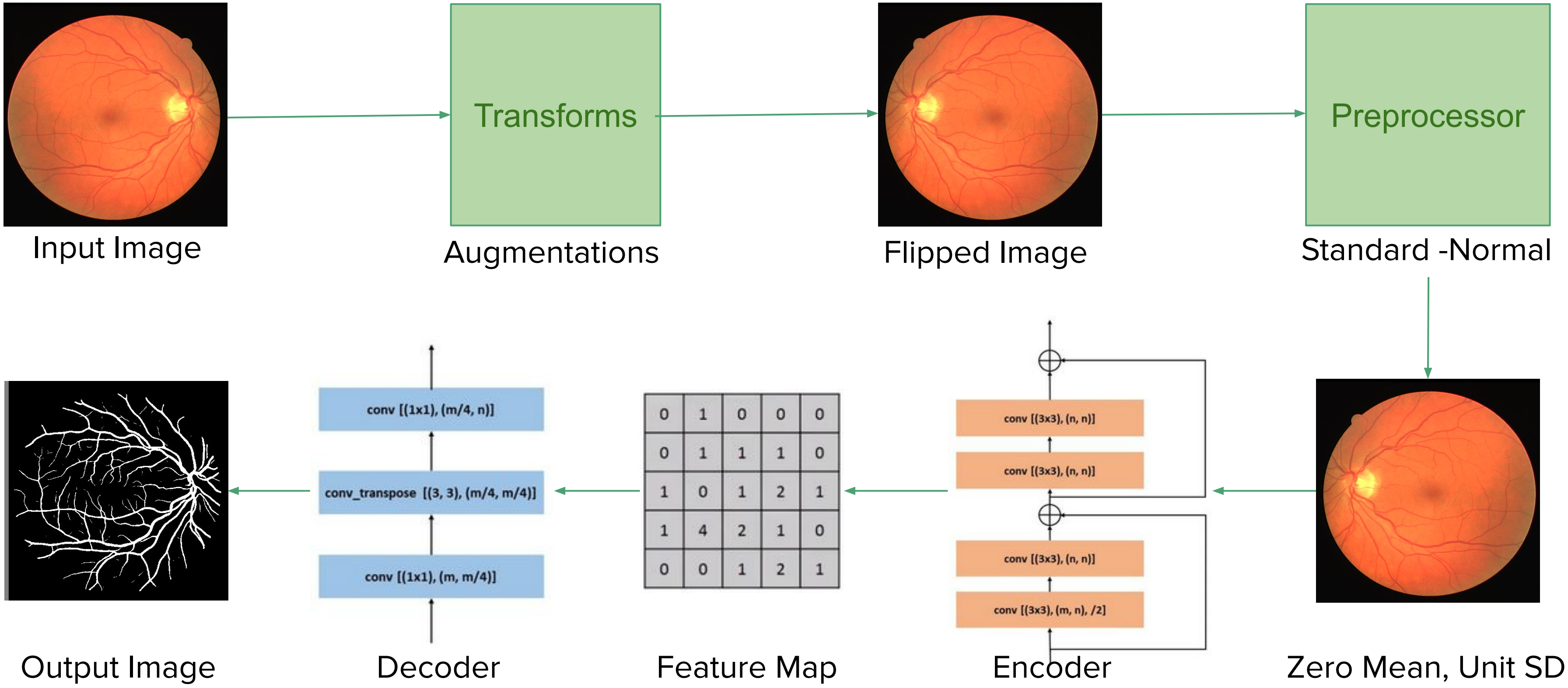
# Modules

Decoder: The decoder network takes the output from the encoder as its input. It comprises 4 decoder blocks and a final segmentation block to perform the main segmentation task.

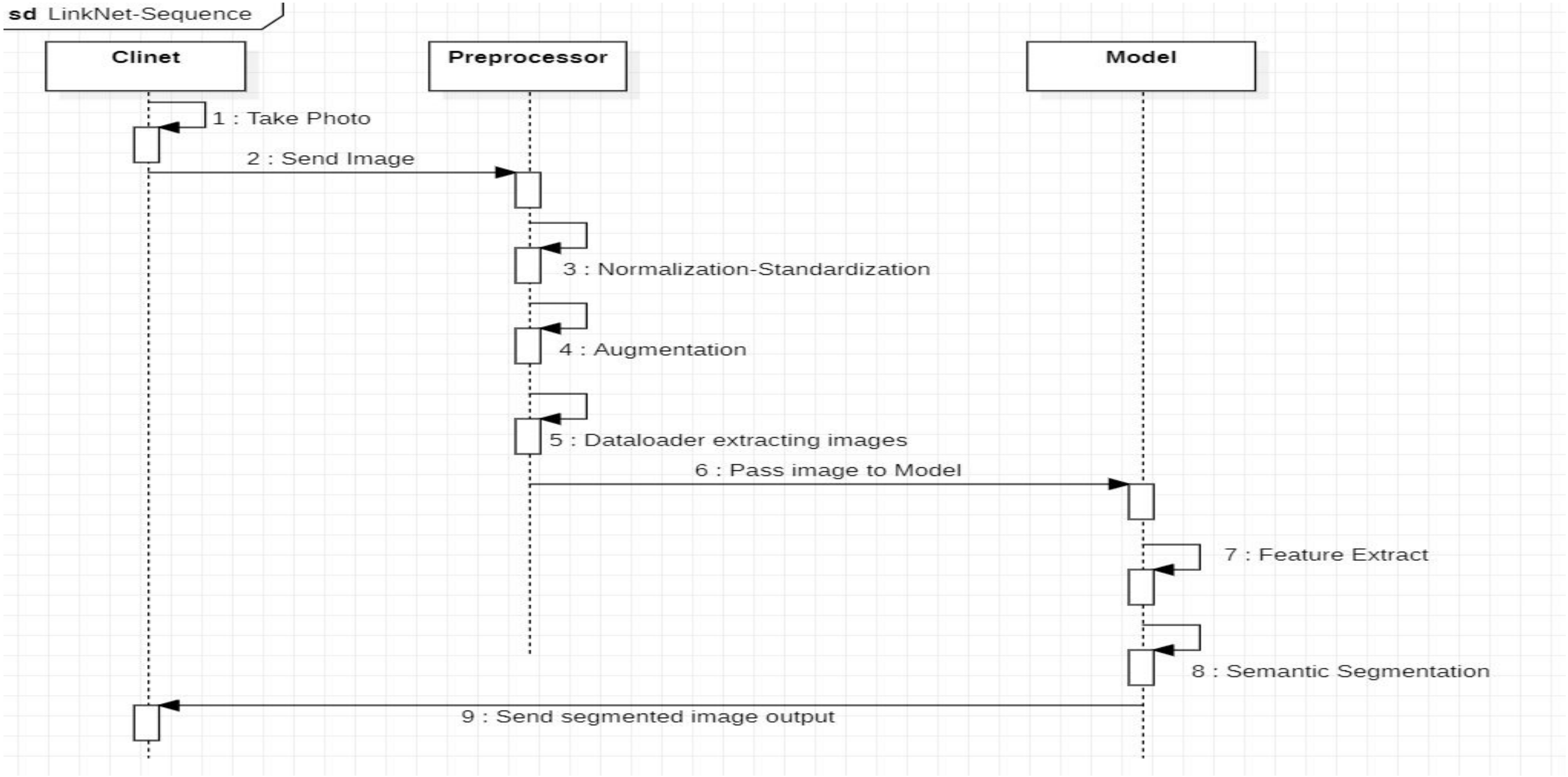
- Decoder Blocks: Each decoder block has 2 convolutional operations with a (1x1) kernel and an Upsample operation in trilinear mode with scale factor 2, between them. The Upsample operation counters the noisy output produced by transpose convolution. The decoder blocks are followed by the final segmentation block.
- Final Segmentation Block: The final segmentation block of the decoder performs the main segmentation task on the output received from Decoder Block 1. A convolution operation with kernel size (3x3) is performed on the decoder output, followed by an Upsample operation with scale factor 2 in the trilinear mode for a noise-reduced output. Finally, another convolution operation with kernel size (3x3) is performed before passing it through a softmax layer to get the final segmentation mask with pixel probabilities.


conv [(1x1), (m/4, n)]
conv_transpose [(3, 3), (m/4, m/4)]
conv [(1x1), (m, m/4)]
Decoder Blocks

# Algorithmic description using Workflow Diagram



# UML Diagrams (Sequence Diagram)



# Experiments and Result Analysis

## A. Dataset

We have used the Digital Retinal Images for Vessel Extraction (DRIVE) dataset for retinal vessel segmentation. It consists of a total of JPEG 40 color fundus images; including 7 abnormal pathology cases. Each image resolution is 584x565 pixels with eight bits per color channel (3 channels), resized to 512x512 for our model. The data has been augmented using the albumentations module. Data Augmentation has been accomplished by doing HorizontalFlip, VerticalFlip and Rotate, thereby generating 160 images for training.

# Experiments and Result Analysis

**Figure 4.2: Original, Horizontal Flip, and Vertical Flip Images**

Pre-processing techniques like normalisation and standardisation have been applied along with augmentations. All images were processed to have zero mean and unit standard deviation as a part of the pre-processing. The standard normal images were passed to the model for training.

# Experiments and Result Analysis

B. Metrics

1. **Dice Loss:** We have adopted Dice Loss as a metric for evaluating the performance of our model. The overlap or resemblance between two sets is calculated using the dice coefficient. This metric is particularly popular with class-imbalance problems. The predicted label and the ground truth can be considered as two sets for the purposes of semantic segmentation. The Dice coefficient ranges in value from 0 to 1, where 0 indicates no overlap and 1 indicates total overlap. The dice coefficient is calculated using the following formula.

   Dice Coefficient = $2 \mid T \cap P \mid / \mid T \mid + \mid P \mid$

   The generalised loss function is calculated from the dice coefficient by subtracting it from 1. This dice loss is minimised which maximises the dice coefficient as higher value implies a better overlap.

# Experiments and Result Analysis

**2. IoU Loss:** We have used Intersection over Union as another metric for evaluating the performance of our model. IoU, as the name suggests, is calculated as a ratio of the overlap of the predicted label with the ground truth to their union, i.e. the total area they cover. Similar to the dice loss, it ranges from 0 to 1 , with 0 indicating no overlap and 1 indicating total overlap. The Intersection over Union value is calculated as:

Intersection over Union = $| T \cap P| / |T \cup P|$

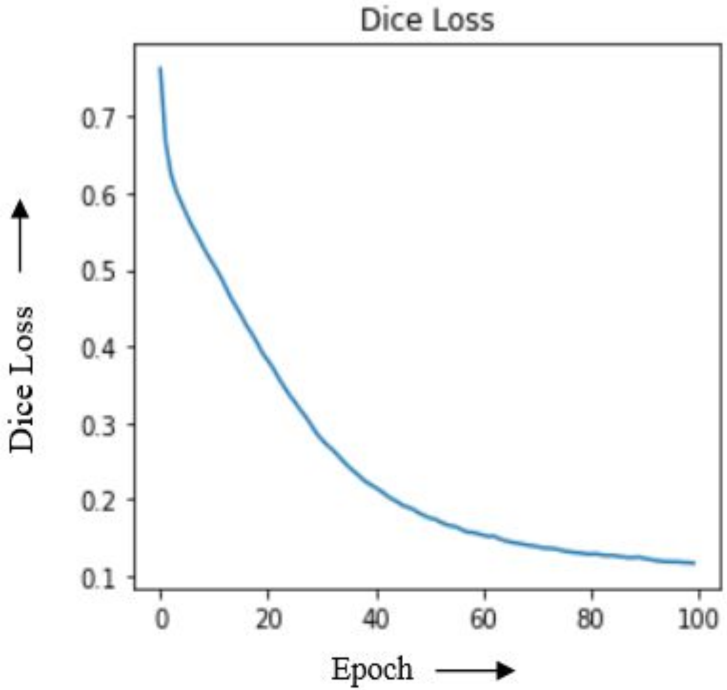
The IoU loss function is calculated by subtracting the IoU Score from 1, and similar to the Dice Loss, a lower IoU loss gives a better overlap, hence is minimised.

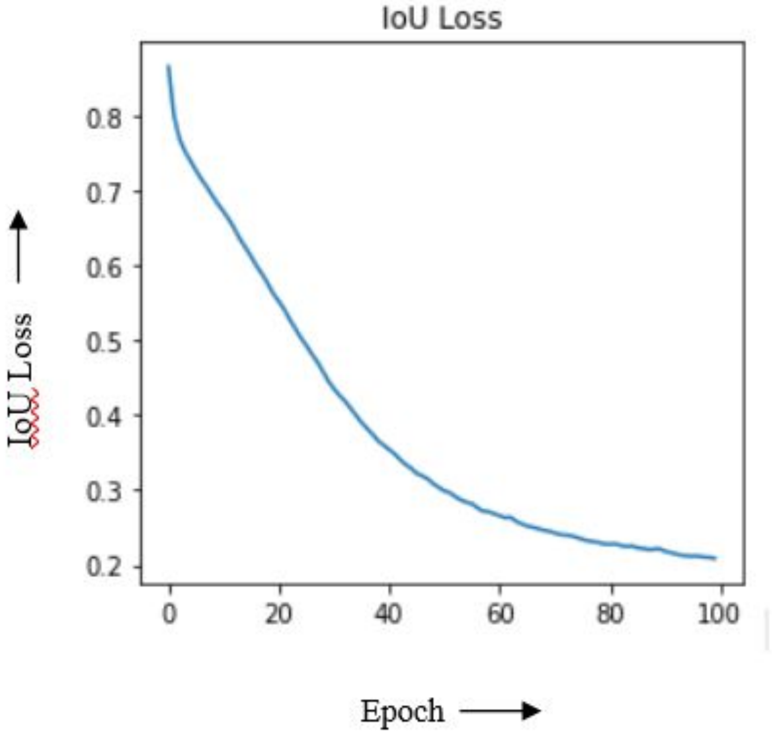
# Experiments and Result Analysis

C. Training Details

The PyTorch framework was used to code the architecture, and the NVIDIA Tesla T4 GPU and CUDA integration were used to train it on 80 images with a batch size of 4. Before being loaded into a dataloader object for training, each picture underwent pre-processing, was enhanced, normalised, and standardised. To reduce the dice and IoU loss, the model was trained using the Adam optimizer for 100 iterations at a learning rate 0.0001.

# Experiments and Result Analysis



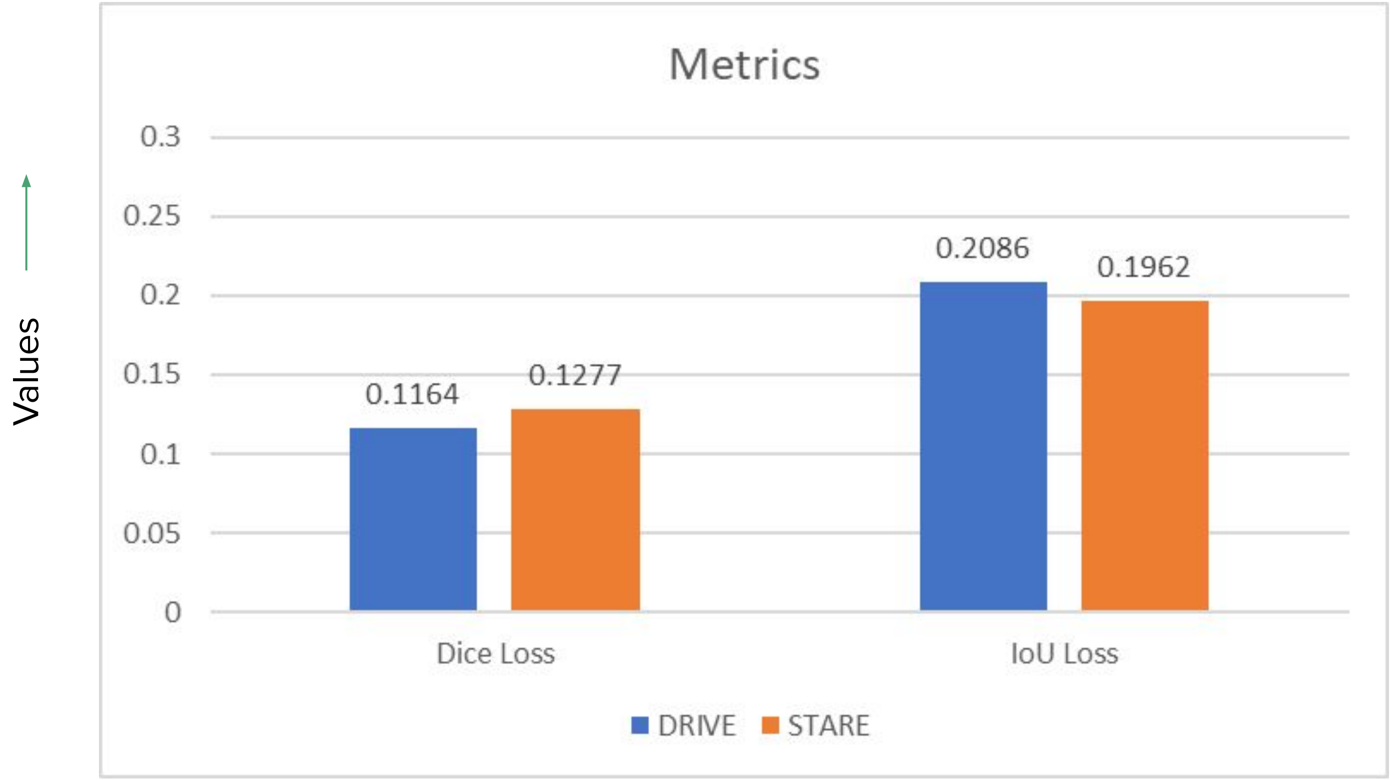
# Experiments and Result Analysis

## D. Experimental Results

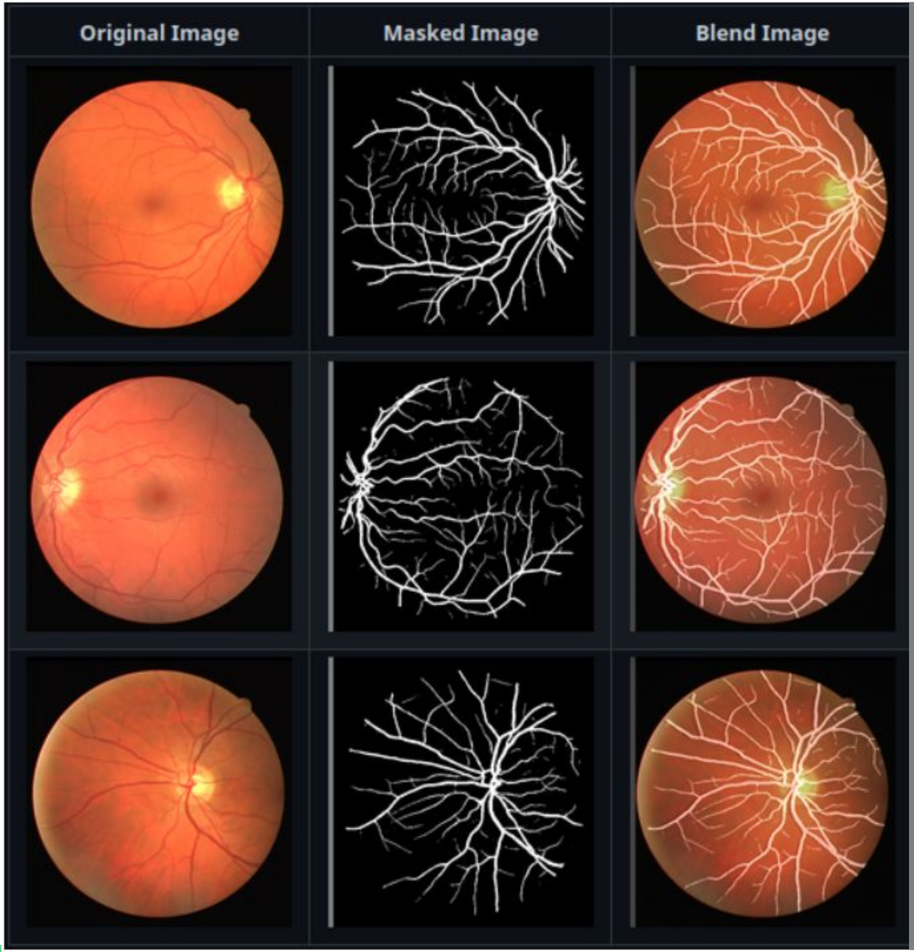
Validation was performed on the DRIVE Dataset and the STARE Dataset using NVIDIA Tesla T4 GPU. The weights were hosted and downloaded via gdown for further tests. The Figure on the next slide shows the test results of the input retinal images from DRIVE Dataset and validation results are mentioned in the below Table.

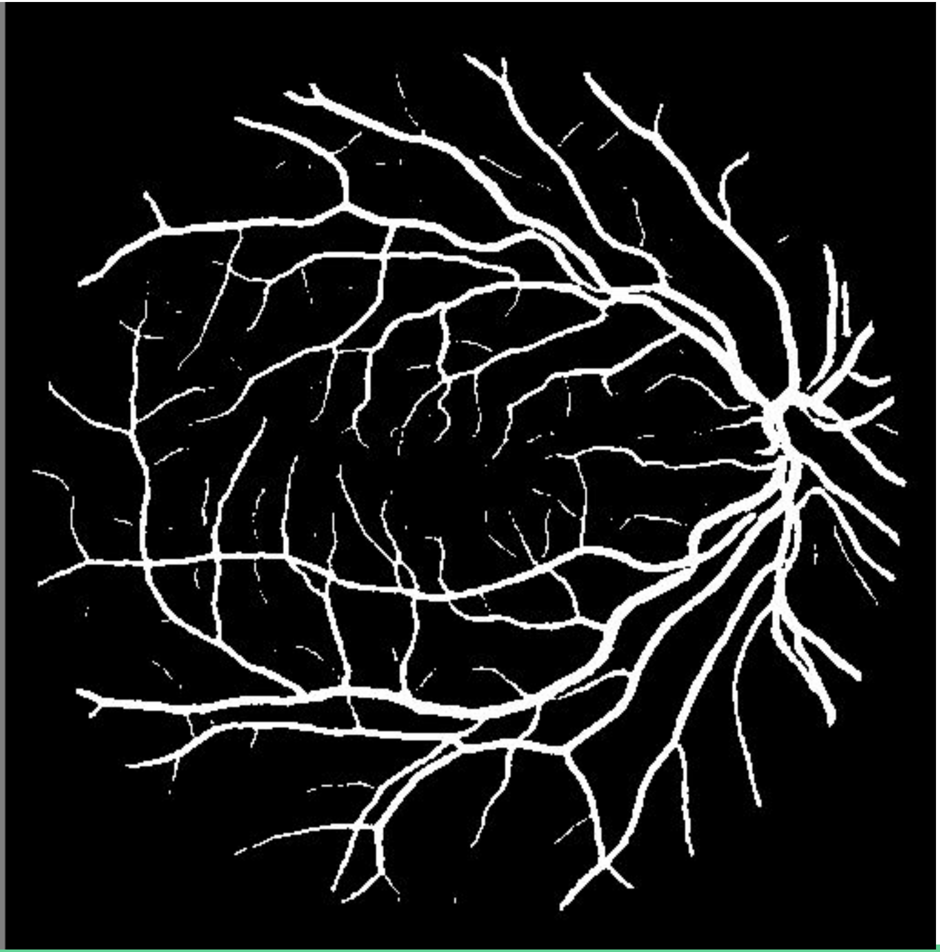
| **Metrics** | **DRIVE** | **STARE** |
|---|---|---|
| **Dice Loss** | 0.1164 | 0.1277 |
| **IoU Loss** | 0.2086 | 0.1962 |

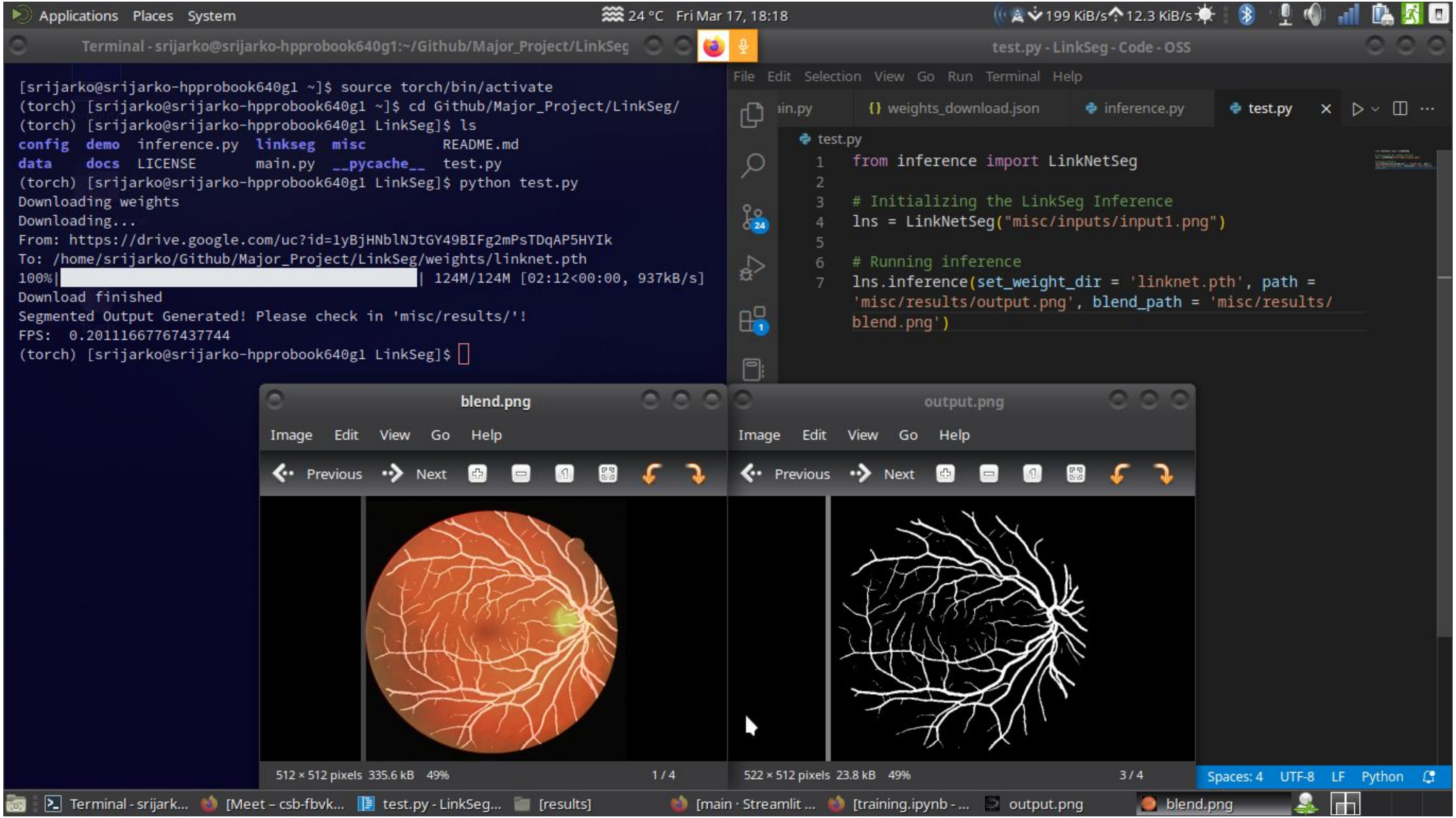
# Experiments and Result Analysis

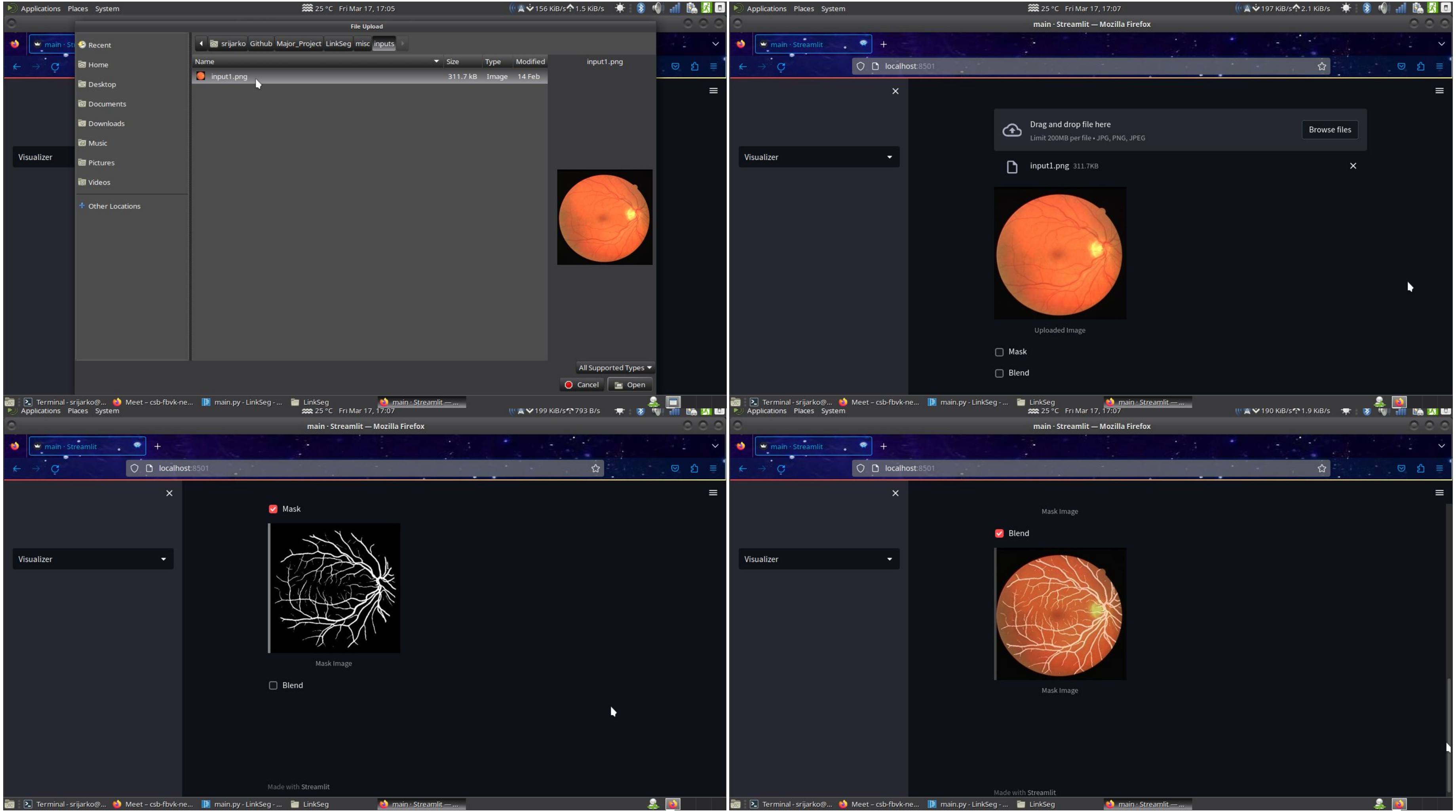


# Experiments and Result Analysis




Applications Places System
24 °C Fri Mar 17, 18:18
199 KiB/s 12.3 KiB/s
Terminal - srijarko@srijarko-hpprobook640g1:~/Github/Major_Project/LinkSeg
[srijarko@srijarko-hpprobook640g1 ~]$ source torch/bin/activate
(torch) [srijarko@srijarko-hpprobook640g1 ~]$ cd Github/Major_Project/LinkSeg/
(torch) [srijarko@srijarko-hpprobook640g1 LinkSeg]$ ls
config demo inference.py linkseg misc README.md
data docs LICENSE main.py __pycache__ test.py
(torch) [srijarko@srijarko-hpprobook640g1 LinkSeg]$ python test.py
Downloading weights
Downloading...
From: https://drive.google.com/uc?id=1yBjHNblNJtGY49BIFg2mPsTDqAP5HYIk
To: /home/srijarko/Github/Major_Project/LinkSeg/weights/linknet.pth
100%| | 124M/124M [02:12<00:00, 937kB/s]
Download finished
Segmented Output Generated! Please check in 'misc/results/'!
FPS: 0.20111667767437744
(torch) [srijarko@srijarko-hpprobook640g1 LinkSeg]$
test.py - LinkSeg - Code - OSS
File Edit Selection View Go Run Terminal Help
ain.py weights_download.json inference.py test.py
test.py
from inference import LinkNetSeg
# Initializing the LinkSeg Inference
lns = LinkNetSeg("misc/inputs/input1.png")
# Running inference
lns.inference(set_weight_dir = 'linknet.pth', path = 'misc/results/output.png', blend_path = 'misc/results/blend.png')
blend.png
Image Edit View Go Help
Previous Next
512 × 512 pixels 335.6 kB 49% 1 / 4
output.png
Image Edit View Go Help
Previous Next
522 × 512 pixels 23.8 kB 49% 3 / 4
Spaces: 4 UTF-8 LF Python
Terminal - srijark... [Meet – csb-fbvk... test.py - LinkSeg... [results] [main · Streamlit ... [training.ipynb - ... output.png blend.png


File Upload
Recent
Home
Desktop
Documents
Downloads
Music
Pictures
Videos
Other Locations
srijarko Github Major_Project LinkSeg misc inputs
Name Size Type Modified
input1.png 311.7 kB Image 14 Feb
input1.png
All Supported Types
Cancel
Open
main · Streamlit — Mozilla Firefox
localhost:8501
Drag and drop file here
Limit 200MB per file • JPG, PNG, JPEG
Browse files
Visualizer
input1.png 311.7KB
Uploaded Image
Mask
Blend
Mask Image
Made with Streamlit

# Comparison with State-of-the-Art

| Architecture | Metric |
| --- | --- |
| UNETs | 0.9790 (AUC ROC) |
| Lattice NN with Dendrite Processing | 0.81 (F1 Score) |
| Multi-level CNN with Conditional Random Fields | 0.9523 (Accuracy) |
| Multi-scale Line Detection | 0.9326 (Accuracy) |
| CLAHE | 0.9477 (Accuracy) |
| Modified SUSAN edge detector | 0.9633 (Accuracy) |
| LinkNet | 0.8836 (Dice Score) |