Report

> Student Name: Alhim Adonai Vera Gonzalez - UC: M16155349

Introduction

This project is submitted for **CSE6073 - Deep Learning**, taught by Dr. Jun Bai, and is part of **Homework 3**, due on 11**/04/2024**.

The task for this project is to develop a deep learning model to segment retinal blood vessels from eye images. Retinal vessel segmentation is important in medical imaging because it helps doctors detect and monitor diseases like diabetes and hypertension, which can affect the eyes. By creating a model that can accurately identify these vessels, we aim to improve the early detection of eye-related health issues. In this project, we experiment with different configurations of the U-Net model, a popular architecture for image segmentation tasks, to find the best setup for accurate vessel segmentation.

Steps of the Homework:

Step 1: Exploratory Data Analysis (EDA)

- **Training Samples**: 80, **Testing Samples**: 20
- **Task**: Segment retinal blood vessels in fundus images.
- **Image Size**: All images are 512 × 512 pixels.
- **Labels**: Binary masks (1 = blood vessels, 0 = background).
- **Data Split**: 80% train, 20% test (10% val and 10% test)
- **Preprocessing**: Resizing, normalization ([0,1] scale), and data augmentation (rotation, flip, zoom), keeping the original train-test split intact.

Step 2: Data Preprocessing

The `preprocess_data` function is designed to prepare images and masks for training a U-Net model on retinal blood vessel segmentation. Here's a breakdown of the preprocessing steps:

1. **Data Loading and Resizing**:
    - Images and masks are loaded from specified folders and resized to a consistent size (e.g., 128×128 pixels) for uniform input to the model and for faster processing.
    - Images and masks are normalized to a [0, 1] range, which helps improve model training stability. (This normalize step is optional, and we are going to compare the model in the last step)
2. **Train-Test-Validation Split**:
    - The data is split into training, validation, and testing sets, following a pre-defined split. Half of the testing data is further set aside for validation, maintaining the original 80/20 train-test split.
3. **Data Augmentation**:
    - For the training data, optional data augmentation generates additional samples with transformations such as rotation, shifting, zooming, and flipping.
    - This augmentation helps the model generalize better by exposing it to varied versions of the images.
    - To ensure alignment, each augmented image is paired with a corresponding transformed mask.
4. **Output**:
    - The function outputs processed `X_train`, `X_val`, `X_test`, `y_train`, `y_val`, and `y_test` arrays, making them ready for model training and evaluation.

Step 3: Model Training and Hyperparameter Tuning

The table below summarizes the performance of different U-Net configurations used to segment retinal blood vessels in fundus images. In this analysis, the U-Net model is adjusted by changing specific hyperparameters, including the number of layers, number of filters, dropout rate, and learning rate. Each configuration was tested on the dataset, and the model's performance was evaluated using two key metrics: the Dice coefficient and Intersection over Union (IoU). Both metrics were calculated using normalized data to ensure consistency.

The number of layers impacts how deep the network goes, allowing it to capture more complex features. The base filters control the number of features learned in the first layer, which doubles as the layers increase. The dropout rate is a regularization method to prevent overfitting by randomly turning off neurons during training, and the learning rate controls how quickly the model learns.

This table provides an easy comparison of each model configuration's effectiveness in capturing and segmenting blood vessels in retinal images. Adjusting these hyperparameters allows for exploration of the best settings to improve segmentation accuracy.

Model Configuration	IoU	Dice
U-Net 2 layers, filters 32, dropout 0.3, lr 1e-3	0.5400	0.7013
U-Net 3 layers, filters 64, dropout 0.4, lr 1e-3	0.5369	0.6986
U-Net 4 layers, filters 64, dropout 0.5, lr 1e-4	0.3464	0.5145
U-Net 5 layers, filters 64, dropout 0.3, lr 1e-4	0.3941	0.5654

Step 4: Objective

For training the U-Net model, the chosen loss function is **binary cross-entropy**. This loss function is commonly used in binary classification tasks, which aligns well with the segmentation problem at hand, where each pixel is classified as either blood vessel (foreground) or background.

Binary cross-entropy measures the difference between the predicted probability of each pixel belonging to the blood vessel class and the actual label (0 or 1) in the ground truth mask. Minimizing this loss function encourages the model to improve its pixel-wise classification accuracy, making it suitable for precise segmentation tasks like this one. Additionally, using metrics such as the Dice coefficient alongside binary cross-entropy can help evaluate the model's performance in terms of overlap between the predicted and actual segmented areas.

Step 5: Optimization

For training the U-Net models, the selected optimization function is the **Adam optimizer**. Adam is widely used in deep learning due to its adaptive learning rate capabilities and efficiency in handling large datasets and high-dimensional spaces.

Reasons for Choosing Adam:

1. **Adaptive Learning Rates**: Adam automatically adjusts the learning rate for each parameter, making it effective for training complex models like U-Net without requiring extensive manual tuning of the learning rate.
2. **Efficient Convergence**: Adam converges faster than traditional gradient descent optimizers, which is beneficial for segmentation tasks that involve large datasets and intricate models.
3. **Handles Sparse Gradients**: Adam is well-suited for segmentation tasks where gradients can be sparse, as it utilizes momentum and adaptive learning rates to achieve stable updates.
4. **Popular for Image Segmentation**: Adam has been commonly used in medical image segmentation studies, making it a reliable choice for achieving stable and accurate segmentation results in this retinal vessel segmentation task.

By using Adam, the U-Net model is more likely to achieve efficient convergence and stable training, leading to higher performance in segmenting the retinal blood vessels accurately.

Step 6: Model Selection

In this step, we evaluate the performance of different U-Net model configurations. Each model is assessed using both normalized and non-normalized data to understand the impact of data preprocessing on segmentation accuracy. By comparing normalized and non-normalized results, we can observe how the data preparation affects the model's ability to learn and distinguish fine details in medical images.

Model Configuration	Dice and IoU with Normalization	Dice and IoU without Normalization
U-Net 2 layers, filters 32, dropout 0.3, lr 1e-3	Dice: 0.7013, IoU: 0.5400	Dice: 0.5260, IoU: 0.3568
U-Net 3 layers, filters 64, dropout 0.4, lr 1e-3	Dice: 0.6986, IoU: 0.5369	Dice: 0.3918, IoU: 0.2436
U-Net 4 layers, filters 64, dropout 0.5, lr 1e-4	Dice: 0.5145, IoU: 0.3464	Dice: 0.7336, IoU: 0.5793
U-Net 5 layers, filters 64, dropout 0.3, lr 1e-4	Dice: 0.5654, IoU: 0.3941	Dice: 0.7620, IoU: 0.6155

The performance of the models appears to be better without normalization for some configurations, as seen with higher Dice and IoU values. This could be due to the fact that without normalization, the model is learning from a broader range of pixel values, which might better capture the contrasts needed to identify blood vessels. Additionally, the non-normalized data may have preserved more of the original intensity differences, making the model more sensitive to distinguishing features.

We tested multiple learning rates, but the `ReduceLROnPlateau` callback was key to optimizing training dynamically. It reduced the learning rate by half when validation loss stopped improving, helping the model make smaller, precise updates as it neared optimal weights. This approach allowed for faster initial training while ensuring fine-tuning towards the end, improving overall performance and preventing overfitting.

A common question might be: "Does using Adam together with `ReduceLROnPlateau` cause any issues?" The answer is no—these two work together effectively without conflict. Adam's adaptive learning rate mechanism helps the model converge quickly in the early stages of training. When combined with `ReduceLROnPlateau`, which dynamically lowers the learning rate based on validation performance, the model benefits from both rapid initial learning and finer adjustments as it approaches optimal performance. This combination ensures stable, efficient training, allowing the model to balance speed and accuracy without needing manual adjustments to the learning rate across epochs.

To avoid overfitting, we applied dropout layers, EarlyStopping, and data augmentation. Dropout reduces reliance on specific neurons, EarlyStopping prevents excessive training, and data augmentation introduces variability to improve generalization. To prevent underfitting, we adjusted hyperparameters like learning rate, filter count, and model depth, ensuring the model complexity was sufficient to capture important features without oversimplifying.

Step 7 Model Demo:

In this step, we provide visual examples to demonstrate the model predictions across different U-Net configurations. Each image in the grid shows the original input, the ground truth label, and the predicted segmentation from each model configuration. This allows us to compare how accurately each model identifies the retinal blood vessels.

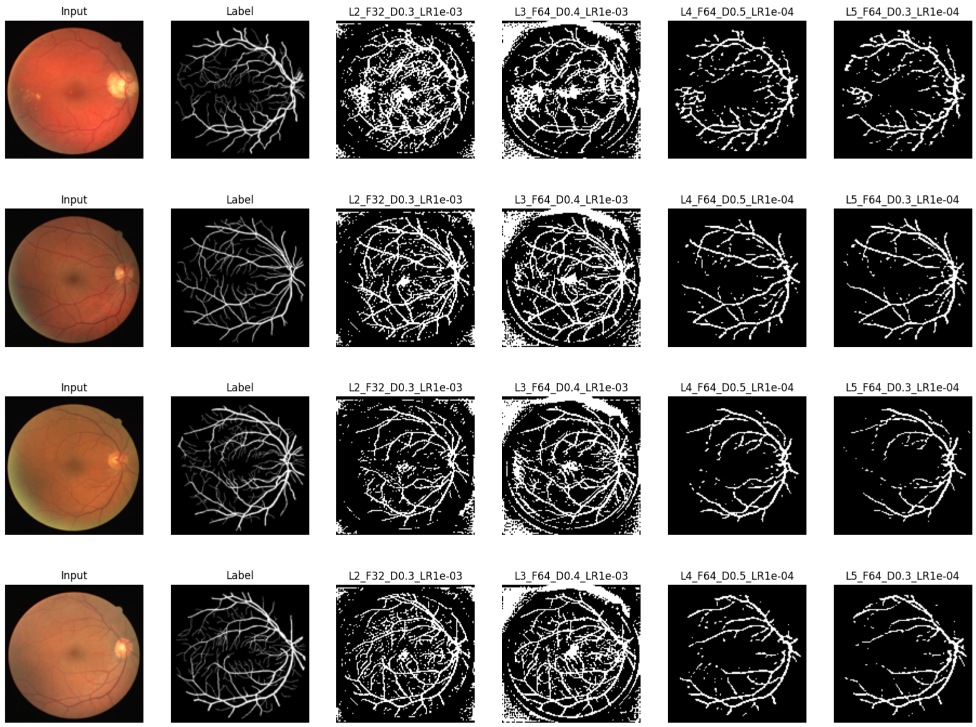

Figure 1: Comparison of model predictions without normalization, showing how each U-Net configuration performs on retinal blood vessel segmentation.

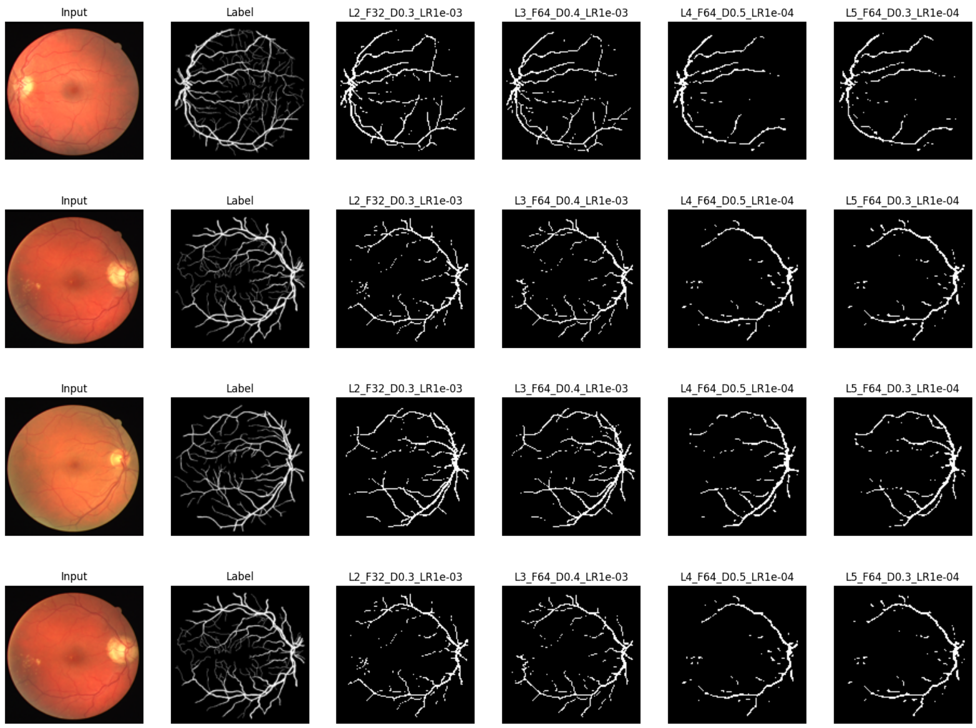

Figure 2: Comparison of model predictions with normalization applied to the dataset, highlighting differences in performance across configurations.

For tasks like retinal blood vessel segmentation, where fine details and small contrast differences are crucial, normalization can sometimes blur these subtle variations.

In our case, the model performed better on non-normalized data. Without normalization, the model retained the small intensity differences between blood vessels and the background, helping it detect vessels more accurately. This is likely because non-normalized data allowed the model to pick up on the slight variations that are critical for identifying vessel structures in retinal images.

Among the tested configurations, the U-Net model with 5 layers, 64 filters, a dropout rate of 0.3, and a learning rate of 1e-4 consistently produced the most accurate segmentation results, as indicated by its higher Dice coefficient and IoU scores without normalization. This configuration showed the clearest and most detailed blood vessel structures, closely matching the labeled ground truth, and therefore stands out as the best-performing model in this setup.

Code Structure:

The main code for this project allows you to run all U-Net model configurations by simply executing `python main.py`. Before running the script, ensure all dependencies are installed by using `pip install -r requirements.txt`. The requirements file is structured to include everything needed for training and evaluating segmentation models.

Additionally, the `test_model.py` script is available for evaluating the best-performing U-Net model on the retinal blood vessel segmentation task. This script loads and preprocesses the dataset, applies the saved model weights, and calculates metrics such as Dice coefficient and IoU. The code is designed for easy testing and performance comparison, making it straightforward to reproduce results or explore further model adjustments.

Repo github: https://github.com/AdonaiVera/retina-segmentation-unet