U-Net Architectures For Automated Retinal Optical Coherence Tomography Segmentation With Pathological Variability

Sivashen Naidoo supervised by Raheleh Kafieh

***Abstract*—Automated segmentation of retinal Optical Coherence Tomography scans is crucial for the quantitative assessment of ophthalmic and neurodegenerative diseases. However, it remains challenged by low inter-layer contrast, pathology-induced distortions, and imaging artifacts. This research project systematically evaluates three convolutional neural network architectures (U-Net, U-Net++, and U-Net 3+) in combination with three loss functions (cross-entropy, Dice, and combined CE–Dice) across varying dataset sizes and training epochs to identify the most robust configuration. Architecture-loss experiments on a subset of the Johns Hopkins OCT dataset demonstrate that U-Net++ with a combined CE–Dice loss achieves the highest scalability and balanced performance. The selected U-Net++ model was then retrained on an expanded dataset and internally validated, achieving a mean Dice coefficient of 0.929, a mean Intersection over Union of 0.870, and a mean boundary Root Mean Square Error of 0.017 —surpassing existing model benchmarks. External validation on the independent NR206 dataset confirms strong generalisability, yielding a mean Dice of 0.916 and a mean IOU of 0.848. A probability error heatmap and signal-error analysis are also introduced in this research project for intuitive visualisation of spatial bias and model confidence, enhancing interpretability and clinical acceptability. The results demonstrate the deep learning model's ability to deliver anatomically precise, high-fidelity segmentation across diverse imaging systems and patient populations, laying the groundwork for large-scale studies of retinal biomarkers in neurodegenerative conditions. Thus, the framework is poised for future integration into routine ophthalmic workflows, with extensions targeting pathology-specific quantification and real-time quality control in clinical deployment.**

***Index Terms*—Optical Coherence Tomography, Retinal Layers, Segmentation, Neurodegenerative Diseases, Deep Learning, U-Net, U-Net++, U-Net 3+.**

I. Introduction

A. Motivation

RETINAL Optical Coherence Tomography (OCT) is vital for diagnosing and managing both ophthalmic conditions and neurodegenerative diseases (NDDs) [1]. NDDs, such as Alzheimer's Disease (AD), Multiple Sclerosis (MS), and Parkinson's Disease, affect the human nervous system and visual pathways, accounting for significant morbidity and mortality worldwide. Each year, approximately 10 million people develop AD, 2.8 million live with MS, and another 10 million are affected by Parkinson's Disease [2]. These disorders are progressive and often debilitating, highlighting the need for early and accurate diagnosis to optimise treatment outcomes. A key indicator of NDD progression is the thinning of specific retinal layers, observable through OCT imaging. However, detecting these subtle changes requires precise segmentation of retinal layers. Manual annotation by clinicians remains the gold standard, yet it is labour-intensive, susceptible to inter-observer variability, and impractical for widespread, large-scale clinical deployment. Deep learning solutions, notably U-Nets and its variants, have shown considerable promise in automating the segmentation process, offering the potential to streamline workflows and enhance diagnostic accuracy. However, several obstacles currently limit the broader clinical adoption of deep learning-driven OCT segmentation. Low inter-layer contrast, pathology-induced anatomical distortions, imaging artifacts, and device- or dataset-specific variations can degrade model performance, making them unreliable. Overcoming these challenges is crucial for improving the robustness and generalisability of segmentation algorithms in real-world clinical settings.

B. Aim

This research project aims to develop a robust and generalisable deep-learning model for retinal OCT segmentation and boundary delineation that seamlessly integrates into clinical workflows. To achieve this, we aim to evaluate and compare three convolutional neural network architectures (U-Net, U-Net++, and U-Net 3+) combined with different loss functions to address the challenges in OCT segmentation, including class imbalance, boundary ambiguity, OCT image inconsistencies, and variability across pathological conditions. By identifying the architecture–loss function pairing that consistently delivers high segmentation performance, this research seeks to advance the clinical deployment of automated OCT analysis tools. The ultimate objective is to produce a clinically viable model that reduces the burden and reliance on manual annotations, enhances diagnostic accuracy, and facilitates adopting automated OCT analysis in routine ophthalmic practice.

II. Background And Theory

A. Retina OCT Scans

OCT is a non-invasive imaging technique that stores cross-sectional retinal data in vendor-specific raw volumetric ("`.vol`") files. Retinal OCT `.vol` files are generated by first acquiring A-scans—depth-resolved reflectivity profiles sampling the axial (depth) direction via near-infrared light and interferometric detection [3]. As the beam scans laterally along the temporal–nasal axis, successive A-scans are compiled into 2D cross-sections (B-scans). By sweeping this scan across the

retina, a dense 3D volume is reconstructed and saved in a `.vol` file [4]. Individual B-scans provide detailed visualisation for retinal layers and structure; in a right-eye scan, Point A (temporal outer) lies toward the temple and Point B (nasal inner) toward the nose as illustrated Figure 1)—these positions reverse in a left-eye scan.

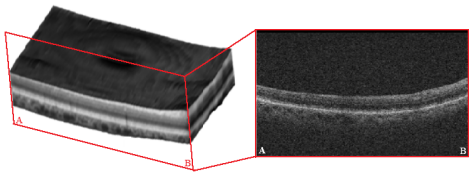

Fig. 1. B-scan extracted from a right eye retina .vol file, with Points A and B denoting temporal and nasal positions, respectively, along the horizontal meridian represented by the red box's horizontal extent.

B. Deep Learning

Deep learning, a subset of machine learning, utilises multi-layered neural networks to model complex data patterns and mimic aspects of human decision-making. Unlike traditional, shallower models, deep learning employs architectures with potentially hundreds of layers to process raw, unstructured data through supervised and unsupervised learning. Deep neural networks consist of layers of interconnected nodes (neurons) that progressively refine predictions via forward propagation, where input data flows through the network to generate outputs. The input and output layers mark the entry and exit points of data. Backpropagation complements forward propagation using optimisation algorithms like gradient descent to minimise prediction error. It calculates the loss—the difference between predicted and actual outputs—and adjusts the network's weights and biases by propagating this error backwards through the layers. This feedback loop allows the model to learn iteratively and improve its accuracy over time. Deep learning supports advanced applications in medical technologies such as image analysis, disease detection/diagnosis, and personalised treatment. Despite its high accuracy, scalability, and robust data handling capabilities, it faces criticism for the "black box" issue due to the limited interpretability of model decisions [5]. Training deep learning models typically requires significant computational resources, often utilising GPUs or cloud-based systems. Key frameworks include TensorFlow, PyTorch, and JAX.

C. Convolution Neural Network

Computer vision applies neural networks to image or video data, where a colour image is treated as a 3D volume (height × width × channels). Even modest resolutions carry many input features—for example, a 256 × 256 RGB image has 256 × 256 × 3 = 196 608 input values—so dimension-reduction via feature extraction is essential. Convolutional Neural Networks (CNNs) excel here by exploiting local connectivity and weight sharing. A typical CNN begins with an input layer matching the image volume. Next, come one or more convolutional layers: each applies a bank of small, learnable kernels (commonly 3×3 or 5×5) that slide across the input with a specified stride and may use padding to control output size. Each kernel produces one feature map (activation map), encoding the response to that filter at every spatial position. Non-linear activations (e.g., ReLU) follow each convolution, allowing the network to learn complex patterns. Optionally, a pooling layer (such as max- or average-pooling) downsamples the feature maps, reducing spatial dimensions and helping guard against overfitting. Finally, the reduced-volume features are fed into one or more dense (fully connected) layers or a global pooling layer, which assemble high-level abstractions for tasks like classification. All weights—including kernel parameters and dense-layer weights—are optimised end-to-end via backpropagation.

D. U-Net

U-Net is a CNN architecture proposed by Ronneberger et al. [6], 2015, primarily designed for biomedical image segmentation. It is widely used for pixel-level tasks due to its proficiency in capturing global context and finer details. The U-Net architecture features a symmetric encoder-decoder

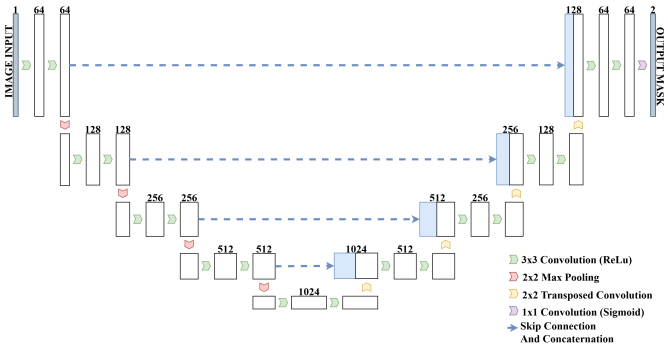

Fig. 2. The U-Net Architecture comprising of an encoder and a decoder pathway connected by a bottleneck, with skip connections between the corresponding layers.

structure that effectively captures complex features while preserving spatial information. The architecture is composed of two main components: the encoder (contracting path) and the decoder (expanding path) connected by a bottleneck, as illustrated in Figure 2. The encoder consists of a series of convolutional layers, each followed by a max-pooling layer. Each convolutional layer (cl) applies a convolution operation using filters of size $k \times k$, using optional padding to preserve the spatial dimensions. The output of each convolutional block is then passed through a non-linear activation function, typically a Rectified Linear Unit (ReLU). The output of a convolutional layer is expressed through Equation (1).

$$Y = \sigma(W * X + b) \tag{1}$$

where W is the convolutional kernel, b is the bias, $*$ is the convolution operation, σ is ReLU and $X \in \mathbb{R}^{h \times w \times c}$ is the input image where h, w, and c represent the height, width, and number of channels respectively. After each convolution, a $m \times m$ max-pooling operation is applied to reduce the spatial resolution by a factor of m. The bottleneck serves as the bridge between the encoder and decoder. It consists of cl, $k \times k$ convolutions, followed by an activation function. This layer captures the deepest level of feature representations with the smallest spatial dimension. The decoder mirrors the encoder as it is structurally symmetrical consisting of upsampling operations followed by convolutional layers. The upsampling operation doubles the spatial resolution and can

be implemented using either transposed convolutions (also referred to as deconvolutions) or interpolation techniques such as bilinear or nearest-neighbour interpolation. This process is defined by Equation (2).

$$Z = W^T * Y \tag{2}$$

where W^T is the transposed convolution kernel and Y is the input from the previous layer. Following each upsampling step, the corresponding feature map from the encoder is concatenated with the decoder feature map to preserve spatial information. This is known as a skip connection, and it is crucial for retaining finer details that are often lost during the downsampling process. Each concatenated feature map is subsequently passed through cl, $k \times k$ convolutions followed by an activation function. This progressively reconstructs the spatial resolution while refining the feature map. The output layer is the final component of the U-Net architecture. It is a 1×1 convolution which reduces the number of channels to the desired number of output classes. For segmentation tasks, the softmax function is usually applied afterwards to produce a probability distribution over the classes for each pixel, enabling precise pixel-wise classification. The softmax function is expressed by Equation (3).

$$P(c_i \mid X) = \frac{\exp(s_i)}{\sum_{j=1}^{C} \exp(s_j)} \tag{3}$$

where s_i is the score for class i and C is the total number of classes. Loss functions are then used to quantify the difference between the predicted segmentation mask and the ground truth (GT) mask, guiding the network's learning during training. During backpropagation, the computed loss is used to update the network's weights, gradually improving pixel-wise accuracy and overall segmentation performance.

E. U-Net++

U-Net++ is a variant of the U-Net architecture, proposed by Zhou et al. [7], 2018. It aims to improve semantic segmentation performance by redesigning skip connections and introducing dense convolutional blocks between the encoder and decoder. Through nested convolutional pathways, it aims

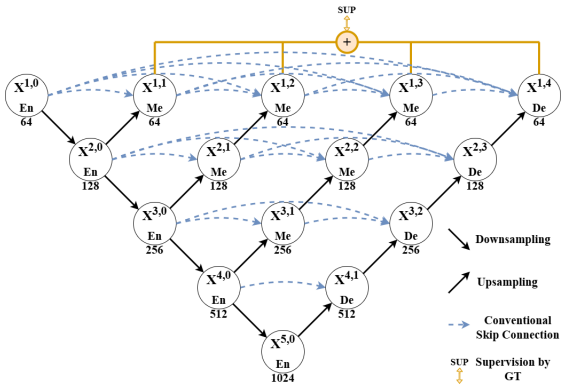

Fig. 3. U-Net++ architecture diagram showing the nested and dense skip connections between encoder and decoder sub-networks.

to achieve higher accuracy that facilitates feature refinement and multi-scale feature fusion. U-Net++ architecture retains the core encoder-decoder structure of the U-Net while introducing significant modifications in the skip connections. Specifically, it incorporates nested convolutional pathways, comprised of a series of convolutional blocks connecting encoder and decoder layers at various depths, as illustrated in Figure 3. Skip connections in a U-Net++ are redefined to include convolutional blocks that progressively refine features before merging them with the corresponding decoder features. Let $X^{i,j}$ denote the feature map at the i-th decoder stage and j-th convolutional layer within the nested pathway. Hence the feature maps are computed recursively by using Equation (4).

$$X^{i,j} = \begin{cases} f\left(X^{i-1,j},\ \mathrm{Up}(X^{i,j-1})\right), & \text{if } j > 0 \\ f\left(X^{i}_{\text{enc}}\right), & \text{if } j = 0 \end{cases} \tag{4}$$

where X^{i}_{enc} is the output feature map from the i-th encoder layer, $f(\cdot)$ represents a convolutional operation, and $\mathrm{Up}(\cdot)$ is an upsampling operation to match the spatial dimensions. The nested pathways in U-Net++ enable the network to aggregate features across multiple different semantic scales, effectively bridging the semantic gap between encoder and decoder features. This design enhances the network's representational capacity by facilitating deeper supervision and more precise feature alignment. Each decoder node $X^{i,j}$ is connected not only to its corresponding encoder feature map X^{i}_{enc} but also to all preceding decoder nodes $X^{k,j-1}$ where $k < i$. This dense connectivity forms a full convolutional block between encoder and decoder stages, encouraging extensive feature reuse and refinement allowing more effective information flow across different levels of abstraction. The final output of the network is attained from the deepest decoder layer after applying a final convolutional layer to map the feature maps to the desired number of segmentation classes.

F. U-Net 3+

U-Net 3+ is a version of the U-Net and U-Net++ architectures, introduced by Huang et al. [8], 2020, to improve multi-scale feature fusion and segmentation accuracy, particularly for pixel-wise prediction tasks. It introduces two key innovations innovations being full-scale skip connections and deep supervision. These modifications enable the integration of feature information across all encoder and decoder layers, allowing the network to capture high-level semantic information and fine-grained spatial details more effectively. The U-Net 3+ architecture maintains the basic encoder-decoder structure of U-Net similarly to the U-Net++ but redefines the skip connections differently. In U-Net 3+, each decoder level aggregates feature maps from all encoder levels through full-scale skip connections. This design allows the fusion of features from multiple resolutions enhancing the network's ability to capture both global context and fine details, as illustrated in Figure 4. The full-scale skip connections in the U-Net 3+ are used to aggregate feature maps from all encoder layers into each decoder layer. Let $Z_{i,j}$ represent the feature map at the i-th level of the decoder after fusion with the encoder outputs for level j. Therefore, The full-scale skip connections are defined by Equation (5).

$$Z_{i,j} = \mathrm{concat}(E_0, E_1, \ldots, E_N, D_{i-1,j}) \tag{5}$$

where E_k is the feature map from the k-th encoder level, $k \in \{0, 1, \ldots, N\}$, $D_{i-1,j}$ is the upsampled feature map from the previous decoder level $(i-1)$, and concat is the concatenation operation across all encoder features and the corresponding decoder feature map. This concatenation allows each decoder layer to access and integrate information from multiple resolution levels, improving the model's ability to capture varying features and refining segmentation precision. The U-Net 3+ also incorporates a deep supervision mechanism by applying auxiliary output layers to intermediate decoder

levels. For each decoder level D_i, an auxiliary output $X^{(i)}_{\text{output}}$ is generated, and a corresponding loss is calculated. The total loss function, $\mathcal{L}_{\text{total}}$, combines the individual losses from all decoder levels, encouraging learning across multiple scales. The total loss function is expressed by Equation (6).

$$\mathcal{L}_{\text{total}} = \sum_{i=1}^{M} \lambda_i \mathcal{L}\left(X^{(i)}_{\text{output}},\ Y\right) \tag{6}$$

where Y is the GT segmentation map, $\mathcal{L}$ is the segmentation loss function, $X^{(i)}_{\text{output}}$ is the predicted segmentation map at the i-th decoder level, and λ_i are the weights for each decoder level's contribution to the total loss. The deep supervision mechanism ensures that each decoder layer is individually optimised, enhancing multi-scale feature learning and improving the final segmentation output. The model's final segmentation map is obtained by aggregating the supervised outputs from all decoder levels, allowing the model to leverage multi-resolution information.

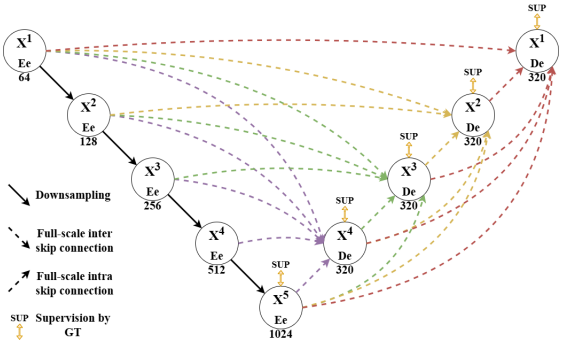

Fig. 4. U-Net 3+ architecture diagram showing full-scale skip connections that aggregate multi-scale features from encoder and decoder paths.

G. Activation Maps And Heatmaps

Neural networks such as U-Nets are often called 'black boxes' because understanding the non-linear computations within their hidden layers becomes increasingly difficult as model complexity grows [5]. To address this opacity, activation mapping techniques provide visualisations highlighting which parts of an input sample the model considers most relevant to its prediction [9]. In recent years, a new field of research—Explainable Artificial Intelligence (XAI)—has emerged, aiming to enhance the interpretability of deep learning models. Heatmaps have become popular tools for visualising and interpreting how neural networks process and prioritise image features within this domain. One of the earliest visualisation methods, saliency maps, was introduced by Simonyan et al. [9] in 2013. These maps are generated by back-propagating gradients from a model's output to its input, thereby identifying input regions most influential to the prediction. This gradient-based approach was later refined into Class Activation Maps (CAM), introduced by Zhou et al. [10] in 2016. CAMs work by weighting and visualising the activations of convolutional layers, offering a class-specific interpretation of the network's focus. Although CAMs are predominantly used in classification tasks to highlight class-discriminative regions, this project takes a different route. Instead of employing CAM-based localisation, a probability error heatmap is utilised. This approach visualises the discrepancy between predicted probabilities and ground truth values, providing insight into areas where the model's confidence is misplaced or uncertain—particularly useful in segmentation tasks. This heatmap is generated by first computing the softmax probabilities across all classes for each pixel in the image. The predicted class at each pixel is then obtained by taking the `argmax` over these probabilities. To focus specifically on model misclassification, we identify only those pixels where the predicted class differs from the ground truth (GT) label. For each of these misclassified pixels, we record the model's confidence in its incorrect prediction—this is the softmax probability assigned to the incorrectly predicted class. The resulting heatmap, therefore, visualises regions where the model is confidently wrong, as formally defined in Equation (7).

$$H(i,j) = I\left(\hat{y}(i,j) \neq y(i,j)\right) \cdot P_{\hat{y}(i,j)}(i,j) \tag{7}$$

where i and j denote the row (vertical) and column (horizontal) indices of a pixel respectively, $\hat{y}(i,j)$ is the predicted class label at pixel (i,j), $y(i,j)$ is the GT label at pixel (i,j), $P_{\hat{y}(i,j)}(i,j)$ is the softmax probability assigned to the predicted class at pixel (i,j), and $I(\cdot)$ is the indicator function which equals 1 when its argument is true (i.e. when the prediction is incorrect). This formulation enables the visualisation of the model's confidence in its incorrect predictions. Higher heatmap values indicate greater confidence in misclassification. These values, which range from 0 to 1, are mapped to a jet colourmap: red indicates high-confidence misclassification (values close to 1), while dark blue corresponds to correct classifications or low-confidence errors (values near 0). Intermediate colours (such as yellow, green, and cyan) form a gradient that visually represents varying levels of error. By integrating this probability error heatmap into the model's visualisation pipeline, we create a concise yet powerful diagnostic tool. It pinpoints not only where misclassification occurs but also how confident the model was in making those errors. This supports more targeted and precise error analysis, ultimately aiding in the evaluation and improvement of segmentation models.

H. Root Mean Square Error and Signal Errors

The accuracy of retinal boundary delineation of the model is assessed using two complementary metrics a normalised root-mean-square error (RMSE) and a column-wise signal error (SE). The normalised boundary difference is expressed by Equation (8).

$$\Delta b(x) = \frac{\hat{b}(x) - b(x)}{H} \tag{8}$$

where $\hat{b}(x)$ represents the predicted boundary location at column x, $b(x)$ is the GT boundary location at the same column, and H denotes the image height in pixels. The overall segmentation error is quantified using this through the RMSE. The RMSE is defined by Equation (9).

$$\text{RMSE} = \sqrt{\frac{1}{N}\sum_{x=1}^{N}\left(\Delta b(x)\right)^2} \tag{9}$$

where N is the total number of columns in the image. This metric aggregates the squared vertical differences between the predicted and GT boundaries across the image width, thereby providing a single scalar value that reflects overall boundary misalignment. In contrast, the SE is obtained by converting the normalised differences into percentage errors. The SE is given by Equation (10).

$$\Delta b_{\%}(x) = \frac{\hat{b}(x) - b(x)}{H} \times 100 \tag{10}$$

When computed over multiple samples and averaged, this resulting column-wise discrepancy produces a SE that highlights where the model systematically over- or under-segments the boundary. In the context of retina .vol files generated from B-OCT scans, this SE shows percentage differences in boundary delineation between the model prediction and GT along the depth dimension, enabling a volumetric assessment of segmentation and delineation performance. This spatially resolved profile not only captures local variations in segmentation and delineation accuracy but also identifies specific regions along the lateral dimension where improvements may be necessary for the model. Together, the RMSE and SE offer complementary insights into model performance. The RMSE serves as a concise global metric for benchmarking and comparing models, while the SE acts as a diagnostic tool, exposing localised errors in boundary segmentation and delineation. This dual-metric strategy enhances evaluation robustness by combining an overall performance measure with fine-grained spatial analysis—especially valuable in high-resolution, structure-sensitive domains like retinal imaging.

III. Methodology

A. Outline

The proposed method evaluates multiple segmentation model frameworks with identical preprocessing steps but varying the architecture and loss function. The models are trained using varying numbers of epochs and dataset sizes and validated using the same validation dataset in experiments for each possible architecture-loss and parameter combination. The top-performing model framework from experiments is then retrained on an expanded dataset to develop a final model internally validated on a held-out test set with pathological variability (MS-affected retinas). The model is externally validated on an independent OCT dataset acquired from a different scanner to assess robustness and generalisability. All experiments, as well as training, validation, and testing of the final model, are conducted in the Google Colab's paid runtime environment, using the NVIDIA A100 GPU (40 GB VRAM), with Python 3.10 and PyTorch 1.13, to ensure reproducibility and scalability.

B. Datasets

The publicly available dataset from Johns Hopkins University [11] was used for model training and internal validation, as described in Appendix-B. A custom data loader was developed to handle this dataset's specific format and structure. For external validation, the NR206 dataset [12] was used, also described in Appendix-B, with a separate custom data loader designed to accommodate its unique structure.

C. Preprocessing

To ensure fair testing, each model uses the same preprocessing pipeline. Each retina OCT image is first converted to a grayscale intensity function, $I(x, y)$, and resized to a standard dimension of 1024×256 pixels to ensure uniformity across samples. To enhance local contrast, we apply Contrast Limited Adaptive Histogram Equalisation (CLAHE) and subsequently perform Z-score normalisation, yielding a transformed image that can be expressed by Equation (11).

$$I_{\text{norm}}(x, y) = \frac{I_{\text{CLAHE}}(x, y) - \mu}{\sigma} \tag{11}$$

where μ and σ denote the mean and standard deviation of the CLAHE-processed image. The preprocessed images then undergo a series of probabilistic data augmentations to increase model robustness. A random horizontal flip is applied with a probability of 0.5. Brightness and contrast are perturbed by adding a shift β, drawn uniformly from the interval $[-0.2, 0.2]$, and scaling by a factor γ, drawn from a uniform distribution over the range $[0.8, 1.2]$, respectively. Elastic deformations are induced by generating displacement fields through Gaussian-filtered random noise, where the magnitude and smoothness of the deformation are controlled by parameters $\alpha = 34$ and $\sigma = 4$, respectively. Additionally, simulated artifacts—such as random occlusions—are introduced with a set probability to mimic imaging imperfections in OCT scans. These preprocessing and augmentation steps standardise the data while accounting for variability in real-world OCT imaging conditions, ultimately enhancing the segmentation performance of the model on the retinal layer boundaries.

D. Training

Cosine Annealing Learning Rate Scheduler (CosineAnnealingLR) introduced by Ilya Loshchilov and Frank Hutter [13] in 2017, is employed to adjust the learning rate (LR) dynamically throughout the training process according to the number of epochs. The LR using CosineAnnealingLR at a given epoch is defined by Equation (12).

$$\eta(t) = \eta_{\text{min}} + \frac{1}{2}(\eta_{\text{max}} - \eta_{\text{min}}) \left(1 + \cos\left(\frac{t}{T_{\text{max}}}\pi\right)\right) \tag{12}$$

where t is the epoch, $\eta(t)$ is the LR at t, η_{min} is the maximum learning rate, η_{min} is the minimum learning rate, and T_{max} is the total number of epochs. For the architecture-loss experiments, η_{min} and η_{min} are set to 1×10^{-3} and 1×10^{-5} respectively. During training, the Adam optimiser was optimised using a weight decay of 1×10^{-5} to regularise the model and reduce overfitting. Due to memory constraints in the Google Colab environment, the batch size was adjusted according to model complexity: a batch size of 8 was used for U-Net and U-Net++, while U-Net 3+, which has higher memory demands, used a reduced batch size of 4. Segmentation performance was evaluated after each training epoch using key metrics—Dice coefficient, Intersection over Union (IoU), Precision, and Recall—to assess the segmentation quality and computational efficiency.

E. Evaluation Metrics

Four standard metrics were used: Dice Coefficient, IoU, Precision, and Recall [14] to evaluate segmentation accuracy. All four evaluation metrics are based on the four outcomes from the confusion matrix for a binary segmentation mask: True Positive (TP), False Positive (FP), False Negative (TN), and True Negative (FN). The Dice coefficient measures the overlap between the predicted segmentation mask and the ground truth mask. It is calculated as twice the size of their intersection divided by the sum of the total number of pixels

in both masks. Each class's score is computed separately, and the average Dice score is reported. Hence, the Dice Coefficient is given by Equation (13).

$$\text{Dice} = \frac{2TP}{2TP + FP + FN} \tag{13}$$

The IoU, also known as the Jaccard Index, measures the similarity between the predicted segmentation and the ground truth. It is defined as the size of the intersection divided by the size of the union of the predicted and ground truth masks. Thus, IoU is expressed by Equation (14).

$$\text{IoU} = \frac{TP}{TP + FP + FN} \tag{14}$$

Precision measures the proportion of all the model's positive classifications that are truly positive. It is calculated by dividing the number of TPs by the sum of TPs and FPs, as defined by Equation (15).

$$\text{Precision} = \frac{TP}{TP + FP} \tag{15}$$

Recall, also known as the True Positive Rate, measures the proportion of actual positive values identified correctly. It is calculated by dividing the number of TPs by the sum of TPs and FNs, as expressed through Equation (16).

$$\text{Recall} = \frac{TP}{TP + FN} \tag{16}$$

F. Loss Functions

Semantic segmentation loss functions fall into different categories based on their focus and objectives. Pixel-level loss functions evaluate segmentation accuracy by individually comparing each predicted pixel with its corresponding ground truth label. These functions measure the discrepancy at the pixel level, ensuring each pixel is classified as accurately as possible within the segmented regions. CE is a pixel-level loss function that measures the difference between two probability distributions for a given random variable. It evaluates how closely the model's predictions align with the ground truth labels in segmentation tasks. By applying the softmax function, the model produces pixel-wise probability maps indicating the likelihood of each pixel belonging to each class. The CE loss is then computed by taking the negative logarithm of the predicted probability assigned to the correct class at each pixel. The CE loss approaches zero as the predicted probability for the target class approaches 1. Thus, CE loss is expressed by Equation (17) [15].

$$\mathcal{L}_{\text{CE}}(\mathbf{y}, \mathbf{t}) = -\sum_{n=1}^{N} \log(t_n \cdot y_n) \tag{17}$$

where N is the number of pixels, t_n is the one-hot encoding vector representing the target class of the n$^{\text{th}}$ pixel, and y_n is the predicted class probabilities for the n$^{\text{th}}$ pixel. Region-level loss functions, in contrast, focus on the overall class segmentation by maximising the alignment between the predicted segmentation mask and the ground truth mask. Rather than evaluating individual pixels, these functions prioritise the overlap between regions, aiming to enhance object-level segmentation performance. The Dice loss is a region-level loss function originating from the Dice Coefficient defined in Equation (13). It was introduced by Milletari et al. [16] in 2016 and is a differentiable approximation of the Dice Coefficient. It is computed separately for each class, and the average across all classes is used as the final loss. Unlike hard binary predictions (0 or 1), the Dice loss uses soft predictions in the range [0, 1], allowing it to be optimised via gradient-based methods. To convert the similarity measure into a loss function, the relaxed Dice Coefficient is subtracted from 1. This loss is especially effective for imbalanced datasets, emphasising the overlap between predicted and ground truth masks, encouraging the model to segment minority classes accurately. Therefore, Dice loss is defined by Equation (18).

$$\mathcal{L}_{\text{dice}} = 1 - \frac{1}{C} \sum_{c=0}^{C-1} \frac{2 \sum_{n=1}^{N} t_n^c y_n^c}{\sum_{n=1}^{N} (t_n^c + y_n^c)} \tag{18}$$

where C is the number of target classes. The combo approach integrates elements from pixel-level and region-level loss functions to optimise semantic segmentation performance. Combining multiple loss functions balances pixel-wise accuracy and object-level segmentation quality. This approach offers flexibility and adaptability, leveraging the strengths of each loss category to better address the varied challenges posed by different segmentation tasks and dataset characteristics. In semantic segmentation, a common approach to address the class imbalance and improve performance is to combine Dice loss with CE loss, resulting in the Combo Loss [17]. The cross-entropy loss provides pixel-wise supervision and ensures smooth gradient flow during training, while the Dice loss enhances segmentation accuracy, particularly for smaller or less represented structures. Moreover, the Dice component helps the model avoid suboptimal local minima. The Combo Loss combines both terms using a modulating factor to balance their contributions. Hence, Combo loss is given by Equation (19).

$$\mathcal{L}_{\text{combo}} = \alpha \cdot \mathcal{L}_{\text{CE}} + (1 - \alpha) \cdot \mathcal{L}_{\text{dice}} \tag{19}$$

where $\alpha \in [0, 1]$ is the modulating factor that controls the contribution of each loss component.

G. Boundary Delineation

Boundary delineation is achieved by analysing the segmentation mask generated by the model. A boundary detection algorithm is applied to the segmentation output in the backend. This algorithm processes the mask column by column in a nasal-to-temporal direction to detect changes in pixel labels that signify transitions between different retinal layers. For each class i (ranging from the background through the successive retinal layers), the algorithm identifies the first pixel in each column where the label transitions from i to $i + 1$. These transition points are recorded as the boundary for class i in that particular column. Once transition points have been identified across all column indices, they are connected to form continuous lines that trace the contours of each retinal layer, thereby defining the anatomical boundaries. This boundary extraction process is heavily dependent on the quality of the segmentation. Any noise or misclassification within the label map can lead to inaccurate boundary detection. Consequently, producing a high-quality, refined segmentation is crucial to ensure that the anatomical structure of retinal boundaries is accurate and precise across the full nasal-to-temporal extent of the OCT.

IV. Experiment

A. Experimental Setup

For architecture–loss experiments, we used a randomly split subset of the Johns Hopkins dataset comprising 16 subjects (8 healthy controls (HC) and 8 multiple sclerosis (MS) patients:

4 HC and 4 MS for training, 2 HC and 2 MS for testing, and 2 HC and 2 MS for validation. After identifying the optimal architecture–loss combination from experiments, the model was retrained on a larger subset of 28 subjects (14 HC, 14 MS), again randomly assigning 10 HC and 10 MS to training, 2 HC and 2 MS to testing, and 2 HC and 2 MS to validation. In both phases, splits were performed at the subject level, maintaining an equal HC–MS ratio to minimise class-imbalance bias and to ensure the model could accurately segment healthy and unhealthy retinas using as much of the overall dataset as possible.

B. Architecture-Loss Experiments

U-Net-based architectures (U-Net, U-Net++, and U-Net 3+) were systematically evaluated using three loss functions—CE, Dice, and Combo Loss—on a smaller subset of the Johns Hopkins retinal OCT dataset. These models were trained under four resource regimes (25 vs. 50 epochs, and 50% vs. 100% of the dataset) to identify the optimal configuration for OCT segmentation. In the first experiment, U-Net models with each loss function were evaluated, as shown in Figure 5. U-

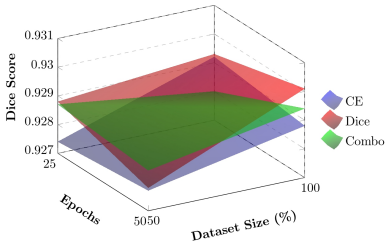

Fig. 5. A 3D surface plot of test set Dice scores from a U-Net architecture across different epochs and dataset sizes, using CE, Dice, and Combo loss functions respectively.

Net with Dice loss achieved the highest mean Dice score (mDice) of 0.9290 (min = 0.9274, max = 0.9301, Δ = 0.0027), peaking at 0.9301 under the most resource-intensive condition (50 epochs, 100% data). Notably, it improved by +0.0023 when increasing data from 50% to 100% at 50 epochs. In contrast, U-Net with CE loss achieved a mDice of 0.9283 (min = 0.9274, max = 0.9292, Δ = 0.0018), with modest scalability response (+0.0005 from doubling data at 25 epochs and –0.0004 from extended training on 100% data). Combo loss yielded a mean of 0.9286 (min = 0.9277, max = 0.9294, Δ = 0.0017)—the tightest spread among all—but slightly underperformed Dice loss overall. U-Net++ was evaluated with the same loss functions in the second experiment, as shown in Figure 6. The Combo loss model exhibited the strongest sensitivity to resource scaling, achieving a mDice of 0.9285 (min = 0.9270, max = 0.9299, Δ = 0.0029), with improvements of +0.0029 when doubling data at 50 epochs and +0.0009 from extended training. The Dice loss model achieved a comparable mean of 0.9290 (min = 0.9278, max = 0.9296, Δ = 0.0018) but demonstrated limited scalability, with just +0.0008 gained from more data at 25 epochs and a performance decline of –0.0009 under extended low-data training. The CE model achieved a mean of 0.9284 (min = 0.9275, max = 0.9294, Δ = 0.0019), improving by +0.0016

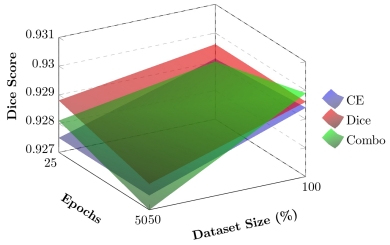

Fig. 6. A 3D surface plot of test set Dice scores from a U-Net++ architecture across different epochs and dataset sizes, using CE, Dice, and Combo loss functions respectively.

with increased data at 25 epochs and +0.0003 from longer training. In the third experiment, results for U-Net 3+ models are presented in Figure 7. The Dice loss model achieved a mDice of 0.9280 (min = 0.9263, max = 0.9300, Δ = 0.0037), peaking at 0.9300. It exhibited a cumulative improvement of +0.0047: +0.0027 from doubling data at 25 epochs, +0.0010 from more epochs at 50%, and another +0.0010 with complete data. Combo loss achieved a mean of 0.9284 (min = 0.9271, max = 0.9294, Δ = 0.0023), showing substantial initial gains (+0.0019 at 25 epochs and +0.0023 at 50 epochs), though performance slightly regressed (–0.0004) under extended low-data training and plateaued at full scale. The CE model had the lowest mDice (0.9274, min = 0.9248, max = 0.9291, Δ = 0.0043) but showed its most significant gain (+0.0043) from increased data at 25 epochs, followed by a +0.0020 boost from longer training, and a minor decline (–0.0003) at full scale. Across all three experiments, U-Net++ with

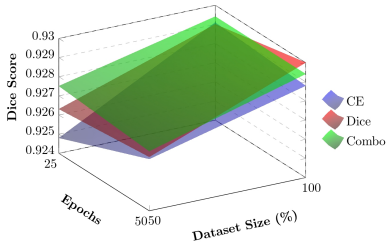

Fig. 7. A 3D surface plot of test set Dice scores from a U-Net 3+ architecture across different epochs and dataset sizes, using CE, Dice, and Combo loss functions respectively.

Combo loss demonstrated the best scalability and sensitivity to resource scaling. Although its absolute peak mDice (0.9299) was slightly lower than U-Net with Dice (0.9301) and U-Net 3+ with Dice (0.9300), it showed the highest overall scalability response—an aggregate gain of +0.0038 (+0.0029 from data scaling and +0.0009 from extended training)—surpassing U-Net with Dice (+0.0023) and marginally exceeding U-Net 3+ with Dice (+0.0037). Given its consistent performance, tight variability, and strong adaptability to increased training resources, U-Net++ with Combo loss emerges as the most balanced and promising candidate for larger-scale segmentation tasks under extended training regimes.

C. Final Model And Fine-Tuning

Building on the architecture-loss experiments, we conducted further trials with U-Net++ using Combo loss on the expanded Johns Hopkins subset focused on fine-tuning the model. We

systematically varied key hyperparameters before training to identify the optimal configuration, looking at performance metrics, heatmaps and signal errors as indications for segmentation performance. The final hyperparameters used to train the final U-Net++ model from this process are tabulated in Appendix-C in Table III which resulted in training and validation curves in Figure 8. From Figure 8, it is observed that

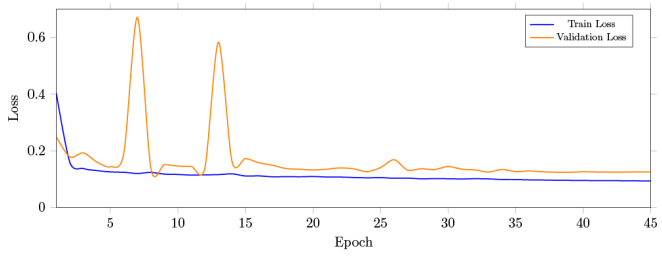

Fig. 8. Learning curves for the final model showing training loss and validation loss over 45 epochs.

training loss fell from 0.403 to 0.094 (76.7%), while validation loss declined from 0.249 to 0.125 (49.8%) over 45 epochs during training. Except for two transient validation spikes at epochs 7 and 13 (likely noise-driven), both curves dropped monotonically, flattening into a plateau around epoch 38, signalling convergence. The minimum validation loss (0.124) occurs at epoch 39; subsequent epochs further reduce training loss but widen the validation-training gap to approximately 0.030, indicating slight over-fitting. Nevertheless, the tight, parallel descent and absence of sustained divergence suggest a well-fitted model. Consequently, the epoch-39 checkpoint was retained as the final model. It was internally validated on the held-out test set and externally validated on the NR206 dataset to quantify segmentation performance on unseen data.

V. Results and Discussion

A. Internal Validation

Internal validation of the final model was performed using the test set from the expanded subset of the Johns Hopkins dataset, with results tabulated in Table I. Traversing the retina from the vitreous to the vascular choroid across all ten segmented retinal tissue classes (vitreous, nine retinal layers, and the choroid), the model reproduced each with high fidelity as shown by the sample output in Appendix-D in Figure 13. Segmentation of the vitreous/background is nearly

TABLE I
Final Model Performance Metrics on the Johns Hopkins Dataset

Metric	Vitreous	RNFL	GCL+IPL	INL	OPL	ONL	IS	OS	RPE	Choroid	Average
Dice	0.9903	0.9387	0.9555	0.8879	0.9067	0.9543	0.8849	0.8887	0.9107	0.9738	0.9292
IoU	0.9808	0.8845	0.9149	0.7984	0.8294	0.9126	0.7936	0.7996	0.8361	0.9489	0.8699
Precision	0.9870	0.9434	0.9553	0.8905	0.9136	0.9516	0.8852	0.9418	0.8850	0.9711	0.9325
Recall	0.9936	0.9341	0.9558	0.8853	0.9000	0.9570	0.8846	0.8413	0.9380	0.9765	0.9266
RMSE	N/A	0.0093	0.0127	0.0134	0.0149	0.0120	0.0086	0.0066	0.0096	0.0088	0.0107

perfect, with an IoU of 0.981 and a Dice score of 0.990. In the innermost neural tissue, the retinal nerve fibre layer (RNFL) achieves an IoU of 0.885 and a Dice score of 0.939. The ganglion-cell + inner-plexiform complex (GCL + IPL) attains even higher performance, with an IoU of 0.915 and a Dice score of 0.956. In the mid-retinal strata, performance metrics decrease slightly: the inner-nuclear layer (INL) and outer-plexiform layer (OPL) record IoUs of 0.798 and 0.829 and Dice scores of 0.889 and 0.907, respectively. Within the photoreceptor support tissue, the outer-nuclear layer (ONL) segmentation performs strongly with an IoU of 0.913 and a Dice of 0.954. Despite their sub-voxel axial thickness, the photoreceptor inner segments (IS) and outer segments (OS) are also successfully segmented, with IoUs of 0.794 and 0.800 and Dice scores of 0.885 and 0.889. At the outer blood-retina barrier, the retinal pigment epithelium (RPE) and vascular choroid are the final tissues segmented, with IoUs of 0.836 and 0.949 and Dice scores of 0.911 and 0.974. These per-layer results yield an overall mean IoU (mIoU) of 0.87 ± 0.06 and a mDice of 0.93 ± 0.04. Macro-averaged precision and recall are 0.932 and 0.927, respectively. Across all nine retinal boundaries, the average RMSE is 0.010px (SD 0.003px); the smallest RMSE occurs at the IS boundary (0.0066px) and the largest at the OPL boundary (0.0149px). All values lie well below the scanner's 3.9µm axial voxel size, indicating axial localisation accuracy comparable to manual grading.

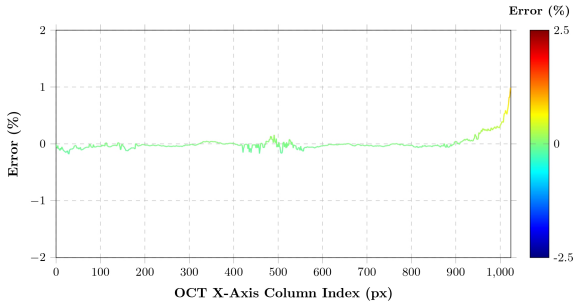

Fig. 9. Signal error for the John Hopkins dataset

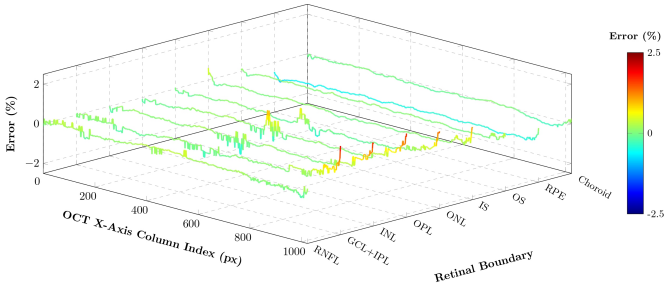

Fig. 10. Per-boundary signal error for the Johns Hopkins dataset, which aggregates to form the overall signal error shown in Figure 9.

Using the column index as the lateral coordinate (0 = temporal, 1024 = nasal), the signal-error curve for the John Hopkins dataset plotted in Figure 9 is almost flat across the dataset, suggesting accurate segmentation, with a global mean of $-0.02\% \pm 0.22\%$, this negative value due to the RPE layer as the boundary delineation seems to be under-fit cross the nasal-to-temporal extent as shown in Figure 10. From Figure 9, it is observed that the temporal region (column indices 0-199) shows a mild under-fitting of $-0.05\% \pm 0.05\%$ (extremes -0.18% to $+0.05\%$); the largest dip (-0.18%) appears near column index 29. The macular region (200–799) is essentially unbiased at $-0.02\% \pm 0.03\%$, never straying beyond $\pm 0.06\%$. The nasal periphery (800–1024) shows mild over-fitting at

$+0.08\% \pm 0.15\%$, rising to a single-column peak of $+0.99\%$ at column index 1024, adjacent to the optic nerve head. Because 98.7% of columns remain inside $\pm 1\%$, the model's performance across the clinically relevant macular region is stable and reliable for ophthalmic diagnosis, such as in the context of multiple sclerosis (MS). Only the far nasal 2% of the scan warrants caution, as biases here could obscure true pathological changes. The final U-Net++ model was benchmarked against the structured-surface network of He et al. [18], 2021, as both models were trained on the same Johns Hopkins dataset and evaluated with an identical 50/50 subject split; this direct methodological match makes their results the most appropriate baseline. The U-Net++ yields a mean boundary RMSE of 5.60µm (1.12% of the 499.2µm axial depth), only 0.40 percentage points (pp) higher than He et al.'s 3.60µm (0.72%), and still within roughly 1.4 axial voxels. Crucially, it provides markedly superior volumetric accuracy with a mIoU of 0.87 and a macro Dice of 0.93 versus the 0.80 and 0.91 reported by He et al., making it the more robust choice for clinical OCT analysis in ophthalmic practice.

B. External Validation

The final model was used to segment all the OCT scans in the NR206 dataset for external validation, producing the results tabulated in Table II to assess the model's generalisation and robustness. A sample output from the model can is displayed in Appendix-D in Figure 14. Segmentation of the

TABLE II
FINAL MODEL PERFORMANCE METRICS ON THE NR206 DATASET

Metric	Vitreous	RNFL	GCL+IPL	INL	OPL	ONL	IS	OS	RPE	Choroid	Average
Dice	0.9827	0.9260	0.9435	0.8710	0.8892	0.9375	0.8607	0.8728	0.9090	0.9699	0.9162
IoU	0.9660	0.8621	0.8931	0.7715	0.8006	0.8824	0.7554	0.7744	0.8332	0.9416	0.8480
Precision	0.9695	0.9450	0.9495	0.8716	0.9060	0.9338	0.8619	0.9000	0.8961	0.9692	0.9203
Recall	0.9963	0.9077	0.9376	0.8704	0.8731	0.9412	0.8594	0.8473	0.9223	0.9706	0.9126
RMSE	N/A	0.0107	0.0192	0.0228	0.0340	0.0258	0.0179	0.0164	0.0140	0.0112	0.0191

background/vitreous remains strong, with an IoU of 0.966 and a Dice score of 0.983. The RNFL and the GCL + IPL maintain high performance, with IoUs of 0.862 and 0.893 and Dice scores of 0.926 and 0.944, respectively. Segmentation accuracy decreases slightly in the mid-retinal layers: the INL and OPL achieve IoUs of 0.772 and 0.801 and Dice scores of 0.871 and 0.889, respectively. The ONL retains strong performance with an IoU of 0.882 and a Dice score of 0.938. The photoreceptor IS, and OS show solid segmentation despite their thin structure, with IoUs of 0.755 and 0.774 and Dice scores of 0.861 and 0.873, respectively. The RPE and choroid maintain high fidelity in the outermost layers, reaching IoUs of 0.833 and 0.942 and Dice scores of 0.909 and 0.970, respectively. These results aggregate to a mIoU of 0.85 ± 0.07 and a mDice of 0.92 ± 0.05. Macro-averaged precision and recall are 0.932 and 0.927, respectively, and overall pixel accuracy is 93.9%. Retinal boundary segmentation remains precise, with an average RMSE of 0.019 px (SD 0.007 px). The highest RMSE is at the OPL boundary, measuring just 0.034px, well within the scanner's 4µm axial sampling resolution.

The signal-error curve for the NR206 dataset plotted in Figure 11 is positive overall, indicating a tendency toward over-fitting hence over-segmentation across the dataset, with a global mean of $+0.25\% \pm 0.34\%$. In the temporal region, the

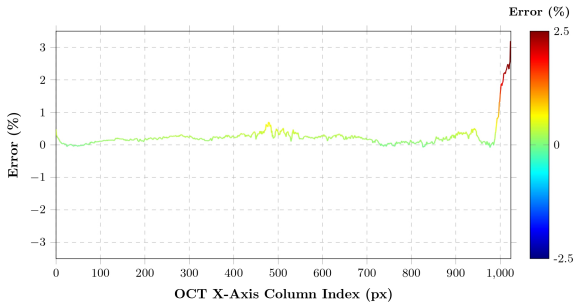

Fig. 11. Signal Error for the NR206 dataset

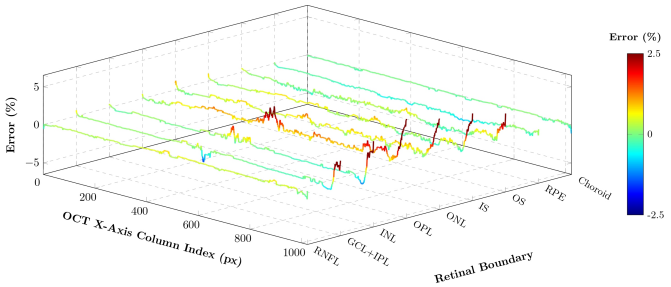

Fig. 12. Per-boundary signal error for the NR206 dataset, which aggregates to form the overall signal error shown in Figure 11.

model shows mild over-fitting ($+0.10\% \pm 0.08\%$), peaking at $+0.48\%$ in the first column index before gradually approaching zero. Across the macular region, the bias remains modest at $+0.23\% \pm 0.11\%$, with values ranging from -0.04% to a maximum of $+0.70\%$ at column index 697, due to the over-fitting of the ONL layer in this region as shown in Figure 12. In the nasal region, the error curve increases more sharply to $+0.42\% \pm 0.66\%$, culminating in a terminal peak of $+3.19\%$ at the far nasal edge appears to be caused by all the retinal boundaries, as shown in Figure 12, which seems highly unlikely given the model's boundary delineations across the nasal-to-temporal extent. Thus, likely reflecting a reduced signal-to-noise ratio near the optic nerve head due to imaging limitations. Having said that, 95% of macular-region columns in Figure 11 remain within $\pm 0.30\%$, ensuring that central-field thickness maps and normative reference values are preserved and unbiased in this external dataset. These findings affirm the model's robustness and anatomy-aware segmentation performance, demonstrating generalisability and reliability across diverse imaging devices, clinical centres, and subject populations. The final U-Net++ model was benchmarked on the NR206 dataset against the ConvNeXt-based segmentation network proposed by He et al.[12], 2023. The final model achieves a mDice of 0.9162, mIoU of 0.8480, and an overall pixel accuracy of 93.88%. In comparison, the ConvNeXt-based network attains a mDice of 0.913, mIoU of 0.8440, and a markedly higher pixel accuracy of 98.80%. Thus, the U-Net++ model rivals or exceeds the performance of the state-of-the-art ConvNeXt-based approach in overlap-based metrics (+0.32 pp mDice; +0.40 pp mIoU) while conceding 4.9 pp in pixel-level accuracy, highlighting complementary advantages: U-Net++ provides more anatomically consistent boundary segmentation, whereas ConvNeXt exhibits superior raw pixel accuracy.

VI. Conclusion

This project's objective was to develop a robust and generalisable deep-learning framework for automated retinal OCT segmentation and boundary delineation by systematically evaluating U-Net, U-Net++, and U-Net 3+ architectures in combination with various loss functions. Through extensive architecture–loss experimentation, U-Net++ paired with a combined Cross-Entropy and Dice loss (Combo loss) emerged as the most scalable and balanced model. After retraining on an expanded Johns Hopkins dataset, the final U-Net++ model was developed, evaluated, and benchmarked. On the internally held-out Johns Hopkins test set, it achieved a mean Dice score of 0.929 and a mean IoU of 0.870, outperforming the structured-surface network of He et al., 2021, which reported a mean Dice of 0.910 and mean IoU of 0.800. Furthermore, external validation on the independent NR206 dataset demonstrated the model's generalisability, with a mean Dice of 0.9162 and mean IoU of 0.8480, again surpassing the ConvNeXt-based network of He et al., 2023, which achieved a mean Dice of 0.913 and mean IoU of 0.844. Additionally, boundary delineation RMSEs remained well below the scanner's inherent axial resolution (model RMSE: 2.1µm vs scanner resolution: 3–5µm), highlighting the model's anatomical precision. Thus, this project underscores the clinical viability of deep-learning-based OCT segmentation of healthy and pathological retinas across diverse imaging systems and patient populations, with the potential for further extension in future work. The signal error plots developed in this project provide a versatile metric for evaluating future deep learning–based retinal OCT segmentation. In addition to offering intuitive visualisations of model performance, these plots can be integrated directly into training and inference workflows—as auxiliary outputs to improve interpretability, as secondary loss objectives during retraining, or as real-time quality-control indicators when deploying models in clinical settings. Furthermore, the final U-Net++ model offers a strong foundational framework for future work and clinical deployment in automated retinal segmentation and disease detection. Its proven anatomical accuracy and cross-dataset generalisability in both healthy individuals and patients with pathologies make it an ideal foundation for adaptation and extension to pathology-specific tasks, such as quantifying retinal layer thinning—a key biomarker for glaucoma, age-related macular degeneration, and other degenerative diseases. By integrating disease-specific training objectives and post-processing, the current framework can be adapted to detect subtle structural changes, support retinal health monitoring, and fit seamlessly into clinical imaging workflows. These targeted refinements will enhance research and translational efforts, paving the way for real-world, AI-driven diagnostic support in ophthalmology.

Acknowledgment

I would like to sincerely thank Dr. Raheleh Kafieh for her invaluable supervision and guidance throughout this project. Her support, feedback, and expertise were instrumental in shaping its direction and success. I would also to extend my heartfelt thanks to Mrs Sagrie Naidoo for sponsoring the paid Google Colab resources, which were crucial for developing, training and testing this project's final working machine-learning model.

References

[1] M. Zeppieri, S. Marsili, E. S. Enaholo, A. O. Shuaibu, N. Uwagboe, C. Salati, L. Spadea, and M. Musa, "Optical coherence tomography (oct): A brief look at the uses and technological evolution of ophthalmology," *Medicina (Kaunas)*, vol. 59, no. 12, p. 2114, Dec 2023.

[2] R. Kapoor, S. P. Walters, and L. A. Al-Aswad, "The current state of artificial intelligence in ophthalmology," *Survey of Ophthalmology*, vol. 64, pp. 233–240, Mar 2019.

[3] S. Aumann, S. Donner, J. Fischer *et al.*, "Optical coherence tomography (oct): Principle and technical realization," in *High Resolution Imaging in Microscopy and Ophthalmology: New Frontiers in Biomedical Optics*, J. F. Bille, Ed. Cham (CH): Springer, 2019, ch. 3.

[4] W. Drexler and J. G. Fujimoto, "State-of-the-art retinal optical coherence tomography," *Progress in Retinal and Eye Research*, vol. 27, no. 1, pp. 45–88, 2008.

[5] F. Cabitza, R. Rasoini, and G. F. Gensini, "Unintended consequences of machine learning in medicine," *JAMA*, vol. 318, p. 517, Aug. 2017.

[6] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*. Springer, 2015, pp. 234–241.

[7] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4*. Springer, 2018, pp. 3–11.

[8] H. Huang, L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, X. Han, Y.-W. Chen, and J. Wu, "Unet 3+: A full-scale connected unet for medical image segmentation," in *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 1055–1059.

[9] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," *arXiv preprint arXiv:1312.6034*, 2013.

[10] B. Zhou, A. Khosla, A. Lapedriza, A. Oliva, and A. Torralba, "Learning deep features for discriminative localization," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2921–2929.

[11] Y. He, A. Carass, S. D. Solomon, S. Saidha, P. A. Calabresi, and J. L. Prince, "Retinal layer parcellation of optical coherence tomography images: Data resource for multiple sclerosis and healthy controls," *Data in Brief*, vol. 22, pp. 601–604, 2019.

[12] X. He, Y. Wang, F. Poiesi, W. Song, Q. Xu, Z. Feng, and Y. Wan, "Exploiting multi-granularity visual features for retinal layer segmentation in human eyes," *Frontiers in Bioengineering and Biotechnology*, vol. Volume 11 - 2023, 2023.

[13] I. Loshchilov and F. Hutter, "Sgdr: Stochastic gradient descent with warm restarts," in *Proceedings of the International Conference on Learning Representations (ICLR)*, 2017.

[14] Z. Wang, E. Wang, and Y. Zhu, "Image segmentation evaluation: a survey of methods," *Artificial Intelligence Review*, vol. 53, no. 8, pp. 5637–5674, 2020.

[15] D. M. Kline and V. L. Berardi, "Revisiting squared-error and cross-entropy functions for training neural network classifiers," *Neural Computing & Applications*, vol. 14, pp. 310–318, 2005.

[16] F. Milletari, N. Navab, and S. A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *Proceedings of the 2016 Fourth International Conference on 3D Vision (3DV)*, 2016, pp. 565–571.

[17] A. Galdran, G. Carneiro, and M. A. G. Ballester, "On the optimal combination of cross-entropy and soft dice losses for lesion segmentation with out-of-distribution robustness," 2022.

[18] Y. He, A. Carass, Y. Liu, B. M. Jedynak, S. D. Solomon, S. Saidha, P. A. Calabresi, and J. L. Prince, "Structured layer surface segmentation for retina oct using fully convolutional regression networks," *Medical Image Analysis*, vol. 68, p. 101856, 2021.

VII. Appendix

A. Code and Dataset Availability

The code developed for this research project is free to use and adapt; if used, please cite this paper. The code can be accessed here: https://github.com/s-naidoo/Automated-Retinal-OCT-Segmentation-And-Boundary-Delineation.git

The whole John Hopkins dataset is publicly available for download at: http://iacl.jhu.edu/Resources

The NR206 dataset is publicly available at: https://github.com/Medical-Image-Analysis/Retinal-layer-segmentation

B. Overview of Datasets

The Johns Hopkins OCT Dataset consists of high-resolution retinal scans from 35 subjects, including 14 healthy controls and 21 individuals diagnosed with multiple sclerosis. Acquired using the Spectralis OCT system (Heidelberg Engineering), each right-eye scan covers a 6 mm $\times$ 6 mm macular region and includes 49 B-scans per volume, with each B-scan composed of 1,024 A-scans and 496 axial pixels. The scans offer an axial resolution of approximately 3.9 µm, a lateral resolution of 5.8 µm, and a through-plane resolution of 123.6 µm. Manual segmentations are provided for eight retinal layers: RNFL, GCL+IPL, INL, OPL, ONL, IS, OS, and RPE. The dataset supports retinal layer segmentation, thickness analysis, and machine learning model training tasks. Raw volumes are available in `.vol` format with corresponding manual annotations in `.mat` files.

The NR206 OCT Dataset is a curated set of B-scan images from 206 healthy eyes acquired using the Cirrus HD-OCT system (Carl Zeiss Meditec) at the Sankara Nethralaya Eye Hospital. Each scan focuses on the fovea with a 2 mm raster scan and a resolution of 500 $\times$ 750 pixels, featuring an axial resolution of 5 µm and a transverse resolution of 15 µm. The dataset includes semantic segmentation annotations for eight retinal layers (NFL, GCL+IPL, INL, OPL, ONL, ELM+IS, OS, RPE) along with a background and a combined layer, resulting in ten total classes. Annotations were manually created and verified by ophthalmology professionals. Designed for semantic segmentation tasks, the dataset is ready-to-use in PNG format without additional preprocessing.

C. Final Model Hyperparameters

TABLE III
Final Model Hyperparameters

Hyper-parameter	Value
Architecture	U-Net++ (in_channels=1, num_classes=10, base_filters=64)
Loss	Combo ($\alpha = 0.5$)
Optimizer	Adam
Initial learning rate	1×10^{-3}
Weight decay	1×10^{-5}
LR scheduler	CosineAnnealingLR ($T_{\max} = 45$, $\eta_{\min} = 1 \times 10^{-5}$)
Batch size	8
Number of epochs	45
Combo-loss weighting (α)	0.5

D. Final Model Output Samples

Figure 13 and Figure 14 are sample visualisation outputs for the code where (A) shows the OCT scan input, (B) displays the table of retinal layers and their associated colours, (C) is the ground truth OCT segmentation mask, (D) is the ground truth boundary mask, (E) shows the model's predicted segmentation output, (F) shows the model's predicted boundary mask, and (G) presents the probability error heatmap in each of the figures.

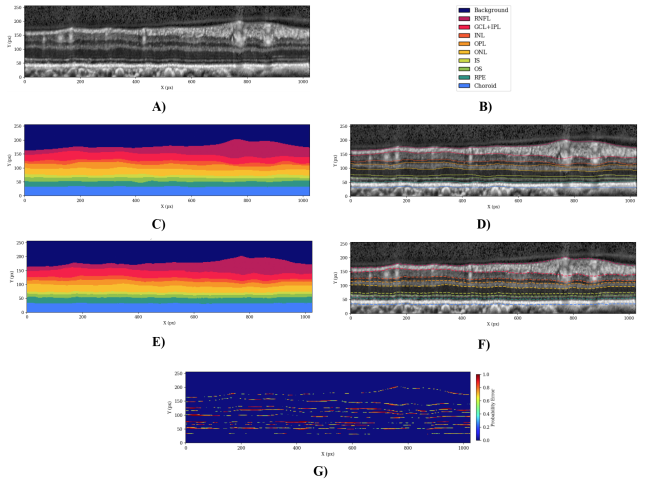

Fig. 13. Final model sample visualisation output from the John Hopkins dataset.

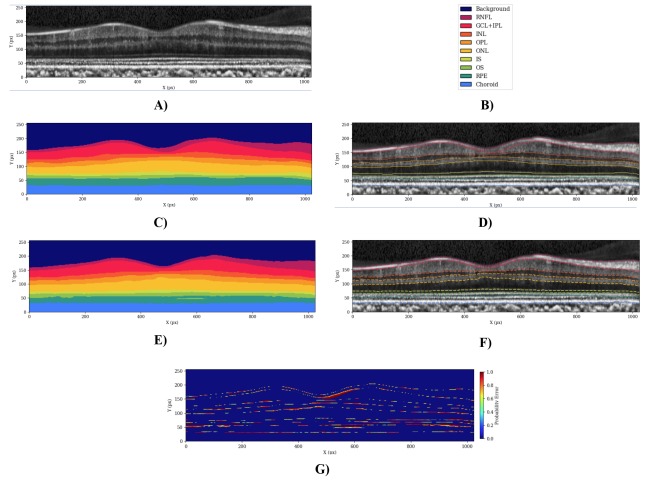

Fig. 14. Final model sample visualisation output from the NR206 dataset.

E. Acknowledgment of Generative AI Use

I acknowledge using ChatGPT to troubleshoot code, optimise performance to avoid unnecessary paid resource consumption, and generate detailed comments for the final model before uploading it to GitHub, ensuring clarity and usability for a broad audience.