DEEP LEARNING BASED ADAPTIVE RESIZING OF HIGH RESOLUTION IMAGES FOR IMPROVED SEGMENTATION PERFORMANCE

A Thesis Submitted to
the Graduate School of Engineering and Sciences of
İzmir Institute of Technology
in Partial Fulfillment of the Requirements for the Degree of

MASTER OF SCIENCE

in Electrical and Electronics Engineering

by
Ekrem FİDAN

June 2025
İZMİR

We approve the thesis of **Ekrem FİDAN**

Examining Committee Members:

Assoc. Prof. Dr. Abdurrahman Gümüş
Electrical and Electronics Engineering, İzmir Institute of Technology

---

Assist. Prof. Dr. M. Zübeyir Ünlü
Electrical and Electronics Engineering, İzmir Institute of Technology

---

Assist. Prof. Dr. Başak Esin Köktürk Güzel
Electrical and Electronics Engineering, İzmir Democracy University

20 June 2025

Assoc. Prof. Dr. Abdurrahman Gümüş
Supervisor's Department
Electrical and Electronics Engineering
İzmir Institute of Technology

Prof. Dr. Barış ATAKAN
Head of the Department of
Electrical and Electronics Engineering

Prof. Dr. Mehtap EANES
Dean of the Graduate School of
Engineering and Sciences

ACKNOWLEDGMENTS

I would like to express my sincere gratitude to the individuals who have supported me throughout my master's journey and contributed to the completion of this thesis.

My deepest gratitude goes to my advisor, Assoc. Prof. Dr. Abdurrahman Gümüş. This research would not have been possible without his helpful advice and persistent assistance. I am particularly grateful for his patience and for fostering an environment of intellectual curiosity that was fundamental to my development as a researcher.

I am also grateful for my time with the MIRALAB Research Group. The friendly and collaborative atmosphere was a constant source of motivation. I am especially thankful to my fellow research assistants for the engaging discussions and shared challenges that made this journey both productive and enjoyable.

I also want to thank the İzmir Institute of Technology for giving me the academic tools and setting I needed to do my research.

Finally, and most importantly, I could not have completed this journey without the unwavering love and support of my family. To my parents, thank you for your endless encouragement and for always believing in me. Your support has been my greatest motivation, and this accomplishment is as much yours as it is mine.

ABSTRACT

DEEP LEARNING BASED ADAPTIVE RESIZING OF HIGH RESOLUTION IMAGES FOR IMPROVED SEGMENTATION PERFORMANCE

Downsampling high-resolution medical images for deep learning models often compromises diagnostic accuracy due to information loss with traditional resizing methods. This thesis explores and advances adaptive resizing techniques to enhance medical image analysis.

Initial work affirmed the superiority of an existing adaptive resizer over bilinear interpolation for colorectal gland segmentation on the CRAG dataset, improving Intersection over Union (IoU) by up to 8.2%. Building on this, the primary contribution is the development and rigorous evaluation of six novel adaptive resizer architectures. These were designed to optimize both segmentation/classification performance and computational efficiency. The proposed resizers were tested on retinal vessel segmentation using the High-Resolution Fundus (HRF) dataset and diabetic retinopathy classification with the Indian Diabetic Retinopathy Image Dataset (IDRiD).

Experimental results show the proposed architectures generally surpass existing methods. For segmentation, 'Resizer MFY' achieved the highest average IoU increase of +21.04% over bilinear interpolation. In classification, 'Resizer A2' proved most effective, with an average F1-score increase of +22.39% over bilinear. Critically, the 'Minimal V1' architecture consistently demonstrated the lowest computational overhead among the novel adaptive resizers. It offers significant performance improvements while being considerably more lightweight than other adaptive methods, including the original adaptive resizer.

This research successfully demonstrates that these new adaptive resizers can significantly improve deep learning model accuracy in medical imaging. The work provides tailored, computationally considerate solutions, highlighting the importance of the resizing strategy in the analysis pipeline and paving the way for more effective diagnostic tools.

ÖZET

GELİŞMİŞ SEGMENTASYON PERFORMANSI İÇİN YÜKSEK ÇÖZÜNÜRLÜKLÜ GÖRÜNTÜLERİN DERİN ÖĞRENME TABANLI UYARLANABİLİR YENİDEN BOYUTLANDIRILMASI

Yüksek çözünürlüklü tıbbi görüntülerin derin öğrenme modelleri için küçültülmesi, geleneksel yeniden boyutlandırma yöntemleriyle bilgi kaybı nedeniyle tanısal doğruluğu sıkça tehlikeye atmaktadır. Bu tez, tıbbi görüntü analizini geliştirmek için uyarlanabilir yeniden boyutlandırma tekniklerini araştırmakta ve ilerletmektedir.

Başlangıç çalışmaları, CRAG veri seti üzerinde kolorektal bezi segmentasyonu için mevcut bir uyarlanabilir yeniden boyutlandırıcının bilineer interpolasyona göre üstünlüğünü doğrulamış, Kesişim üzeri Birleşim (IoU) oranını %8.2'ye kadar artırmıştır. Bu bulgular üzerine inşa edilen temel katkı, altı yeni uyarlanabilir yeniden boyutlandırıcı mimarisinin geliştirilmesi ve titiz bir şekilde değerlendirilmesidir. Bunlar, hem segmentasyon/sınıflandırma performansını hem de hesaplama verimliliğini optimize etmek için tasarlanmıştır. Önerilen yeniden boyutlandırıcılar, Yüksek Çözünürlüklü Fundus (HRF) veri seti kullanılarak retina damar segmentasyonunda ve Hint Diyabetik Retinopati Görüntü Veri Seti (IDRiD) ile diyabetik retinopati sınıflandırmasında test edilmiştir.

Deneysel sonuçlar, önerilen mimarilerin genellikle mevcut yöntemlerden daha iyi performans gösterdiğini ortaya koymaktadır. Segmentasyon için, 'Resizer MFY' bilineer interpolasyona göre ortalama IoU artışında +%.21.04 ile en yüksek performansı elde etmiştir. Sınıflandırmada, 'Resizer A2' bilineere göre ortalama F1 skorunda +%.22.39 artışla en etkili olduğunu kanıtlamıştır. Kritik olarak, 'Minimal V1' mimarisi, yeni uyarlanabilir yeniden boyutlandırıcılar arasında sürekli olarak en düşük hesaplama yükünü göstermiştir. Orijinal uyarlanabilir yeniden boyutlandırıcı da dahil olmak üzere diğer uyarlanabilir yöntemlere göre önemli ölçüde daha hafifken, dikkate değer performans iyileştirmeleri sunmaktadır.

Bu araştırma, bu yeni uyarlanabilir yeniden boyutlandırıcıların tıbbi görüntülemede derin öğrenme modeli doğruluğunu önemli ölçüde artırabildiğini başarılı bir şekilde göstermektedir. Çalışma, özel olarak tasarlanmış, hesaplama açısından dikkate değer çözümler sunarak, analiz ardışık düzeninde yeniden boyutlandırma stratejisinin önemini vurgulamakta ve daha etkili tanı araçlarının önünü açmaktadır.

To the person I was three years ago, beginning with hope, yet carrying self-doubt every step of the way.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

CHAPTER 1

INTRODUCTION

The proliferation of high-resolution imaging across various domains, particularly in medical diagnostics, has provided unprecedented detail for analysis. However, the direct application of these high-resolution images in deep learning frameworks often encounters practical limitations. Deep learning models, especially convolutional neural networks (CNNs), are typically designed with specific input dimensionality constraints. These constraints arise from architectural choices, such as fixed-size fully connected layers, and the necessity to manage computational resources effectively. Consequently, models often expect input images of standardized sizes, for instance, 224×224, 256×256 pixels, or dimensions that are powers of two, to facilitate efficient processing through network layers like pooling layers. This necessitates the downsampling of high-resolution source images. The challenge of effectively bridging the gap between high-resolution data acquisition and the input requirements of deep learning models, while minimizing information loss, has been a significant point of investigation in various studies (Bakhtiarnia et al., 2024; Jin et al., 2022).

Traditional image downsampling techniques, such as bilinear or bicubic interpolation, are widely used due to their simplicity and computational efficiency. However, these numerical methods operate based on fixed mathematical rules, often leading to a significant loss of critical fine-grained features and high-frequency details. This information loss can be particularly detrimental in applications where subtle details are paramount for accurate interpretation, such as in medical image analysis for disease diagnosis or precise object segmentation. The degradation of image quality during downsampling can therefore limit the performance ceiling of sophisticated deep learning models. This problem is well-documented, with research showing how standard downsampling methods can discard critical spatial information necessary for tasks like semantic segmentation (Xu et al., 2023; Hesse et al., 2023) or obscure subtle yet vital indicators in medical image analysis (Khan et al., 2023; Yang et al., 2023).

Recognizing these limitations, the research community has actively explored advanced image downsampling algorithms. The goal has been to develop techniques that can reduce image resolution while better preserving task-relevant information. Within this evolving landscape, adaptive image downsampling has emerged as a promising research

area. Unlike fixed algorithms, adaptive methods aim to tailor the downsampling process by considering the content of the image or the specific requirements of the subsequent computer vision task. Such adaptive strategies are diverse; some works, for example, focus on learning to sample image regions non-uniformly based on content importance for segmentation (Jin et al., 2022), while others develop content-adaptive downsampling schemes directly within network architectures (Hesse et al., 2023). In different contexts, such as processing 3D scan data or detailed medical slides, adaptive approaches also aim to preserve critical geometric or semantic features that would be lost with uniform reduction (Qiu et al., 2022; Yang et al., 2023). Other parallel studies have focused on designing novel pooling layers that are inherently more adaptive and information-preserving (Stergiou and Poppe, 2023).

A key innovation in adaptive downsampling is the concept of a learnable resizer, notably introduced by Talebi & Milanfar in 2021. This approach treats the image resizer not as a static preprocessing step but as an integral part of the deep learning pipeline (Talebi and Milanfar, 2021). Typically, an adaptive resizer is a compact neural network module positioned at the beginning of the main deep learning architecture. A crucial aspect of this paradigm is the joint training of the resizer module and the main deep learning model (e.g., for classification or segmentation). Through this end-to-end training, the resizer learns to downsample images in a way that is specifically optimized for the task at hand and the subsequent network's "preferences." The objective is to transform the input images into a lower-resolution representation that maximally retains features salient to the model for improved understanding and performance, rather than just applying a generic reduction algorithm. This philosophy of making the resizing process itself a learnable and task-aware component is a departure from earlier paradigms and has inspired further research into how input transformations can be optimized. For example, related studies have also explored learnable modules for downsampling in the context of ultra-high-resolution image segmentation, demonstrating the benefits of task-specific adaptation (Jin et al., 2022), and the concept of learnable resizers is often discussed in contrast to or in conjunction with other adaptive techniques (Hesse et al., 2023).

This thesis delves into the domain of adaptive image resizing, aiming to contribute to its advancement and practical application in demanding image analysis tasks. The work is structured around two primary studies. The first study (Chapter 2) investigates the efficacy of an established adaptive resizer architecture for semantic segmentation, a task where information preservation is critical. Specifically, it evaluates the Original Adaptive Resizer (based on Talebi & Milanfar's work) against bilinear resizing for colorectal gland

segmentation using the CRAG dataset, demonstrating the potential of adaptive resizing to enhance segmentation accuracy in medical histopathology.

Building upon the insights from the initial investigation, the second, more extensive study (Chapter 3) focuses on addressing some of the limitations of existing adaptive resizers by proposing a novel suite of adaptive resizer architectures. These new architectures are designed with the dual goals of further improving performance in both image classification and segmentation tasks while also considering and optimizing computational efficiency. The development of these novel resizers involved extensive experimentation with various architectural components, including a wide array of vision attention mechanisms such as Channel attention, spatial attention, self-attention-based modules, and combinations thereof (e.g., SENet, ECANet, SKNet, NonLocal, CBAM, BAM, FcaNet, SA-Net, DA-Net, EMA-Net), to enhance feature discriminability (Guo et al., 2022). This part of the research involves rigorous comparative analyses of the finally proposed architectures against traditional methods and the established adaptive resizer using diverse, high-resolution medical imaging datasets—the High-Resolution Fundus (HRF) dataset for retinal vessel segmentation and the Indian Diabetic Retinopathy Image Dataset (IDRiD) for diabetic retinopathy grading.

Through these investigations, this thesis aims to provide a comprehensive analysis of novel adaptive resizing strategies, highlighting their potential to significantly enhance the accuracy and efficiency of deep learning models in medical image analysis. The subsequent chapter is a conclusion that summarizes and presents these studies, culminating in an overall conclusion of the findings and their implications for the field.

CHAPTER 2

IMPROVED COLORECTAL GLAND SEGMENTATION IN HISTOPATHOLOGY IMAGES WITH ADAPTIVE RESIZER-ENHANCED U-NET MODELS

2.1. Study Abstract

Utilizing low-resolution images for computer vision tasks such as classification and segmentation can sometimes hinder the model's ability to accurately learn essential features. While using high-resolution images and designing compatible models might seem like viable solutions, they are not always feasible due to energy efficiency and graphical computation constraints. Downsizing images for model training and application is an effective approach for improving computational efficiency and optimizing model performance. The bilinear resizing method, commonly employed for this purpose, inherently causes information loss due to its numerical approach, which relies solely on the four nearest pixel values to compute each target pixel. This limitation becomes more pronounced with high-resolution images, where the down sampling process intensifies the loss of critical information. However, recent advancements have introduced adaptive resizer modules, which dynamically adjust image dimensions to better preserve essential features before processing by deep learning models. In biomedical image analysis tasks, such as segmentation, where high accuracy is paramount, the consequences of information loss are more detrimental compared to tasks prioritizing processing speed. Therefore, adaptive resizing presents a promising solution to mitigate information loss during the down sampling process. In this study, we propose an adaptive resizer-based segmentation framework for the gland segmentation task which is crucial for accurate disease diagnosis and treatment planning, particularly in cancer analysis. Three distinct encoder-decoder architecture segmentation models are assessed for image segmentation using the CRAG Gland Segmentation database. Each architecture was tested separately, employing six different backbone encoders that were pre-trained on the ImageNet dataset. Our comparative analysis revealed that the adaptive resizer improved the IoU metric by

approximately 5.6%. The lowest IoU using bilinear resizing was 62%, rising to 70% with the adaptive resizer. The highest IoU reached was 78%, an 6% improvement over the baseline of 72%. Each experiment shows an increase in performance that underscores the adaptive resizer's potential to improve gland segmentation methodologies. The codes will be shared on GitHub as an open-source software package in a way that benefits everyone interested.

2.2. Introduction

Segmentation is a foundational process in computer vision that divides an image into segments or classes based on predefined criteria. In the domain of computer vision, AI models are equipped to handle a variety of segmentation tasks. Among these tasks are semantic segmentation, where every pixel in an image is assigned a specific class (Minaee et al., 2021); instance segmentation, which can be viewed as an advanced form of semantic segmentation that differentiates between objects of the same class within an image (Gu et al., 2022); and panoptic segmentation. The latter combines the concepts of semantic and instance segmentation, proving especially useful in complex environments (Elharrouss et al., 2021). Panoptic segmentation identifies both individual objects and background elements that, although not distinct objects themselves, form an essential part of the overall environment. Semantic segmentation finds its applications in a range of fields, from medical imaging to the technology behind autonomous vehicles and even in the analysis of satellite imagery (Minaee et al., 2021; Elharrouss et al., 2021).

Glands are organs or groups of cells in an organism that synthesize substances, including hormones and enzymes. The precise detection and segmentation of these glands are of utmost importance, especially for the early detection of diseases like cancer (Rastogi et al., 2022). Characteristics such as the shape, size, and arrangement of glands can provide valuable insights into underlying pathologies. Utilizing deep learning models for the task of gland segmentation not only makes the process more efficient but also ensures optimal use of available resources (Rastogi et al., 2022; Wang et al., 2022). However, achieving accurate gland segmentation is not without its challenges, primarily due to the diverse appearance of glandular tissue in images. In such detailed images, it is crucial to retain the resolution to ensure that even the most minute features are not overlooked.

To enhance accuracy rates in segmentation tasks, extensive research has been conducted on models employing various architectures, including convolutional, encoder-decoder, transformer structures, and other methodologies (Mo et al., 2022; Lateef and

Ruichek, 2019). A notable contribution in this domain is the U-Net model, introduced by Ronneberger et al. in 2015, which has proven to be highly effective for biomedical image segmentation (Ronneberger et al., 2015). The impressive accuracy rates achieved by U-Net spurred further research, leading to the development of several U-Net-based models. These include U-Net++ (Zhou et al., 2018), Attention U-Net (Oktay et al., 2018), Trans U-Net (Chen et al., 2021), U2-Net (Qin et al., 2020), and U-Net 3+ (Huang et al., 2020), each boasting either enhanced accuracy rates or more efficient parameter configurations. Beyond the U-Net derivatives, other models such as DeepLabv3 (Chen et al., 2017), DeepLabv3+ (Chen et al., 2018), SegNet (Badrinarayanan et al., 2017), MILDNet (Graham et al., 2019), and CMD-Net (Zhang et al., 2020) have also demonstrated commendable performance in segmentation tasks. However, a common limitation across these models is their need for input images of lower resolution than the maximum attainable photographic quality. This constraint necessitates the use of down-scaled images, which inherently restricts the potential of the segmentation tasks.

In the realm of deep learning, a prevalent challenge arises when high-resolution images are input into models designed for low-resolution images. Traditional resizing methods often result in the loss of data in an image, especially the loss of high-frequency features. This is particularly concerning in critical sectors like medicine, where image segmentation and classification accuracy are paramount. Despite advancements, the accuracy rate for such tasks plateaus, highlighting the need for improved image processing techniques.

Recognizing the inherent necessity of image resizing in image processing tasks, the search for a more advanced method becomes crucial, especially as the scope for improvements through traditional mathematical techniques narrows. In this context, Talebi and Milanfar innovatively proposed an adaptive resizing approach, a significant deviation from conventional methods, which dynamically adjusts the resizing process based on the specific requirements of the task (Talebi and Milanfar, 2021). Their proposed framework comprises a compact model with approximately 12,000 parameters, strategically positioned at the onset of the deep learning model and concurrently trained. This approach was evaluated using both a learnable resizer and a binary resizer across four different image classification models, revealing a notable performance enhancement attributed to the learnable resizer.

Following the introduction of the learnable resizer by Talebi and Milanfar, the model has been adopted across various datasets and integrated into diverse models. Additionally, the success of this model has spurred the development of new adaptive resizer

models. Hao Li, in his research, explored the impact of the learnable resizer for few-shot learning. The module that is proposed in the study is the modified version of the original learnable resizer, and the name of the model is MAR (Model adaptive resizer). Instead of using residual blocks, this module uses MAR blocks, which contain channel attention mechanisms in these blocks (Li et al., 2023). Li Zou and his colleagues undertook a study employing the learnable resizer model to classify signals that were converted into time-frequency images. Their primary objective was to discern whether the signal from a rotating component was indicative of a malfunctioning machine or one that was operating correctly (Zou et al., 2023). Rahman et al. conducted research to enhance the performance of segmentation and classification models by integrating the learnable resizer model. (Rahman, 2023). In another notable study, Duzyel et al. demonstrated the effectiveness of an adaptive resizer to improve the performance of deep learning models for the diagnosis of breast cancer using histopathology images. Their findings highlighted the adaptive resizer as a powerful tool for enhancing image classification. By preserving important details and adapting to the unique characteristics of images, the adaptive resizer achieved better performance across all magnification factors, particularly excelling at 40× magnification, where it significantly outperformed bilinear interpolation (Duzyel et al., 2023). Han and Chen enhanced their classification task for COVID-19 CT scan images by incorporating a learnable resizer model. They integrated this resizer with the MobileNet, creating a hybrid model that was benchmarked against various other CNN architectures. The accuracy of the new model (96.9%) is higher than the original MobileNet (92.6%) and also other CNN architectures like VGG19, ResNet50V2, and Inception-v3 (Han and Chen, 2021). Zhang et al. designed a new resizer. In contrast to traditional approaches, the study by Zhang et al. introduces a streamlined learnable resizer that employs convolution to project features into a higher-dimensional space. This is further refined using a self-attention mechanism. The authors claim that their method not only simplifies the process but also outperforms existing techniques, particularly when tested on the Pittsburgh30K dataset. However, they acknowledge that their results are constrained by available computational resources and time (Zhang et al., 2022).

Bilinear resizing, a commonly used approach, often leads to significant information loss due to its reliance on only the four nearest pixel values. This issue is especially pronounced in high-resolution images and pixel-wise tasks like segmentation, where preserving spatial information is critical. The adaptive resizer represents a significant advancement in adaptive downsampling. However, to the best of our knowledge, existing applications of this technique have predominantly focused on classification tasks, with

no prior studies exploring its potential for segmentation tasks. Recognizing this research gap, our study investigated the potential of the adaptive resizer for segmentation tasks, addressing the unique challenges these tasks pose. Unlike classification, which outputs a single label or probability, segmentation requires pixel-level predictions that precisely align with the input dimensions, demanding greater preservation of spatial information and structural details throughout the downsampling process. This makes the integration of the adaptive resizer to segmentation both innovative and critical for advancing the field.

Our research focuses on the critical area of tissue segmentation, a cornerstone in medical imaging analysis, employing the CRAG colorectal gland segmentation dataset for in-depth analysis. This dataset, characterized by its diverse glandular structures, provides a robust platform for evaluating the effectiveness of segmentation models in preserving spatial information. Our core objective is to highlight the influence of a novel deep learning-based adaptive resizer model, crafted to downsize images, on the efficacy of segmentation. We integrated the adaptive resizer into three U-Net-based architectures: U-Net, U-Net 3+, and Attention U-Net, to evaluate its ability to preserve spatial information and improve segmentation accuracy compared to bilinear resizing. Additionally, we analyzed the trade-offs between accuracy and computational efficiency to provide a comprehensive assessment of the resizer's performance. Furthermore, we provided a thorough visual exploration of our findings, including original images, ground truths, and outputs obtained using both bilinear and adaptive resizing, along with corresponding model predictions. Heatmaps illustrating the adaptive resizer's focus on critical regions demonstrated its ability to retain essential features during downsampling. These results underscore the adaptive resizer's potential not only for segmentation but also for related tasks like object detection and instance segmentation, where detail preservation is crucial. By addressing a gap in the existing literature, this study not only establishes the feasibility of using the adaptive resizer in segmentation but also lays a foundation for future research into optimizing adaptive preprocessing methods for deep learning pipelines.

The ensuing sections of this chapter are structured as follows: The Methods section covers the hardware and software frameworks, the chosen dataset, the segmentation models, and the CNN models that support them. This section concludes with an in-depth overview of the adaptive resizer, its implementation, and the metrics used for performance evaluation. Following this, the Results and Discussion section details the study's methodology, sets the comparison criteria, and discusses the obtained results and accompanying visuals. The chapter ends with a Conclusion section.

2.3. Methods

This section details the comprehensive methodology employed in the initial study. It begins by outlining the computational environment, including the specific hardware and software frameworks utilized. Subsequently, it describes the CRAG dataset and the necessary preprocessing steps, followed by an in-depth explanation of the U-Net-based segmentation architectures and their convolutional backbones. The section further elaborates on the adaptive resizer model, the integrated framework for segmentation, the performance evaluation metrics, and the specific design of the experiments conducted.

2.3.1. Computational Setup

Our comparative experiments, encompassing segmentation models, backbones, and the evaluation of the adaptive resizer against the conventional resizer, were conducted in an environment utilizing the TensorFlow and Keras libraries. The NumPy and matplotlib libraries were employed for image processing and plotting tasks. A pre-existing library accessible online was leveraged, rather than building the models from the ground up. Specifically, an assortment of U-Net architectures complete with their corresponding codes was utilized from a GitHub repository named "keras unet-collection" (Sha, 2021). Furthermore, the learnable resizer model architecture code was pre-developed and accessible for immediate deployment from the Keras.io webpage (Paul, 2021).

The hardware infrastructure for our experiments was powered by an Nvidia A100 GPU, boasting 40 GB of memory, provided by Google Colab Pro Plus. In the experiments, any data augmentation techniques were refrained from being employed. Given the limited number of images in the dataset, the k-fold cross-validation method was adopted for all training sessions to enhance the generalizability of the validation score. With k set at 5, models were trained on a total of 213 images, each with five distinct validation sets.

2.3.2. Dataset and Preprocessing Steps

Our study employed the CRAG dataset, as introduced by Graham et al. (Graham et al., 2019). We also recognize the pioneering work of Awan et al., where this dataset was first utilized (Awan et al., 2017). The CRAG dataset, designed for colorectal gland segmentation in colon histological images, was released by the University Hospitals

Coventry and Warwickshire (UHCW) NHS Trust, located in Coventry, United Kingdom. It comprises 213 H&E images, partitioned into training and validation subsets. The training set encompasses 173 images, each accompanied by its annotation, while the validation set consists of 40 images, again with their respective annotations. Predominantly, the images in this dataset hover around the dimensions of 1500x1500. To ensure these images and their annotations are compatible as input for deep learning models, a preprocessing step was undertaken to standardize them into square images. This involved establishing a base dimension of 1504x1504, onto which the dataset images were superimposed. This method facilitated the acquisition of images with uniform height and width, all the while preserving their original aspect ratios.

2.3.3. Segmentation Architectures

In this study, three distinct segmentation models were explored: U-Net, U-Net 3+, and Attention U-Net. All three models are derivatives of the foundational U-Net architecture. The name "U-Net" is derived from its characteristic U-shaped architecture. Unlike classification tasks, segmentation requires not only the identification of a class but also its precise location within an image. U-Net and its variants excel in this regard.

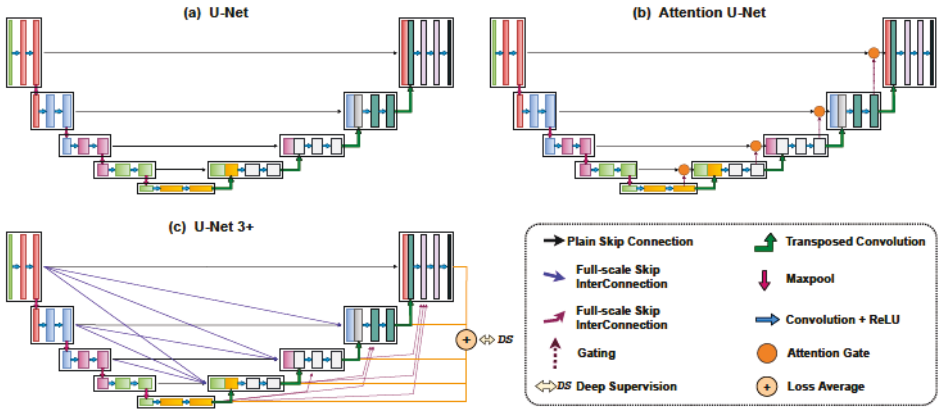

Figure 2.1. Segmentation models used in the study: (a) U-Net model; (b) Attention U-Net model; (c) U-Net 3+ model.

The architecture of U-Net, depicted in Figure 2.1(a) , was conceived by Ronneberger, Fischer, and Brox from the Computer Science Department and BIOSS Centre

for Biological Signaling Studies at the University of Freiburg, Germany (Ronneberger et al., 2015). The U-Net model comprises three primary modules: the contracting path, bottleneck path, and expansive path. The contracting path, often referred to as the "encoder," diminishes spatial information while extracting contextual information through convolutional and pooling layers. Conversely, the expansive path or "decoder" amplifies spatial dimensions. The integration of skip connections between the encoder and decoder aids in the restoration of spatial information, ensuring the final output retains both spatial and contextual details.

The architecture of Attention U-Net, illustrated in Figure 2.1(b), was developed by Oktay et al., in collaboration with institutions like Imperial College London and Nagoya University (Oktay et al., 2018). As a variant of the original U-Net, the Attention U-Net incorporates attention gates within its decoder. These gates determine the significance of the tensor from the standard expansive path in relation to the skipped tensor. By doing so, the model can concentrate on the most pertinent sections of an image for a specific task, enhancing segmentation precision.

The architecture for U-Net 3+, presented in Figure 2.1(c), was devised by Huang et al., with affiliations to institutions such as Zhejiang University, Sir Run Shaw Hospital, Ritsumeikan University, and Zhejiang Lab (Huang et al., 2020). U-Net 3+ is an evolved version of both U-Net and U-Net++. It incorporates full-scale skip connections and deep supervision into its design. The full-scale skip connections, in tandem with the standard skip connections, refine the accuracy of segmented object positions and boundaries. Additionally, each decoder block receives guidance from the ground truth, further enhancing the model's performance.

2.3.4. Convolutional Backbones of the Segmentation Models

The segmentation models employed in this study are all rooted in the U-Net framework, characterized by its encoder-decoder architecture. The encoder components of these models are often constructed using established CNN architectures renowned for their robustness and precision. By harnessing pre-trained weights from models such as VGG, ResNet, and DenseNet each trained on the "ImageNet" dataset and subsequently freezing these weights, feature extraction capabilities can be amplified even from a limited dataset, thereby enhancing the performance of the models.

VGG (Visual Geometry Group): This CNN architecture, a brainchild of the Visual Geometry Group at Oxford University, employs 3x3 convolutional layers paired with max

pooling layers. Among its variants, VGG16 and VGG19 stand out. VGG16, with its 16 weight layers, houses 138 million parameters, while VGG19, comprising 19 weight layers, contains 144 million parameters. While the VGG architecture's design is straightforward, its vast parameter count renders it computationally demanding (Simonyan and Zisserman, 2014).

DenseNet (Densely Connected Convolutional Networks): Developed by the Computer Vision Group at Cornell University, DenseNet is another CNN architecture. Its popular iterations include DenseNet121, DenseNet169, and DenseNet201. The trailing numbers indicate the layer count; for instance, DenseNet121 possesses 121 layers with roughly 8 million parameters, while DenseNet169 has 169 layers with an estimated 14 million parameters. Despite its relatively fewer parameters, which lends it an edge over similar architectures, the computational intricacy can pose challenges, especially for profoundly deep networks (Huang et al., 2017).

ResNet (Residual Network): Introduced by He, Zhang, Ren, and Sun from Microsoft Research, ResNet is a CNN architecture that incorporates skip connections and shortcuts, allowing certain layers to be bypassed. These residual connections empower the model to learn residual features, effectively addressing the vanishing gradient dilemma. Notable versions of this architecture encompass ResNet-34, ResNet-50, ResNet-101, and ResNet-152. To illustrate, ResNet50 boasts approximately 23 million parameters, ResNet 101 houses around 43 million parameters, and ResNet-152 has close to 58 million parameters. A distinct advantage of ResNet lies in its capacity to train exceptionally deep models, with the residual connections ensuring gradient preservation (He et al., 2016).

2.3.5. Adaptive Resizer

For deep learning models to be effective, they must be both robust and computationally efficient during training. To achieve this, many models utilize low-resolution images as input. Models trained on such low-resolution images tend to perform optimally when provided with input images of a similar size, creating an inevitable cycle of reliance on low-resolution imagery. Several algorithms have been developed to reduce image size, including Nearest-Neighbor interpolation, bilinear interpolation, bicubic interpolation, Lanczos Resampling, Area-based (Pixel-Averaging) resampling, antialiased resampling, and the Gaussian pyramid method. While these algorithms are effective in general scenarios, their efficacy diminishes in tasks demanding higher precision and accuracy, such as segmentation. Utilizing these conventional resizing algorithms often

results in information loss.

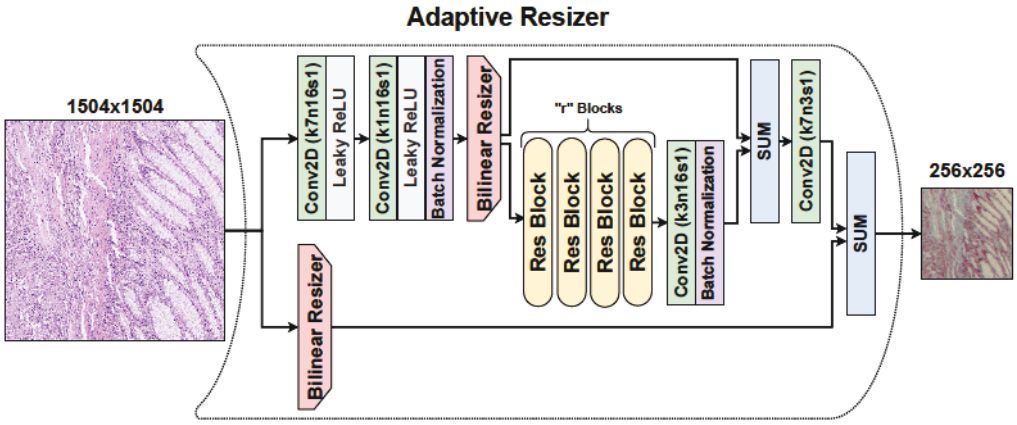

Figure 2.2. Architecture of the adaptive resizer model, a specialized neural network designed for image downscaling in computer vision tasks. The model resizes images from 1504x1504 to 256x256 using a combination of bilinear resizing and deep learning techniques. The output from the bilinear resizer and the convolutional pathways are summed to generate the final resized image, optimizing input for subsequent AI models.

The adaptive resizer, a novel approach in image resizing, was developed by Talebi et al. from Google Research (Talebi and Milanfar, 2021). Unlike traditional resizing algorithms, the adaptive resizer, illustrated in Figure 2.2, is not merely an algorithm but a neural network designed to work in synergy with the primary computer vision model. The adaptive resizer architecture efficiently processes high-resolution images by combining bilinear resizing with convolutional neural network (CNN) feature extraction. Initially, images are processed through two parallel pathways: one directly applies bilinear interpolation to resize images to target dimensions, while the other pathway enhances features through convolutional layers. This second pathway starts with a 7x7 kernel convolution followed by Leaky ReLU activation, and then a 1x1 kernel convolution with Leaky ReLU and batch normalization. After these initial convolutions, the features are resized to the target dimensions using another bilinear interpolation, creating a bottleneck that consolidates the extracted features. These features can be further refined through a series of optional residual blocks, each adding depth and complexity to the feature extraction. After passing through a final convolution with a 3x3 kernel and batch normalization, these processed features are merged with the output from the direct bilinear resizing pathway via element-wise addition. This method effectively combines detailed

feature enhancements with preserved spatial accuracy of the original image, resulting in a high-quality resized image that is ideal for applications requiring precise image fidelity. The adaptive resizer identifies crucial pixels or pixel locations for a designated task through training on a specific set of images. In essence, it produces images that are fine-tuned for models that have been co-trained with this resizer, ensuring optimized performance (Talebi and Milanfar, 2021).

2.3.6. Adaptive Resizer Based Segmentation Framework

The adaptive resizer module is strategically positioned at the beginning of the segmentation model, effectively merging the two into a unified deep learning model. In the context of this study, the segmentation model is designed to accept 256x256, 3-channel RGB images as input and subsequently produce a binary black and white mask as its output. The adaptive resizer's role is to transform the larger 1504x1504 images into the required 256x256 format. Consequently, the integrated model, a fusion of the adaptive resizer and the segmentation model, processes 1504x1504, 3-channel images and outputs 256x256 predicted binary masks.

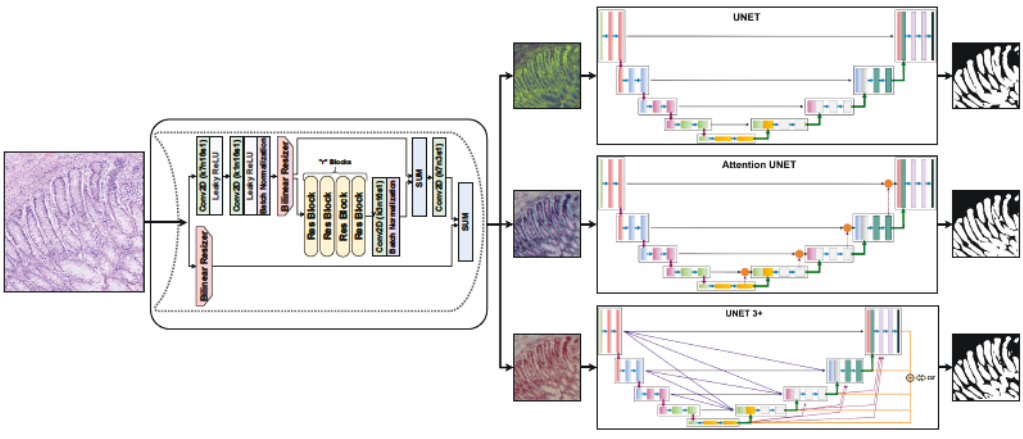

Figure 2.3. Overview of the proposed adaptive resizer based segmentation framework. Adaptive resizer is jointly trained with the specific models for the gland segmentation task. For each of the segmentation architectures, separate adaptive resizer models were trained.

Figure 2.3 illustrates the process flow within our system: initially, a larger image

is fed into the adaptive resizer model, which then outputs a lower-dimensional image. This resized image is subsequently processed by the segmentation model to produce the desired segmentation result. Notably, this entire system, integrating both the adaptive resizer and the segmentation model, is trained and operated as a unified model. This integration allows for simultaneous training and application, effectively merging what were previously two separate models into a single, cohesive system.

2.3.7. Performance Evaluation Metrics

Semantic segmentation classifies each image pixel into distinct categories. To evaluate trained segmentation models, three metrics are used: Accuracy, Dice coefficient (DSC), and Intersection over Union (IoU). Accuracy measures the percentage of correctly classified pixels in the predicted image, with values ranging from 0 (no correct classifications) to 1 (perfect classification). The calculation is given in Equation 2.1. Dice Coefficient (DSC) assesses the similarity between the ground truth and the model's prediction. Its values can vary between 0 (no overlap) and 1 (complete overlap). The formula is in Equation 2.2. Intersection over Union (IoU), also known as the Jaccard Index, evaluates the overlap between predicted and annotated segments. Its values range from 0 (no overlap) to 1 (full overlap), as shown in Equation 2.3.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP} \tag{2.1}$$

$$\text{Dice} = \frac{2TP}{2TP + FP + FN} \tag{2.2}$$

$$\text{IoU} = \frac{TP}{TP + FP + FN} \tag{2.3}$$

2.3.8. Experimental Details

The primary objective of this study is to demonstrate that integrating the adaptive resizer model with any segmentation model enhances performance, yielding more refined results. Three different U-Net based segmentation models were employed: U-Net, U-Net 3+, and Attention U-Net. For the backbone architecture, seven options were available.

Six of these are established backbones pretrained on the ImageNet dataset, while the seventh represents a model without any specific backbone. The backbones in consideration are VGG16, VGG19, ResNet-50V2, ResNet-152V2, DenseNet121, DenseNet201, and No-backbone. By integrating the adaptive resizer with these U-Net based models, 21 additional models were generated. The impact of the adaptive resizer was assessed across these 42 models. Table 2.1 shows the total number of parameters of the 21 different combinations of models and backbones. When the adaptive resizer model is combined with these models, their parameters count increases by 12 thousand parameters only.

Table 2.1. Number of total parameters in the used segmentation models (in millions).

Model	VGG 16	VGG 19	ResNet 50V2	ResNet 152V2	DenseNet 121	DenseNet 201	No Backbone
U-Net	31.17 M	36.48 M	20.68 M	55.44 M	19.72 M	27.88 M	46.71 M
Attention U-Net	28.38 M	33.69 M	17.80 M	52.56 M	16.87 M	25.04 M	43.94 M
U-Net 3+	24.88 M	30.19 M	17.23 M	51.99 M	14.92 M	21.98 M	39.26 M

In the subsequent phase of the experiment, one of the 21 combinations, which exhibited relatively superior performance, was selected. Two distinct scenarios were then explored by making modifications to the adaptive resizer module of this selected model, without altering its core structure. Firstly, the impact of varying the number of "residual blocks" within the adaptive resizer module, ranging from 1 to 5, was investigated. Additionally, the effects of combining outputs from the bilinear resizer and the neural network branch in different proportions during the summation process in the adaptive resizer's final stage were examined.

For the experimental process, 213 images and their corresponding mask images from the dataset were combined into a single file. Given the dataset's limited image count, this amalgamation aimed to enhance the generalizability of the findings. To further enhance the robustness of the evaluations, a k-fold cross-validation strategy was adopted with k set to 5.

During the model training phase, a "save the best weights" checkpoint approach was utilized. Although the training spanned 100 epochs, the storage of the model was

based on the best result achieved within these epochs, as determined by the dice coefficient value. Since the task is semantic segmentation with only two classes, binary cross-entropy was selected as the loss function. The Adam optimizer, with a learning rate of 10^{-3}, was chosen for optimization.

2.4. Results and Discussions

In this section, the results and discussions are presented from our investigation into the integration of adaptive resizers with various U-Net-based segmentation models across different architectural backbones. This analysis, structured into four main parts, explores the enhancement of model performance through adaptive resizing versus traditional methods, demonstrates the adaptive resizer's unique output characteristics through visualizations, provides a comparative visual assessment of model outputs with and without the adaptive resizer, and examines the effects of hyperparameter adjustments within the resizer's architecture.

2.4.1. Adaptive Resizer Increases the Performance

The performance of the U-Net, Attention U-Net, and U-Net 3+ models across various backbones is detailed in Table 2.2, which elucidates the metrics of accuracy, dice coefficient, and intersection over union. For each model the table is separated and all the backbone choices along with adaptive resizer integration are shown. The highlighted rows in the table pinpoint the optimal model-backbone combinations. Among the trio of models, the U-Net 3+ emerges as the top performer. The average IoU of all backbones for U-Net 3+ is 74.31 while U-Net is 73.89 and Attention U-Net is 73.43. Delving deeper into the backbone analysis, DenseNet201 stands out as the most effective backbone. The highest IoU values for all 3 segmentation models observed when the encoder part of the architectures are replaced with the DenseNet201 backbone.

Table 2.2. Performance metrics of segmentation models with different backbone encoders and with or without the adaptive resizer. The metrics include Accuracy, Dice Similarity Coefficient (DSC), Intersection over Union (IoU), and the increase in IoU when using the adaptive resizer.

Model Name	Backbone	Adaptive Resizer	Accuracy	DSC	IoU	Increase of IoU
U-Net	VGG16	No	0.840	0.823	0.700	6.2%
		Yes	0.875	0.865	0.762	
	VGG19	No	0.833	0.815	0.689	6.8%
		Yes	0.871	0.861	0.757	
	ResNet-50V2	No	0.804	0.788	0.652	4.1%
		Yes	0.836	0.818	0.693	
	ResNet-152V2	No	0.784	0.764	0.620	8.2%
		Yes	0.837	0.824	0.702	
	DenseNet121	No	0.835	0.816	0.690	7.2%
		Yes	0.877	0.864	0.762	
	DenseNet201	No	0.843	0.828	0.708	5.8%
		Yes	0.879	0.867	0.766	
	No Backbone	No	0.820	0.792	0.660	7.0%
		Yes	0.869	0.843	0.730	
Attention U-Net	VGG16	No	0.828	0.813	0.686	6.5%
		Yes	0.870	0.857	0.751	
	VGG19	No	0.820	0.804	0.674	6.6%
		Yes	0.863	0.850	0.740	
	ResNet-50V2	No	0.801	0.788	0.651	5.1%
		Yes	0.835	0.824	0.702	
	ResNet-152V2	No	0.784	0.769	0.627	6.8%
		Yes	0.828	0.819	0.695	
	DenseNet121	No	0.835	0.821	0.698	5.6%
		Yes	0.869	0.859	0.754	
	DenseNet201	No	0.845	0.830	0.710	5.3%
		Yes	0.874	0.865	0.763	
	No Backbone	No	0.806	0.768	0.628	10.8%
		Yes	0.873	0.847	0.736	
U-Net 3+	VGG16	No	0.837	0.824	0.702	5.7%
		Yes	0.870	0.863	0.759	
	VGG19	No	0.846	0.831	0.713	1.7%
		Yes	0.853	0.843	0.730	
	ResNet-50V2	No	0.820	0.809	0.681	2.9%
		Yes	0.839	0.829	0.710	
	ResNet-152V2	No	0.805	0.793	0.659	5.1%
		Yes	0.842	0.829	0.710	
	DenseNet121	No	0.854	0.841	0.727	4.5%
		Yes	0.876	0.870	0.770	
	DenseNet201	No	0.847	0.835	0.719	6.1%
		Yes	0.883	0.876	0.780	
	No Backbone	No	0.866	0.849	0.739	0.5%
		Yes	0.872	0.852	0.744	

Figure 2.4 offers a visual comparison between the training and validation plots for three distinct models with 5-fold cross validation. One set of models uses the bilinear resizing method, while the other set integrates the adaptive resizer. This comparison specifically focuses on the DenseNet201 backbone, our chosen architecture for this part of the study. Each training takes 100 epochs, and by using the save the best feature, attempts are made to obtain the best model for up to 100 epochs. The U-Net model's accuracy improved from 84% to 88%, while both the Attention U-Net and U-Net 3+ models saw increases from 85% to 88%. As a result, an average increase of 3.3% in accuracy was observed.

Figure 2.5 showcases the training plots for the Attention U-Net model. This visualization contrasts the model's efficacy across three distinct backbones, both with and without the adaptive resizer's integration. Specifically, the model utilizing the VGG19 backbone experienced an increase in accuracy from 82% to 86%. Similarly, the model with the ResNet-152V2 backbone recorded an increase from 78% to 83%. Furthermore, the model employing the DenseNet201 backbone reported an increase from 85% to 88%. As a result, an average increase of 4% in accuracy was observed.

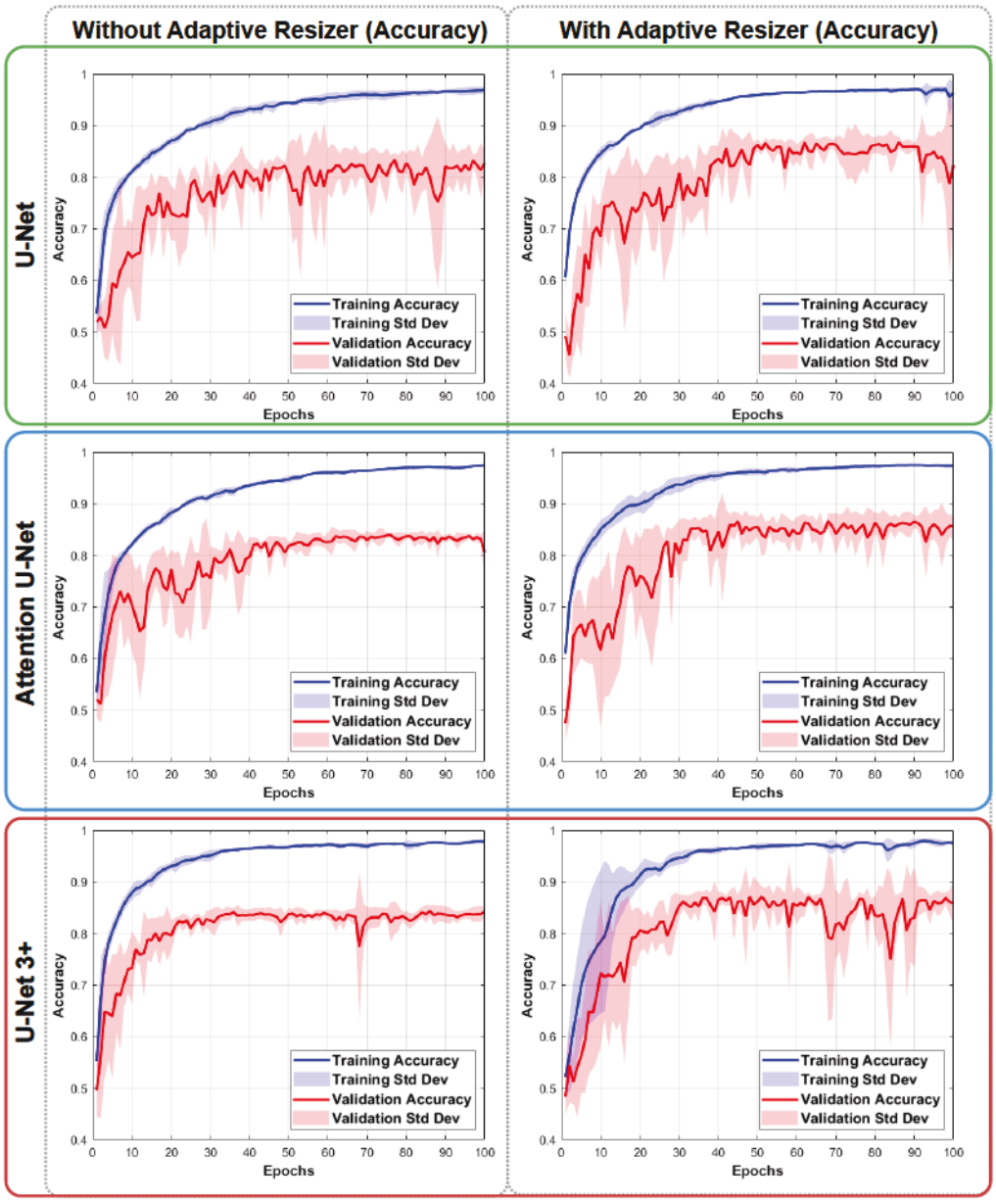

Figure 2.4. Training performance visualizations of the segmentation models with and without the adaptive resizer integration for the selected DenseNet201 backbone. Adaptive resizer combined models reach higher accuracy rates compared to the models using bilinear resizing.

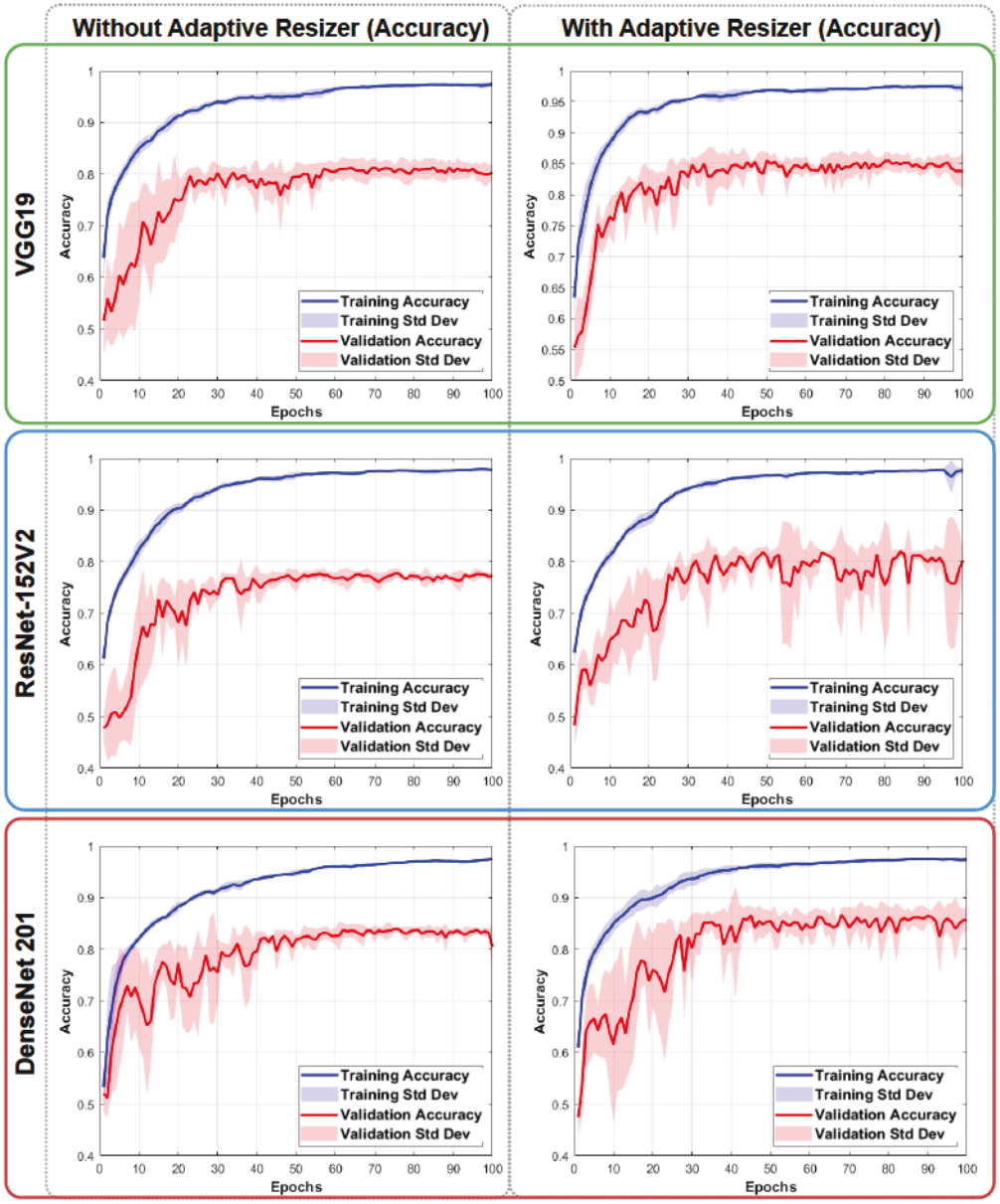

Figure 2.5. Training performance visualizations of the segmentation models' backbones with and without the adaptive resizer integration for the selected Attention U-Net model. Adaptive resizer combined models reach higher accuracy rates compared to models using bilinear resizing.

The adaptive resizer is a compact module that is trained jointly with the model that it is integrated with. It comprises only 12,000 parameters, which is considerably small compared to the millions of parameters in the primary models. Two sets of results were specifically compared: the best outcomes from the original models trained with images resized using the bilinear method and the results from the same models after the adaptive resizer was integrated. This comparison aimed to highlight the contributions of the adaptive resizer to these high-performing models. A comparative analysis of the results underscores the adaptive resizer's capability to enhance even the highest accuracy levels.

Even though the resized images may appear less visually appealing to human observers, with slight distortions, the adaptive resizer optimizes them specifically for the model's interpretation. This optimization enhances the extraction of meaningful features, allowing the segmentation model to achieve superior performance compared to using bilinear resizing, where crucial structural details might be lost.

Training time is a critical factor in the field of deep learning, where each second of computation often requires a high-powered GPU. Table 2.3 presents the training durations for the U-Net 3+ model, both with and without the inclusion of an adaptive resizer, across different backbone architectures. The times listed in the table represent the cumulative training durations for five models, each trained as part of a 5-fold cross-validation. Notably, incorporating the adaptive resizer into the U-Net 3+ model approximately doubles the overall training time compared to the original model. The significant disparity in training duration between the standalone model and its counterpart integrated with the adaptive resizer primarily stems from the processing of high-dimensional images. Despite the addition of a relatively small model, comprising merely 12,000 parameters, the crux of the issue lies in the size of the images utilized. Images measuring 1504x1504 pixels possess over 34 times the pixel count of those with dimensions of 256x256. Consequently, this vast increase in pixel density necessitates a proportional escalation in the time required for executing the convolution process. Should larger images be employed, the anticipated increase in training time would be expected to exceed a mere doubling, potentially requiring substantially more time due to the increased processing demands.

We conducted an in-depth analysis of the computational costs associated with integrating the adaptive resizer into the U-Net 3+ model, as shown in Table 2.3. The comparison focuses on key metrics such as FLOPs, inference time on two different hardware platforms (CPU: AMD EPYC 7B12 and GPU: NVIDIA A100-SXM4-40GB), and training duration. The inclusion of the adaptive resizer adds approximately 12,035 parameters

to each model, which is minimal compared to the total number of parameters ranging from about 14 million to 52 million in the U-Net 3+ architectures.

Table 2.3. Comparison for U-Net 3+ with and without adaptive resizer in terms of training duration, FLOPs, inference times, parameters.

Model Name	Parameters	FLOPs (Billion)	Inference Time (CPU)	Inference Time (GPU)	Training Duration
U-Net 3+ (VGG16)	24.88 M	369.27	1392.23 ms	154.32 ms	2215 s
AR + UNet 3+ (VGG16)	+12,035	382.16	1645.13 ms	197.68 ms	4437 s
U-Net 3+ (VGG19)	30.19 M	379.94	1335.48 ms	79.48 ms	2201 s
AR + U-Net 3+ (VGG19)	+12,035	393.03	1636.69 ms	88.95 ms	4401 s
U-Net 3+ (ResNet50V2)	17.23 M	470.85	1624.29 ms	139.42 ms	2493 s
AR + U-Net 3+ (ResNet50V2)	+12,035	483.94	1997.29 ms	145.11 ms	4741 s
U-Net 3+ (ResNet152V2)	51.99 M	490.26	2115.38 ms	308.11 ms	2824 s
AR + U-Net 3+ (ResNet152V2)	+12,035	503.35	2523.47 ms	312.66 ms	5007 s
U-Net 3+ (DenseNet121)	14.92 M	471.61	1719.77 ms	202.15 ms	2576 s
AR + U-Net 3+ (DenseNet121)	+12,035	484.70	2063.79 ms	214.39 ms	4892 s
U-Net 3+ (DenseNet201)	21.98 M	475.14	1902.15 ms	287.44 ms	2747 s
AR + U-Net 3+ (DenseNet201)	+12,035	488.23	2151.17 ms	294.22 ms	5142 s
U-Net 3+ (No Backbone)	39.26 M	377.80	1391.55 ms	98.97 ms	2664 s
AR + U-Net 3+ (No Backbone)	+12,035	390.65	1704.02 ms	107.88 ms	4875 s

However, we observed an increase in computational complexity when the adaptive resizer is incorporated. The FLOPs required for a forward pass increase across all models. For instance, in the U-Net 3+ with VGG16 backbone, the FLOPs rise from approximately 369 billion to 382 billion operations. This pattern is consistent across other backbones,

indicating additional computations introduced by the resizer.

Inference times also show an upward trend with the inclusion of the adaptive resizer. On the AMD EPYC 7B12 CPU, the inference time for the U-Net 3+ VGG16 model increases from approximately 1392 milliseconds to 1645 milliseconds. On the NVIDIA A100-SXM4-40GB GPU, the inference time increases from about 154 milliseconds to 198 milliseconds for the same model. This increase in inference time, although more pronounced on the CPU, highlights the additional processing required due to the resizer.

Training durations roughly double when the adaptive resizer is used. For example, training the U-Net 3+ VGG16 model extends from approximately 2215 seconds to 4437 seconds. This significant increase suggests that the resizer adds complexity to the training process, possibly due to the need for joint optimization of both the resizer and the segmentation network.

Despite the higher computational costs, the adaptive resizer contributes to improved segmentation accuracy by better preserving important features during downsampling. This enhancement is particularly valuable in medical imaging applications where precise segmentation is critical. The trade-off between computational cost and accuracy must be carefully considered based on the specific requirements and resource constraints of the application.

While the adaptive resizer introduces additional computational overhead in terms of FLOPs, inference time, and training duration, it offers the benefit of enhanced segmentation performance. The decision to incorporate the resizer should balance the need for higher accuracy against the available computational resources, especially in real-world scenarios where efficiency is crucial.

2.4.2. Visualization of Adaptive Resizer Outputs

Various methods exist to downscale an original image to desired dimensions. Broadly, these methods can be categorized into traditional and adaptive techniques. Traditional methods are algorithmically defined and executed by computers. Prominent among these are the bilinear, bicubic, Gaussian Pyramid, Lanczos Resampling, and Area-based (Pixel Averaging) techniques.

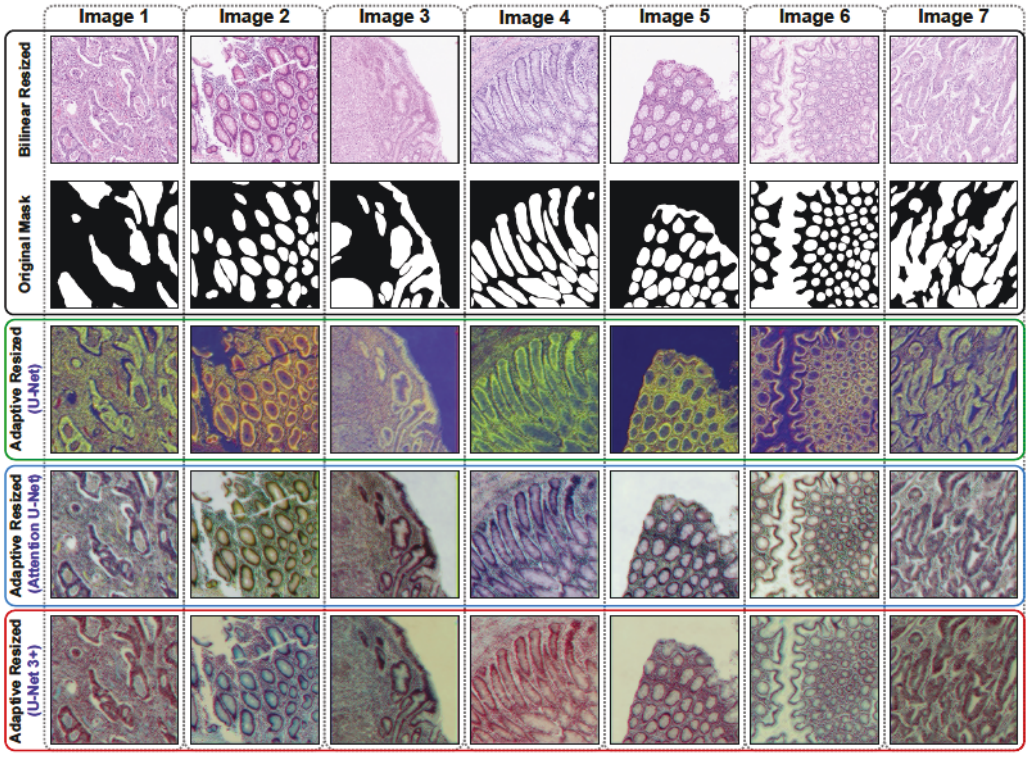

Figure 2.6. Adaptive resizer outputs trained jointly with segmentation models: U-Net, Attention U-Net, and U-Net 3+. Columns shows 7 images from the dataset. The first row shows the bilinear resized form of the orignal image and the corresponding masks. Other rows are the adaptive resizer outputs which are jointly trained with the segmentation models.

The adaptive resizer, shown in Figure 2.2, tailors the resizing process to each specific model. Not only does it reduce the dimensions of the original image, but it also blends RGB channels, leading to color alterations. This is attributed to the convolution operations with varying filter numbers, which are then reconverted to a 3-channel image. The outputs of the adaptive resizers, when trained in conjunction with a segmentation model, are displayed in Figure 2.6. During the extraction of the output from the adaptive resizer, the combined model is divided into two parts, isolating the adaptive resizer as an independent model. This isolated model, containing 12,000 parameters, generates outputs that appear as negative and positive fractional values. These values are then normalized to a 0-255 scale to render them interpretable. It is essential to recognize that the output from the adaptive resizer is not a conventional image. Due to the image undergoing multiple convolutions, increasing the feature channels from three (RGB) to sixteen and then reducing back to three, color mixing and changes are expected.

2.4.3. Visual Comparison: Effects of the Adaptive Resizer

Analyzing Figures 2.7 and 2.8, it becomes apparent that the adaptive resizer has a distinctive impact on the segmentation models' performance. Figure 2.7 provides a visual representation of annotations, bilinear resize output (BRO), BRO combined with the segmentation model output, adaptive resizer output (ARO), and ARO combined with the segmentation model output for all three models using the DenseNet201 backbone. Each model is represented in two rows, with each row depicting a distinct image from the dataset. The integration of the adaptive resizer enhances the segmentation model's ability to align its outputs closely with the ground truth. This improvement is particularly noticeable in the precise depiction of contours and structural details, emphasizing the resizer's contribution to maintaining spatial integrity during the resizing process.

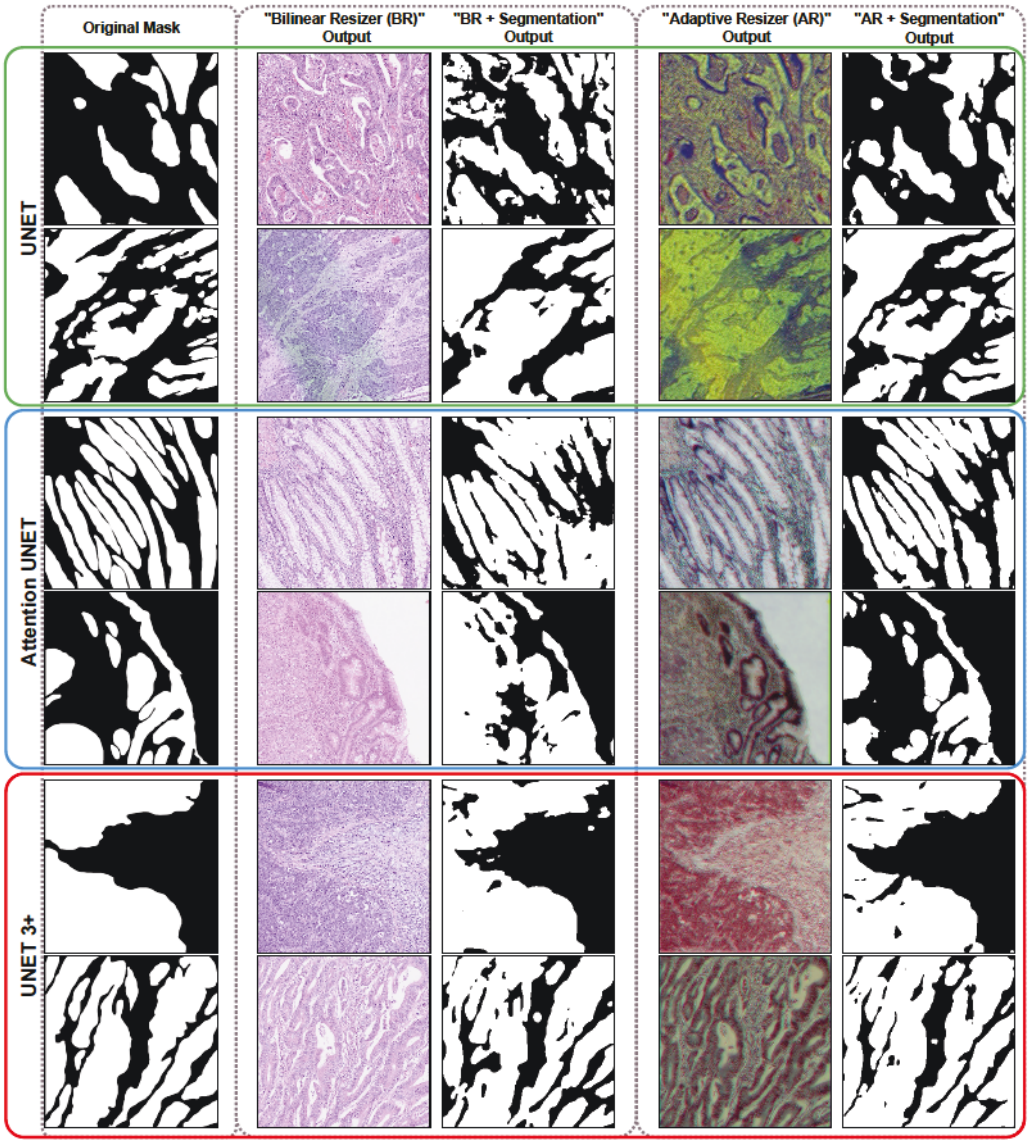

Figure 2.7. Comparison of the segmentation results to show input images seen by the model and their resulting segmentations. Outputs from two resizing methods, bilinear resizing and adaptive resizing, are used as inputs to three segmentation models: U-Net, Attention U-Net, and U-Net 3+. Columns show the original mask, bilinear resizer output, segmentation results with bilinear resizer, adaptive resizer output, and segmentation results with adaptive resizer. The adaptive resizer adjusts images based on model needs, which alters the general visual form of the image and results in more accurate segmentation.

Figure 2.8 presents a clear visual enhancement in model performances using the DenseNet201 backbone as it displays prediction outputs from models with and without the integration of the adaptive resizer. The first column shows four images from the validation subset of the dataset, while the second column presents the ground truth for the intended segmentation. Subsequent columns shows the predictions of the three models in their original and adaptive resizer combined versions, facilitating an easy comparison. A noticeable pattern emerges where models incorporating the adaptive resizer consistently present better-defined edges and fewer instances of over-segmentation. By incorporating selective feature scaling and preservation, the adaptive resizer minimizes the loss of significant image details during the resizing process. This tailored approach allows the segmentation model to leverage a higher-quality representation of the input, ultimately improving segmentation performance.

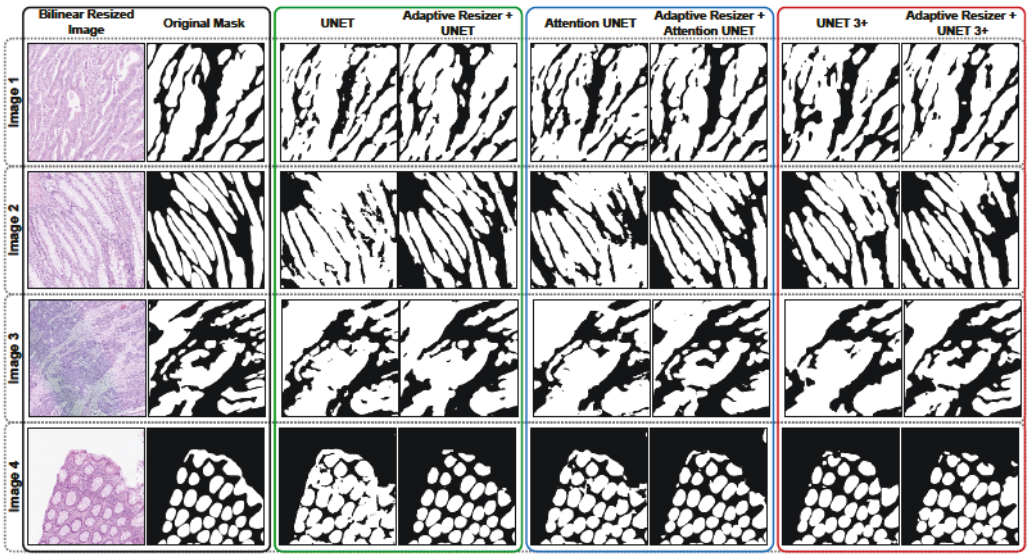

Figure 2.8. Comparison of segmentation performances using the adaptive resizer versus bilinear resizing across three models: U-Net, Attention U-Net, and U-Net 3+. Each row shows an original histology image and its corresponding ground truth mask, followed by segmentation predictions from the bilinear resizer used models and adaptive resizer-enhanced models. The results illustrate the improvement in segmentation accuracy when the adaptive resizer is incorporated into each model.

The segmentation performance of the U-Net 3+ model, with and without the integration of the adaptive resizer, is compared in Figure 2.9. The figure illustrates a series of images, organized by increasing segmentation complexity. Each column showcases

a different stage of this complexity, including the original histopathology image, the corresponding ground truth mask, predictions from the U-Net 3+ model without the adaptive resizer, and predictions from the U-Net 3+ model with the adaptive resizer integrated. The IoU values are displayed below each prediction, offering quantitative measures of the models' performance.

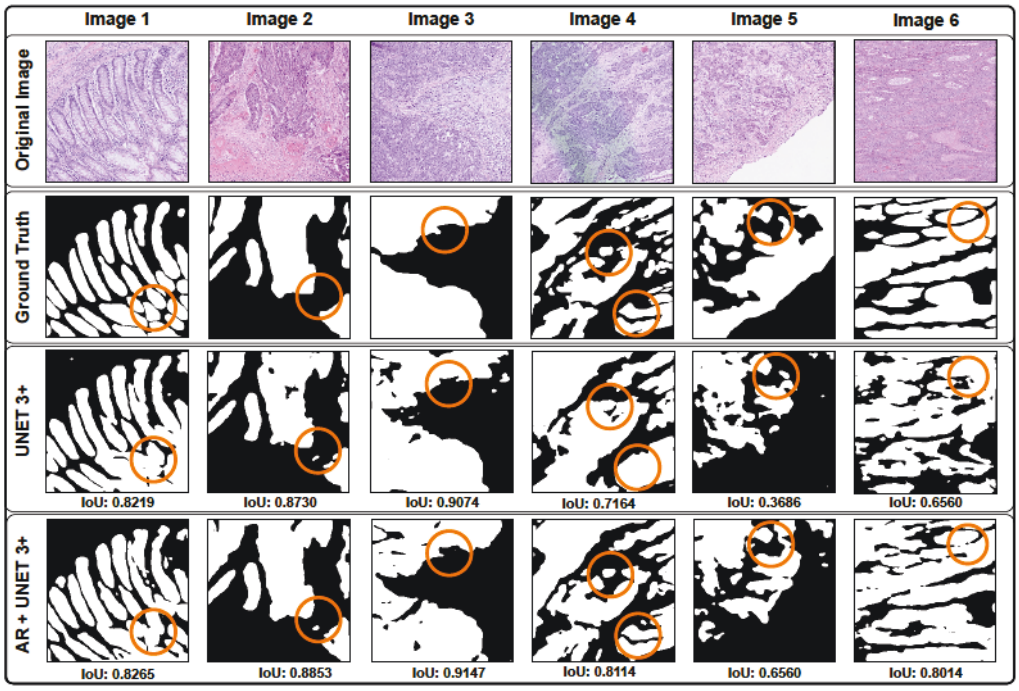

Figure 2.9. Comparative segmentation analysis of simple and complex histopathology images using UNet 3+ and AR-enhanced UNet 3+ with visual marks and IoU values.

Regions where the adaptive resizer improves segmentation are marked with orange circles, emphasizing areas where it enables the model to capture finer details or boundaries that the model without the resizer struggles to delineate accurately. The results demonstrate that the adaptive resizer enhances segmentation performance, especially as the complexity of the glandular structures increases.

For simpler glands, both models perform comparably, with marginal differences in IoU values. However, as complexity grows, the adaptive resizer consistently achieves higher IoU scores, reflecting its ability to handle challenging structures more effectively. This pattern underscores the adaptive resizer's advantage in scenarios where fine-grained segmentation is critical, although there may still be occasional instances where the re-

sizer's impact varies based on the specific image characteristics.

By providing both qualitative (visual inspection of circled regions) and quantitative (IoU scores) analyses, figure offers a comprehensive perspective on the impact of the adaptive resizer. The orange-circled regions and the higher IoU scores for complex gland structures highlight the resizer's ability to enhance segmentation precision in difficult cases, underscoring its potential value in applications where accurate delineation of complex structures is essential.

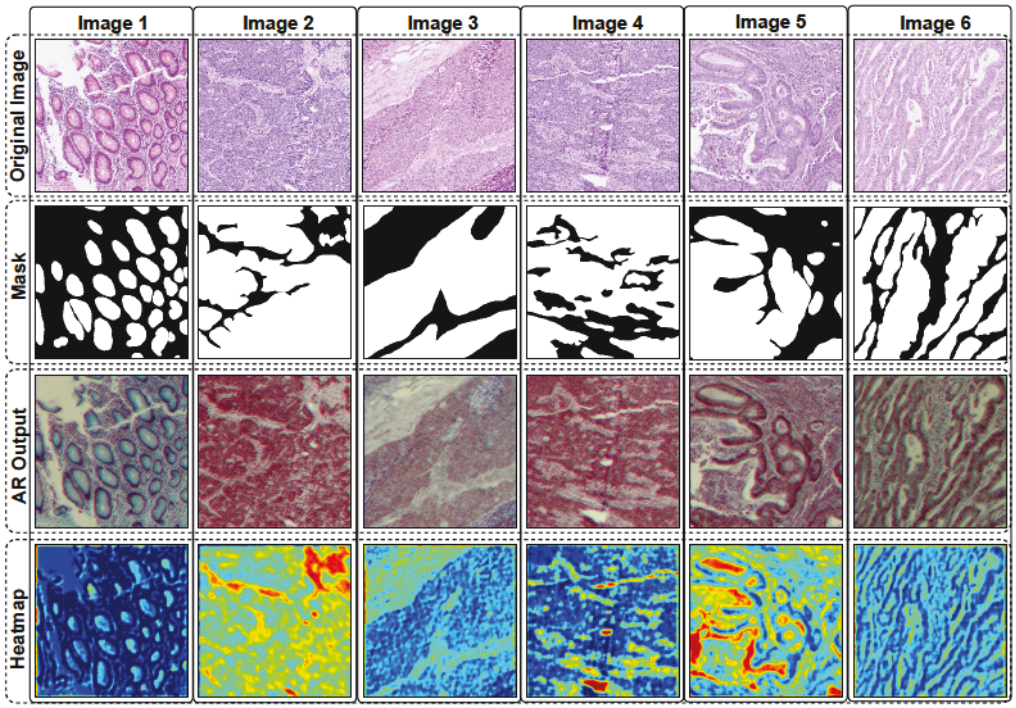

Figure 2.10. Comparative analysis of resized outputs and Grad-CAM heatmaps for segmentation enhancement. Grad-CAM overlays highlighting region-specific attention in adaptive resizer's downsampling process

To illustrate how the adaptive resizer enhances segmentation performance in specific image regions or structures, Figure 2.10 provides interpretable visualizations, including heatmaps. This figure presents the original histopathology images, their corresponding ground truth masks, the outputs generated by the adaptive resizer, and Grad-CAM (Gradient-weighted Class Activation Mapping) heatmaps, with Grad-CAM results derived from the last layer of the adaptive resizer.

In Figure 2.10, the first column displays the original images as input to the adaptive resizer. The second column shows the ground truth segmentation masks, highlighting

the glandular structures requiring precise identification. The third column presents the images processed by the adaptive resizer, demonstrating how the resizer transforms the input while aiming to preserve essential features. The fourth column includes Grad-CAM heatmaps, overlaid onto the original images, visualizing the regions where the adaptive resizer focuses its attention during downsampling.

By examining these heatmaps, it becomes evident that the adaptive resizer places significant emphasis on critical areas within the images, such as complex glandular structures and intricate boundaries. The highlighted regions in the Grad-CAM overlays indicate that the resizer assigns greater importance to challenging areas, ensuring that essential details are retained during resizing. This targeted approach contrasts with traditional resizing methods, which treat all regions uniformly and may lose important information necessary for accurate segmentation.

These interpretable visualizations provide valuable insights into the internal workings of the adaptive resizer. They demonstrate how the resizer selectively prioritizes specific structures within the images, contributing to improved segmentation performance. By highlighting the regions where the resizer allocates greater attention, readers can better understand the advantages of using the adaptive resizer over conventional resizing techniques. This analysis underscores the adaptive resizer's effectiveness in handling complex glandular structures and supports further research into adaptive preprocessing methods in medical image segmentation.

The adaptive resizer enhances the segmentation process by incorporating feature extraction directly within its resizing operations, bridging the gap between preprocessing and model-specific learning. This integration allows the segmentation task to be initiated earlier in the processing pipeline, which traditional resizing methods such as bilinear resizing do not aim to achieve. By maintaining critical spatial details during downsampling, the adaptive resizer better aligns the resized images with the U-Net models' feature extraction requirements, leading to improved segmentation outcomes.

The adaptive resizer is fundamentally a two-branch network. The first branch comprises convolutional layers followed by bilinear resizing layers, while the second branch performs only bilinear resizing. The outputs of these two branches are combined to produce the final result. In this setup, the first branch extracts features from the image before downsampling and processes the image after downsampling. This enables the segmentation task to begin before the U-Net models start their operation. In contrast, the bilinear downsampling method applies a straightforward numerical approach. It estimates each pixel value in the downsampled image by calculating a weighted average of the

surrounding pixels based on the downsampling ratio. For instance, when downsampling by a factor of 2, the algorithm identifies the four nearest pixels in the original image to obtain a weighted average of the original pixel values.

Comparing these two methods shows that the convolutional output in the adaptive resizer provides additional information about the segmented regions, enhancing the segmentation results. The Grad-CAM heatmaps in Figure 2.10 illustrate this difference: the adaptive resizer begins to understand the segmentation task and supports the U-Net models, while U-Net models with only bilinear resizing do not initiate the segmentation process as effectively.

The visual data suggests that the adaptive resizer allows for a segmentation that is more faithful to the intricate structures within the images. In images exhibiting more complex glandular structures, segmentation models enhanced by the adaptive resizer consistently achieve superior fidelity compared to those employing traditional resizing approaches. This performance gain highlights the resizer's capacity to preserve crucial image features throughout the resizing process, thereby laying a stronger foundation for subsequent segmentation tasks. Furthermore, comparative analyses reveal that the adaptive resizer's influence is not uniform across all samples but varies in response to the unique attributes of each image. While the resizer generally improves overall model performance, the degree of enhancement differs from one image to another, demonstrating its ability to dynamically adapt to diverse data characteristics. This variation underscores the potential need for further optimization to fully harness the adaptive resizer's capabilities for different types of medical imagery.

2.4.4. Analysis of Adaptive Resizer Overhead

We evaluated the computational costs of resizing high-resolution images from 1504×1504 pixels to 256×256 pixels, comparing traditional bilinear resizing with our proposed adaptive resizer module. The computational complexity was measured in terms of floating-point operations (FLOPs) required for each method. For the bilinear resizer, resizing from 1504×1504×3 to 256×256×3 consistently required approximately 2,819,084 operations. This cost remained virtually constant across different input sizes because bilinear interpolation relies on a fixed number of computations per output pixel, regardless of the input dimensions. In contrast, the adaptive resizer introduced significantly higher computational overhead. Resizing from 1504×1504 to 256×256 demanded 13,166,477,696 operations, which is several orders of magnitude greater than the bilinear method. This

substantial increase in FLOPs is due to the additional convolutional layers and residual blocks that process high-resolution inputs, causing the computational complexity to scale quadratically with the input size.

Table 2.4. Comparison of FLOPs for bilinear resizing and adaptive resizer module at different resolutions.

Resolution	Bilinear Resizing FLOPs	Adaptive Resizer FLOPs
2048x2048 to 256x256	2,819,084	23,368,958,336
1504x1504 to 256x256	2,819,084	13,166,477,696
1024x1024 to 256x256	2,819,084	6,759,514,496
512x512 to 256x256	2,819,084	2,607,153,536
256x256 to 256x256	2,819,084	1,569,063,296

We also analyzed other input dimensions to further illustrate this impact, as shown in Table 2.4. For example, resizing from 256×256 required 1,569,063,296 operations, while resizing from 512×512 demanded 2,607,153,536 operations. Similarly, resizing from 1024×1024 required 6,759,514,496 operations, and from 2048×2048, it required 23,368,958,336 operations. These results clearly demonstrate that the computational burden of the adaptive resizer increases significantly with larger input dimensions. Meanwhile, the bilinear resizer's FLOPs remained around 2,819,084 operations across all input sizes. This consistency emphasizes that the computational cost of bilinear resizing does not vary with input size, unlike the adaptive resizer, which scales quadratically.

This analytical comparison highlights a trade-off introduced by the adaptive resizer. While it significantly enhances segmentation accuracy by learning to retain critical features during resizing, it does so at the expense of increased computational complexity and overhead. The additional processing demands more resources and time, which can be limiting in real-time applications or environments with constrained computational resources. However, when comparing the computational cost of the adaptive resizer to the overall model complexity, the impact remains relatively small. The average FLOPs for the segmentation models without the adaptive resizer is 433 billion, while with the adaptive resizer, it increases to 446 billion, resulting in an average increase of only 13 billion FLOPs as hown in Table 2.3. This increase is minor compared to the total computational cost of the segmentation models themselves, which typically require hundreds of billions

of FLOPs. Therefore, despite its additional computational cost, the adaptive resizer remains a computationally feasible enhancement, particularly given the improvements in segmentation accuracy and feature preservation it provides.

The increased inference time due to the adaptive resizer also affects real-time performance, particularly in applications requiring high frame rates. Since FPS is inversely proportional to inference time, any increase in inference time results in a decrease in FPS. Given the average CPU inference time increase from 1,640.12 ms to 1,960.22 ms, this corresponds to an increase of approximately 19.5%. Similarly, the average GPU inference time increases from 181.41 ms to 194.41 ms, reflecting a 7.2% increase (Table 2.3). While these increases may not seem drastic, they can still pose challenges for time-sensitive tasks such as real-time medical imaging and autonomous systems, where even small latencies can impact decision-making. The effect is more pronounced in CPU-based environments, which experience a nearly 19.5% increase in inference time, whereas GPU-based inference sees a relatively smaller 7.2% increase due to optimized parallel processing. However, despite this increase, the adaptive resizer's computational demand remains small compared to the overall model complexity, and its benefits in segmentation accuracy and feature preservation make it a valuable enhancement for deep learning-based image analysis tasks.

One potential approach to balance computational efficiency and segmentation performance is a hybrid resizing strategy, where images are first downsampled to an intermediate resolution (e.g., 1,024×1,024) using bilinear interpolation, followed by adaptive resizing to the final target resolution (256×256). While this strategy helps reduce the computational overhead of the adaptive resizer, its main advantage is enhancing segmentation accuracy by allowing the adaptive resizer to focus on refining feature details at a more optimized scale. This hybrid two-step resizing process can help retain crucial high-frequency information, leading to improved segmentation performance in constrained environments. However, despite these advantages, this hybrid approach represents only a partial solution that does not fundamentally address the computational inefficiencies inherent in the current adaptive resizer architecture.

The substantial computational demands of the adaptive resizer highlight the necessity for developing more lightweight architectures that can achieve similar or better performance with reduced resource requirements. Designing inherently efficient resizer models is essential for enabling practical implementation in real-time medical imaging applications and resource-constrained environments. Such optimizations should focus on restructuring the core components of the resizer to minimize redundant operations while

preserving the feature extraction capabilities that make adaptive resizing valuable. By addressing these architectural limitations, future adaptive resizers could overcome current computational barriers and become more viable for widespread adoption in time-sensitive clinical applications and edge computing scenarios.

Despite the relatively small computational overhead introduced by the adaptive resizer compared to the overall model complexity, the increase in inference time reinforces the need for developing lightweight adaptive resizer models. The adaptive resizer increases the total FLOPs from an average of 433.55 billion to 446.58 billion, which corresponds to an approximate 3.0% increase, as shown in Table 2.3. Additionally, inference time on a GPU (NVIDIA A100-SXM4-40GB) increases from an average of 181.41 ms to 194.41 ms, reflecting a 7.2% increase. While these increases remain relatively minor compared to the overall computational cost, optimizing the resizer's architecture remains crucial. Reducing the computational cost while maintaining the benefits of feature preservation and segmentation accuracy is essential for real-time applications. Future work should focus on refining the resizer's design to achieve a more efficient balance between accuracy and computational efficiency, ensuring its practicality in resource-constrained environments.

2.4.5. Hyperparameter Tuning of Adaptive Resizer

Delving into the adaptive resizer's architecture, the original image traverses two distinct paths. One is solely the bilinear resizer, while the other is a more intricate route involving convolutional operations. The culmination of these paths results in the adaptive resizer's output. In the standard adaptive resizer, images from both paths are combined in equal proportions. In this section, the performance implications of varying these proportions are investigated. While the original resizer network maintains a 50:50 ratio, alternative ratios such as 20:80, 30:70, 40:60, 60:40, 70:30, and 80:20 were tested, resulting in six unique versions of the adaptive resizer. The accuracy metrics corresponding to these versions are presented in Figure 2.11, assessed using the attention U-Net model with the DenseNet201 backbone.

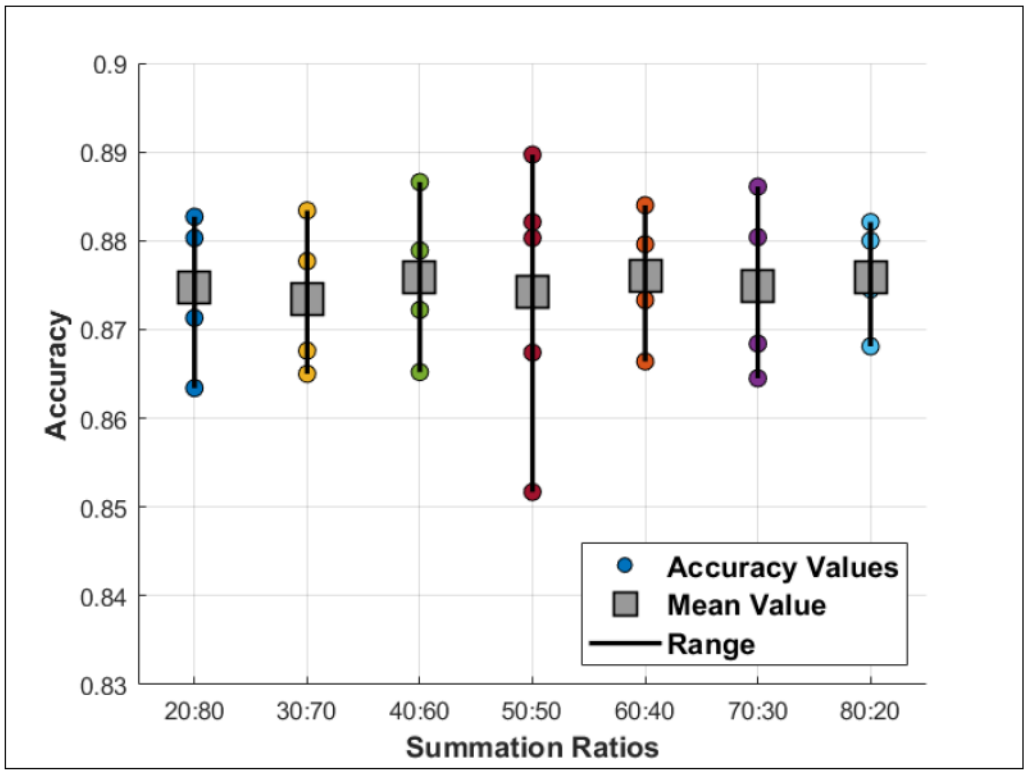

Figure 2.11. Validation accuracies of the adaptive resizer model across varying summation ratios between the outputs of convolutional blocks and the bilinear resizer branches (20:80, 30:70, 40:60, 50:50, 60:40, 70:30, 80:20). Colored dots represent individual fold accuracies, grey squares denote the mean value for each configuration, and vertical lines illustrate the accuracy range observed with 5-fold cross-validation.

Within the adaptive resizer's architecture, there's a segment comprising 'r' residual blocks. While the default configuration sets 'r' to 1, variations from 1 to 5 were investigated to gauge their impact on model performance.

Figure 2.12 displays the outcomes of adjusting the number of residual blocks within the adaptive resizer's architecture. Despite experimenting with five different configurations of residual blocks, no significant impact on accuracy was observed. The Attention U-Net model with DenseNet201 backbone did not exhibit substantial performance shifts in conjunction with variations in the adaptive resizer. This suggests that the segment of the adaptive resizer containing the residual blocks, which represents a minor component of the deep learning-based pathway, does not critically alter the model's effectiveness. As such, the quantity of residual blocks within the adaptive resizer's architecture is determined to have a marginal effect on the overall system's accuracy.

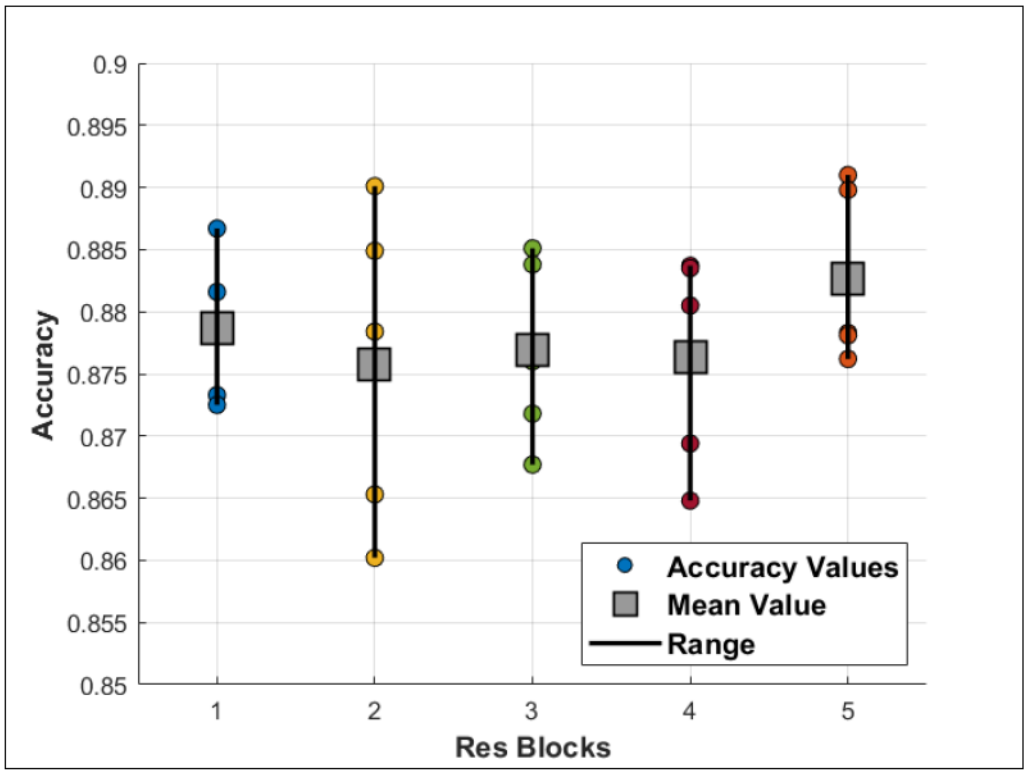

Figure 2.12. Validation accuracies of the adaptive resizer model across varying numbers of sequential residual blocks (1-5). Colored dots represent individual fold accuracies, grey squares denote the mean value for each configuration, and vertical lines illustrate the accuracy range observed with 5-fold cross-validation.

In summary, while Figures 2.11 and 2.12 illustrate our experiments with varying hyperparameters such as summation ratios and the number of residual blocks, the results demonstrate that these modifications have minimal or inconsistent effects on the adaptive resizer's performance without following any clear pattern. This finding indicates that simply fine-tuning these specific hyperparameters is not enough for meaningful performance improvements. However, the limited impact of hyperparameter variations reinforces the necessity of exploring deeper architectural optimizations beyond hyperparameter tuning to ensure that the adaptive resizer can be fully leveraged for more accurate and efficient image processing applications.

In this research, the adaptive resizer-based segmentation framework was investigated. The extensive evaluations detailed in this section confirm the significant benefits of integrating an adaptive resizer with U-Net-based segmentation models. The adaptive resizer not only improves the models' accuracy but also refines their capability to maintain crucial image details, enhancing the overall quality of segmentation. These improvements

are consistently observed across various model configurations and imaging conditions, demonstrating the adaptive resizer's robustness and versatility. The visual and quantitative analyses further validate the resizer's efficacy, making a compelling case for its adoption in advanced segmentation tasks. Moving forward, these insights could pave the way for more sophisticated adaptations and optimizations in medical image processing, potentially transforming current practices in the field.

2.5. Conclusion

The need for an adaptive resizer arises from the limitations of traditional resizing methods like bilinear resizing, which often result in significant information loss, particularly in high-resolution medical images. Such loss is especially problematic in segmentation tasks, where preserving spatial information is crucial for generating accurate pixel-wise predictions. Traditional resizing methods fail to adapt to the unique characteristics of medical images, leading to suboptimal segmentation performance. To address this, our study investigated the integration of an adaptive resizer module in segmentation, leveraging its ability to dynamically preserve essential image details while adapting to unique image characteristics. We proposed an adaptive resizer-based segmentation framework for the analysis of Colorectal Gland images. Gland segmentation is important since it allows for more accurate histopathological analysis, which is crucial for the diagnosis and staging of cancer. The influence of the adaptive resizing module on the efficacy of the segmentation task was investigated. The research embarked on an exploration of three different segmentation models, with seven different backbone choices utilizing a singular dataset. Performance comparisons using the adaptive resizer versus traditional bilinear resizing across three segmentation models were conducted, followed by analyses of computational time consumptions and hyperparameter modifications of adaptive resizer. The results revealed that the adaptive resizer significantly improved segmentation performance. Specifically, the integration of the adaptive resizer with segmentation models resulted in a marked improvement in the intersection over union (IoU) metric: a 6.47% increase for U-Net, 6.67% for Attention U-Net, and 3.79% for U-Net 3+. These results highlight the adaptive resizer's potential to enhance gland segmentation methodologies, making it a valuable tool in biomedical applications where precision and accuracy are critical. By preserving essential image details and adapting to unique image characteristics, the adaptive resizer ensures more accurate and efficient segmentation outcomes. Future efforts should re-evaluate the internal structure of the resizer, integrate

more selective feature processing strategies, and optimize computational efficiency to ensure maximal benefit across diverse imaging applications. These advancements will further refine the adaptive resizer into a more robust and versatile preprocessing tool, extending its applicability to a broader range of computer vision tasks beyond segmentation. Nevertheless, our study confirms that the adaptive resizer remains an effective solution for enhancing segmentation in high-resolution medical imaging. Given the associated increase in computational overhead, striking a balance between accuracy and efficiency becomes crucial, particularly for real-time processing or systems with limited computational resources. Overall, these findings highlight the resizer's ability to preserve spatial information, underscoring its potential value not only in segmentation tasks but also in other pixel-level computer vision applications.

CHAPTER 3

COMPARATIVE ANALYSIS OF NOVEL ADAPTIVE RESIZER ARCHITECTURES FOR IMAGE CLASSIFICATION AND SEGMENTATION TASKS

3.1. Study Abstract

High-resolution medical images often necessitate downsampling to align with the input constraints of deep learning models, a process that frequently leads to the loss of critical fine-grained features essential for accurate diagnosis. This study investigates the efficacy of various image resizing strategies, comparing traditional numerical methods against established adaptive resizing techniques and a novel suite of proposed adaptive resizer architectures. We evaluated these approaches on demanding medical imaging tasks: retinal vessel segmentation using the High-Resolution Fundus (HRF) dataset and diabetic retinopathy grading via a five-class classification task on the Indian Diabetic Retinopathy Image Dataset (IDRiD). Our research introduces new adaptive resizer architectures designed to optimize the trade-off between preserving task-relevant information and maintaining computational efficiency. Experimental results demonstrate that our proposed resizers generally outperform existing methods. Specifically, for segmentation, 'Resizer MFY' yielded the highest performance gains (average IoU increase of +21.04% over bilinear), while 'Resizer A2' proved most effective for classification tasks (average F1 score increase of +22.39% over bilinear). In terms of computational efficiency, the 'Minimal V1' architecture consistently emerged as the most lightweight among the proposed adaptive resizers, offering substantial performance improvements with a significantly lower computational overhead compared to other adaptive methods. This work underscores the potential of tailored adaptive resizers to enhance the accuracy of deep learning models in medical image analysis while carefully managing computational resources.

3.2. Introduction

The proliferation of high-resolution imaging in medical diagnostics has provided unprecedented detail for analysis. However, directly applying these high-resolution images to deep learning frameworks, particularly Convolutional Neural Networks (CNNs), presents practical limitations. CNNs are often designed with specific input dimensionality constraints due to architectural choices like fixed-size fully connected layers and the need to manage computational resources effectively. This necessitates downsampling high-resolution source images to standardized sizes (e.g., 224×224 or 256×256 pixels).

Traditional downsampling techniques, such as bilinear or bicubic interpolation, while simple and computationally efficient, operate on fixed mathematical rules. This often leads to a significant loss of critical fine-grained features and high-frequency details, which can be particularly detrimental in medical image analysis where subtle details are paramount for accurate diagnosis or precise segmentation. This information loss can limit the performance ceiling of sophisticated deep learning models.

Recognizing these limitations, adaptive image downsampling has emerged as a promising research area, aiming to tailor the downsampling process by considering image content or specific task requirements. A key innovation is the concept of a learnable resizer, notably introduced by (Talebi and Milanfar, 2021). This approach treats the image resizer as an integral, compact neural network module trained jointly with the main deep learning model, allowing it to learn task-specific downsampling. While the "Original Adaptive Resizer" demonstrated potential, as explored in Chapter 2, there remains a need for architectures that further enhance performance and, critically, improve computational efficiency.

This chapter (Chapter 3) builds upon these insights by proposing and rigorously evaluating a novel suite of adaptive resizer architectures. These new architectures are designed with the dual goals of improving performance in both image classification and segmentation tasks while also optimizing computational efficiency. Their efficacy is assessed on demanding medical imaging tasks: retinal vessel segmentation using the High-Resolution Fundus (HRF) dataset and diabetic retinopathy grading via a five-class classification task on the Indian Diabetic Retinopathy Image Dataset (IDRID). This study aims to provide tailored, computationally considerate solutions for resizing high-resolution medical images, thereby enhancing the accuracy and efficiency of subsequent deep learning models.

3.2.1. Related Works

The challenge of effectively resizing images for deep learning models, particularly in sensitive fields like medical imaging, has spurred a progression of research from traditional algorithms to sophisticated, learnable approaches.

Historically, image downsampling has been dominated by numerical methods such as bilinear and bicubic interpolation. These techniques are widely used due to their simplicity and computational efficiency. Bilinear interpolation, for instance, considers the four nearest pixels to compute the new pixel's value, while bicubic interpolation uses a larger neighborhood of 16 pixels for a generally sharper output. Lanczos interpolation employs a wider pixel window and a specialized mathematical function to minimize artifacts and maintain detail, though at a higher computational cost. Gaussian blurring is also often used as a pre-processing step before downsampling to mitigate aliasing artifacts, though it can soften the image. The primary drawback of these fixed numerical algorithms is their inherent tendency to cause information loss, especially of fine-grained features and high-frequency details, as they do not adapt to the image content. This loss can be particularly detrimental in medical image analysis where subtle details are crucial for diagnosis and segmentation.

Recognizing the limitations of fixed algorithms, the research community has explored more advanced downsampling techniques that aim to be content-adaptive. These methods seek to tailor the downsampling process by considering the image content or the specific requirements of the subsequent computer vision task. Strategies have included non-uniformly sampling image regions based on content importance, developing content-adaptive downsampling schemes within network architectures, and designing novel pooling layers that are more adaptive and information-preserving. Some approaches utilize edge-aware filtering, varying smoothing based on pixel similarity to preserve edges while smoothing flat areas. While these methods offer improvements, many early adaptive techniques still relied on handcrafted features or complex heuristics.

A significant advancement in adaptive downsampling is the concept of the learnable resizer, which integrates the resizing process into the deep learning pipeline as a trainable module. The foundational work by (Talebi and Milanfar, 2021) introduced such an adaptive resizer: a compact neural network module positioned at the beginning of a main deep learning architecture and trained jointly with it. This end-to-end training enables the resizer to learn how to downsample images in a way that is specifically optimized for the task at hand, aiming to retain features most salient for the model's

performance.

The introduction of this learnable resizer by Talebi and Milanfar spurred further research and adaptations across various domains:

(Li et al., 2023) developed the Model Adaptive Resizer (MAR), which modified the original learnable resizer by incorporating channel attention mechanisms within its "MAR blocks" for few-shot learning tasks.

(Zou et al., 2023) utilized a learnable resizer model in the domain of fault diagnosis for rotating machinery, applying it to classify signals that were converted into time-frequency images.

(Rahman, 2023) conducted an empirical analysis on integrating learnable resizers to enhance the performance of both classification and segmentation models in the medical field.

(Duzyel et al., 2023) showcased the utility of an adaptive resizer-based transfer learning framework for diagnosing breast cancer from histopathology images, finding that it significantly outperformed bilinear interpolation, particularly at high magnification factors.

(Han and Chen, 2021) integrated a learnable resizer with the MobileNet architecture for COVID-19 CT scan image classification, reporting improved accuracy over the original MobileNet and other standard CNNs.

(Zhang et al., 2022) proposed a minimalist learnable resizer, "Mini-resizer," for image geolocation, which employed convolution to project features into a higher dimensional space, followed by a self-attention mechanism for refinement.

While these learnable resizers have demonstrated considerable benefits, particularly in classification tasks, and Chapter 2 of this thesis affirmed their potential for segmentation, there is ongoing scope for improvement. The initial study in Chapter 2 highlighted that while the original adaptive resizer improved segmentation accuracy, it also significantly increased computational demands. This underscores the need for novel architectures that not only push the boundaries of performance but also offer better computational efficiency.

The current study (Chapter 3) is motivated by these observations. It aims to address the limitations of existing adaptive resizers by proposing and evaluating a new suite of architectures designed to optimize the trade-off between preserving task-relevant information for both segmentation and classification tasks and maintaining computational efficiency, especially in the context of high-resolution medical imaging.

3.2.2. Motivation for Novel Architecture Designs

While the original adaptive resizer and subsequent modifications have shown considerable promise, Chapter 2 of this thesis confirmed its efficacy for segmentation tasks but also highlighted the associated increase in computational load. Existing adaptive resizers might still face limitations in terms of achieving an optimal balance between performance enhancement across diverse tasks (like segmentation and classification) and computational efficiency. There is a clear need for resizer architectures that are not only effective in preserving task-critical information but are also computationally more efficient and robust across different types of medical imaging tasks.

This study directly addresses this gap by proposing a suite of six novel adaptive resizer architectures. These architectures have been developed through extensive experimentation and are specifically designed to optimize the trade-off between performance (for both segmentation and classification) and computational cost, with a focus on high-resolution medical image analysis.

3.3. Materials and Methodology

This section provides a thorough overview of the materials and methods used in the second major study of this thesis. It details the computational setup, the two high-resolution medical imaging datasets, and the diverse range of deep learning models selected for each task. Furthermore, it discusses various numerical and adaptive resizing strategies, introduces the six novel adaptive resizer architectures proposed in this work, and outlines the comprehensive experimental design and evaluation metrics used to assess their performance and computational cost.

3.3.1. Computational Setup

All model development, training, and evaluation were conducted using the Google Colaboratory platform, which provides a cloud-based environment equipped with configurable high-performance hardware resources. Two different runtime configurations within Colab were utilized based on the requirements of the task.

During model training, the environment was configured with an Intel® Xeon® CPU @ 2.20 GHz, 83.48 GB of system RAM, and an NVIDIA A100-SXM4-40GB GPU

with 39.56 GB of dedicated video memory. This high-performance setup enabled the efficient training of deep convolutional neural networks, particularly for high-resolution medical imaging tasks where both memory capacity and computational throughput are critical.

The Colab runtime was configured in CPU-only mode for model evaluation, especially to measure inference time and analyze computational overhead. In this setting, the system was equipped with an AMD EPYC 7B12 CPU and 50.99 GB of RAM, with no GPU detected. This configuration allowed for the assessment of model performance in CPU-limited scenarios, providing insights into the computational cost and deployment feasibility of the proposed architectures in environments lacking dedicated GPUs.

All models were implemented using the PyTorch deep learning framework, which offers flexibility and modularity for defining and training neural architectures. For classification tasks, architectures were sourced from the torchvision.models module, which includes various pre-trained models widely used in image classification. For semantic segmentation tasks, models were obtained from the segmentation-models-pytorch (smp) library, a high-level interface that provides standardized implementations of state-of-the-art encoder-decoder segmentation networks.

3.3.2. Datasets

This work utilizes two distinct datasets, selected for their high-resolution characteristics: one dedicated to a segmentation task and the other for a classification task.

3.3.2.1. Dataset for Segmentation Task: HRF

The High-Resolution Fundus (HRF) collection consists of 45 high-quality color fundus photographs, equally divided among three diagnostic categories: 15 images from healthy persons, 15 from patients with diabetic retinopathy, and 15 from those diagnosed with glaucoma. Each image has a resolution of roughly 3304x2336 pixels and is supplied in JPEG format. The dataset was specifically created for retinal vascular segmentation research and contains expert-annotated binary vessel maps for each image. (Budai et al., 2013) established it in their publication "Robust Vessel Segmentation in Fundus Images" in the International Journal of Biomedical Imaging, and it has subsequently been extensively utilized as a benchmark for assessing vessel segmentation algorithms.

This study used the HRF dataset just for vascular segmentation. A total of 36 images were assigned for training, while the remaining 9 images were allotted for testing. To maintain uniformity in input dimensions, all photos were initially padded vertically to make a square format and then cropped to a standardized size of 3504x3504 pixels, retaining the core content area. This preprocessing step was implemented to ensure compatibility with the utilized segmentation architectures.

3.3.2.2. Dataset for Classification Task: IDRiD

The Indian Diabetic Retinopathy Image Dataset (IDRiD) comprises a total of 516 high-resolution fundus images captured from diabetic patients, each accompanied by disease severity labels. These images were acquired using a fundus camera with a 50° field of view and have a resolution of approximately 4288x2848 pixels, stored in JPEG format. Out of the 516 images, 413 are designated for training and the remaining 103 for testing. The dataset provides ground truth labels at the image level based on the standard 5-level diabetic retinopathy (DR) grading scale, ranging from 0 (no DR) to 4 (severe DR), and includes annotations for diabetic macular edema (DME) severity. It serves as a benchmark for the development and evaluation of DR screening and classification algorithms. This dataset was introduced by (Porwal et al., 2018) in their work titled "Indian Diabetic Retinopathy Image Dataset (IDRiD): A Database for DR Screening Research," published in the Data journal.

In our study, we have specifically utilized the IDRiD dataset for the five-class diabetic retinopathy classification task. We did not use it for segmentation purposes or for the alternative three-class classification task. To enhance the spatial uniformity of the images, we first applied vertical padding to convert the original 4288x2848 images into square dimensions of 4288x4288 pixels. Subsequently, we performed a center-cropping operation to remove the padded margins and extract only the content-relevant area, resulting in final images of size 3550x3550 pixels. This preprocessing step ensured that the input data was better aligned with the requirements of our classification pipeline.

3.3.3. Segmentation and Classification Models

This section outlines the various deep learning architectures employed for the primary tasks of retinal vessel segmentation and diabetic retinopathy classification. For

each task, a selection of established and most known models was utilized, leveraging their distinct architectural strengths.

3.3.3.1. Segmentation Models

Several state-of-the-art convolutional neural network architectures were employed for the segmentation task, specifically retinal vessel segmentation. One of the foundational models utilized was U-Net, a fully convolutional network characterized by its symmetric encoder-decoder structure and skip connections. The model's "U"-shaped architecture facilitates the fusion of low-level and high-level feature representations through these skip pathways, enabling precise pixel-wise segmentation. U-Net was originally proposed by Ronneberger et al. for biomedical image segmentation and demonstrated strong performance in settings with limited training data (Ronneberger et al., 2015).

An enhanced version of the original U-Net, known as U-Net++, was also considered. U-Net++ introduces a series of nested dense skip connections between the encoder and decoder sub-networks, which serve to reduce the semantic gap between the corresponding feature maps. This architecture further incorporates deep supervision during training, promoting more effective gradient flow and feature learning. (Zhou et al., 2018) reported that U-Net++ achieves improved segmentation accuracy compared to the standard U-Net by leveraging these densely connected up-sampling paths.

Another architecture evaluated was DeepLabV3+, which extends the DeepLabV3 framework by incorporating an encoder-decoder design. The model utilizes an Atrous Spatial Pyramid Pooling (ASPP) module to capture contextual information at multiple scales using dilated convolutions. A lightweight decoder is subsequently employed to refine segmentation outputs, particularly along object boundaries. The addition of the decoder and ASPP was shown by (Chen et al., 2018) to significantly improve segmentation of fine structural details while maintaining strong performance on global context.

The study also included MA-Net (Multi-scale Attention Network), which enhances feature representation through the integration of two complementary attention mechanisms: a position-wise attention block that models long-range spatial dependencies, and a channel-wise attention block that supports multi-scale feature fusion. This dual attention framework enables MA-Net to capture both global context and discriminative local features, improving segmentation accuracy on challenging datasets. The effectiveness of MA-Net was validated by (Li et al., 2020), particularly in remote sensing applications, where it outperformed earlier architectures.

Finally, LinkNet was selected due to its efficiency-oriented design. LinkNet employs an encoder and a lightweight decoder, with skip connections bridging each encoder block to its corresponding decoder block. This structure significantly reduces computational cost while maintaining high segmentation performance. The architecture, introduced by (Chaurasia and Culurciello, 2017), was shown to match the accuracy of more computationally intensive models while offering substantially faster inference.

It is important to note that for all the segmentation architectures discussed—U-Net, U-Net++, DeepLabV3+, MA-Net, and LinkNet—a consistent encoder strategy was adopted. Each of these models utilized a ResNet-50 architecture, pre-trained on the ImageNet dataset, as its primary feature extraction backbone. This approach ensures that the variations in performance can be more directly attributed to the unique decoder designs and feature integration strategies of each network, rather than differences in the initial feature encoding stage.

3.3.3.2. Classification Models

For the diabetic retinopathy severity classification task, a diverse set of convolutional neural network (CNN) architectures from the `torchvision.models` library was employed, each pre-trained on ImageNet and subsequently fine-tuned for a five-class output.

DenseNet-121 is a compact and efficient architecture characterized by densely connected blocks in which each layer receives the feature maps of all preceding layers as inputs (Huang et al., 2017). This dense connectivity alleviates vanishing gradients and promotes feature reuse. The final classifier layer of DenseNet-121 was adapted accordingly.

ResNeXt-50 (32×4d) introduces the concept of cardinality, which expands representational power by increasing the number of parallel transformations within each residual block rather than its depth or width (Xie et al., 2017). This model structure enables performance gains without significant increases in complexity, and its output layer was similarly modified for five-class classification.

ShuffleNet V2, designed for resource-constrained environments, employs channel shuffling and grouped pointwise convolutions to optimize actual runtime performance on mobile hardware (Ma et al., 2018). The architecture achieves an excellent trade-off between accuracy and efficiency, and the final fully connected layer was replaced for task-specific adaptation.

GoogLeNet, also referred to as Inception v1, is a 22-layer deep CNN that introduced the Inception module—an architectural innovation that processes multi-scale convolutional operations in parallel within each block (Szegedy et al., 2015). Its classifier head was adjusted to match the classification target.

MnasNet, a mobile-optimized model developed via platform-aware neural architecture search (NAS), strikes a balance between inference latency and accuracy. The architecture was tuned to perform efficiently on mobile CPUs while maintaining competitive performance (Tan et al., 2019). Its final classifier layer was modified for five-class prediction.

RegNetX-1.6GF is a variant within the RegNet family, which was derived from design space exploration of regular network patterns with quantized linear progression in width and depth (Radosavovic et al., 2020). This architecture offers favorable accuracy-speed trade-offs, particularly on GPU hardware, and its fully connected head was replaced during fine-tuning.

Finally, EfficientNet-B1, which was designed through compound scaling of depth, width, and resolution based on a baseline NAS-discovered model, was included in the evaluation. EfficientNet-B1 provides strong accuracy with reduced parameter count and computational cost, making it suitable for medical imaging applications (Tan and Le, 2019). Its classifier was adjusted to output five classes.

3.3.4. Numerical Resizing and Adaptive Resizing

Image downsampling is the process of reducing the spatial resolution of an image. This action is often done to reduce file size and decrease processing time of the computations. When downsampling, it's crucial to do it in a way that minimizes artifacts like aliasing and preserves as much important image detail as possible. Several interpolation techniques are used to calculate the pixel values for the new, smaller image based on the original pixel values. Interpolation is essentially an "educated guess." The software needs to decide what color a new pixel should be, based on the colors of the existing pixels in the original image. Different interpolation methods use different approaches to make these guesses, leading to varying results in terms of image quality, sharpness, and processing time. In our experiments, numerical downsampling methods and adaptive downsampling modules were compared. Numerical downsampling algorithms do not perform an iterative process; they calculate the pixel values in the target matrix using the pixel values in the source matrix within a certain rule.

Some of the most well-known numeric resizing methods are bilinear, bicubic, Lanczos, and Gaussian resizing algorithms.

Bilinear interpolation looks at the four closest existing pixels (a 2×2 square) that surround the location of the new pixel. It then calculates the new pixel's color by taking a weighted average of these four neighbors. This method is generally used in tasks where speed is a consideration and sharpness is not a priority. Thus, it causes loss of fine details and appears a bit soft and blurry.

Bicubic interpolation is a more complex method than bilinear interpolation. Instead of using the 4 nearest pixels, this method uses 16 nearest pixels (a 4×4 square). This method generally produces sharper and more detailed images compared to bilinear interpolation. Image editing programs use it as a default for better quality resizing. It is also more computationally intensive than bilinear interpolation, meaning it takes longer to process.

A more sophisticated technique is Lanczos interpolation. It seeks to produce downsampling tasks with high-quality results. It examines an even wider range of the original pixels (for instance, a popular variant called Lanczos-3 takes into account a window of 6×6 or 8×8 surrounding pixels). It determines the weights for these nearby pixels using a unique mathematical function. The goal of this function is to minimize undesired artifacts while maintaining as much detail as possible. This algorithm is very good at maintaining the sharpness and detail of images, minimizing aliasing, and preventing the blurring that comes with using simpler techniques. The primary disadvantage is its computational expense; because of the complexity of its computations and the larger number of pixels it processes, it is among the slowest popular interpolation techniques.

The Gaussian method is not an interpolation technique. Rather, it is frequently employed as a pre-processing step. Before discarding pixels or calculating new ones, the original image is supposed to be slightly blurred. It averages pixels smoothly, giving central pixels more weight and distant pixels less, resembling a bell curve. Aliasing artifacts can be effectively avoided by applying a Gaussian blur prior to downsampling. Blurring naturally softens and lessens the sharpness of the image. The final downsampled image may appear too soft or lack detail if the blur is too strong. Applying just enough blur to avoid aliasing without sacrificing too much crucial detail is crucial. It gives the resizing process an additional step. Combining Gaussian blur and Lanczos downsampling is one of the comparison techniques used in the experiments.

Adaptive downsampling techniques adopt a more complex strategy by examining the image content either prior to or during the resizing process, in contrast to the fixed nu-

merical algorithms previously discussed. These methods modify their behavior according to local image features, like the existence of edges, textures, or smooth areas, rather than imposing a consistent rule over the whole image. While more aggressive smoothing or information reduction may be applied in less crucial, uniform areas, the main objective is frequently to maintain visually significant features, such as sharp edges that define objects. Instead of using a single fixed filter everywhere, this content-aware approach seeks to improve the downsampled image's perceptual quality.

Adaptive downsampling can be applied in several ways. Some approaches may make use of edge-aware filtering techniques, which smooth flat areas while maintaining edges by varying the amount of smoothing according to pixel similarity. Others may use distinct interpolation techniques in accordance with the explicit detection of distinct region types. Additionally, learning-based methods, which frequently make use of deep neural networks, represent an important category. Adaptive approaches, especially learning-based ones, can be much more complicated and computationally demanding than conventional numerical techniques, even though they may produce better visual results. The primary focus of this study is the mentioned learning-based approach.

3.3.5. Adaptive Resizer Architectures

In order to improve performance and computational efficiency in image resizing tasks, this study presents six innovative adaptive resizer architectures. Each of these has distinct qualities. Out of more than 60 different designs that were created and assessed, these six architectures were chosen. For a comparative baseline, the "Original Resizer" architecture is first discussed.

Original Resizer: As shown in Figure 3.1, this architecture employs a two-branch design. The first branch directly resizes the input image to the target dimensions using bilinear interpolation. The second branch initially applies two convolutional layers (with 16 kernel tensors of size $7\times7\times3$ and 16 of size $1\times1\times16$, respectively) to the input tensor. This feature-extracted tensor is then resized using bilinear interpolation, followed by further convolutional operations. Finally, the outputs from these two branches are summed to produce the resized image.

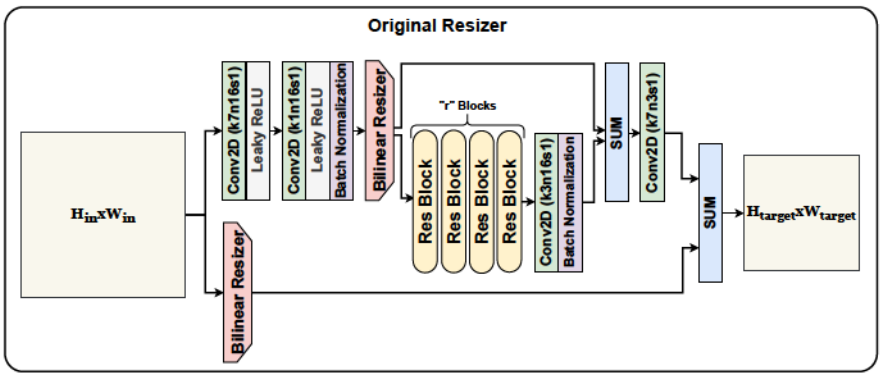

Figure 3.1. Original resizer architecture

Minimal V1: This architecture, shown in Figure 3.2, first interpolates the input tensor's width or height to four times the target resolution using bilinear interpolation. Subsequently, two strided convolution operations, each with a stride $s = 2$, are applied to this tensor, bringing the output tensor to the target size. The relationship between input size (I), kernel size (K), padding (P), stride (S), and output size (O) of a convolutional layer is given by Equation 3.1.

$$O = \frac{I - K + 2P}{S} + 1 \tag{3.1}$$

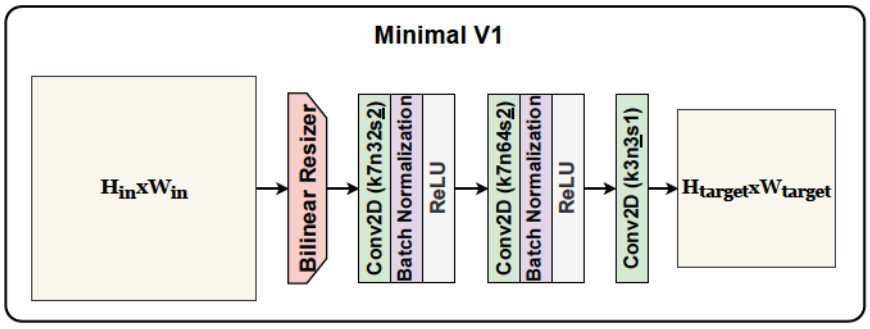

Figure 3.2. Minimal V1 resizer architecture

Minimal V2: Depicted in Figure 3.3, applies two strided convolutions (with $s = 2$) to the input tensor first. The resulting tensor is then interpolated to the target size using bilinear interpolation.

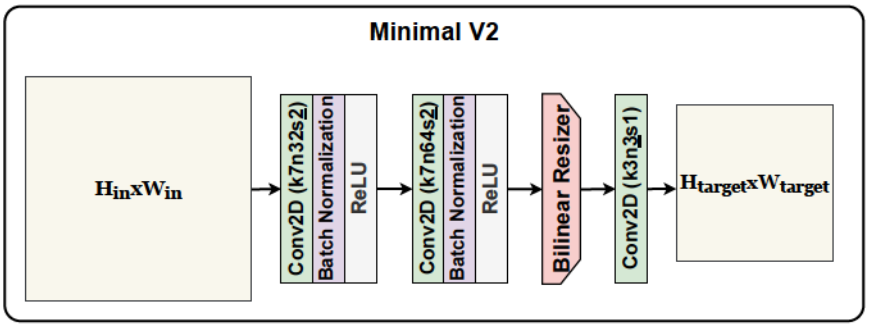

Figure 3.3. Minimal V2 resizer architecture

Resizer A: Shown in Figure 3.4, is a dual-branch architecture. The first branch implements the Minimal V1 pipeline, and the second branch implements the Minimal V2 pipeline. The 3-channel tensor outputs from both branches are concatenated and then fused via a final convolutional layer to produce the 3-channel output image.

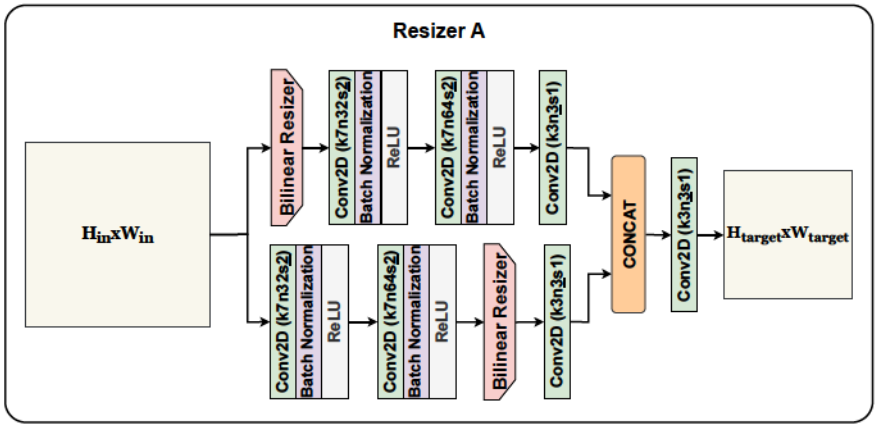

Figure 3.4. Resizer A architecture

Resizer A2: Illustrated in Figure 3.5, is a variant of Resizer A. The projection convolutions at the end of the individual Minimal V1 and Minimal V2 branches, which would convert feature maps to 3 channels within each branch, are omitted. Instead, the two 64-channel tensors (assuming 64 channels before projection in the branches) are directly concatenated, followed by a final convolutional layer to produce the 3-channel output.

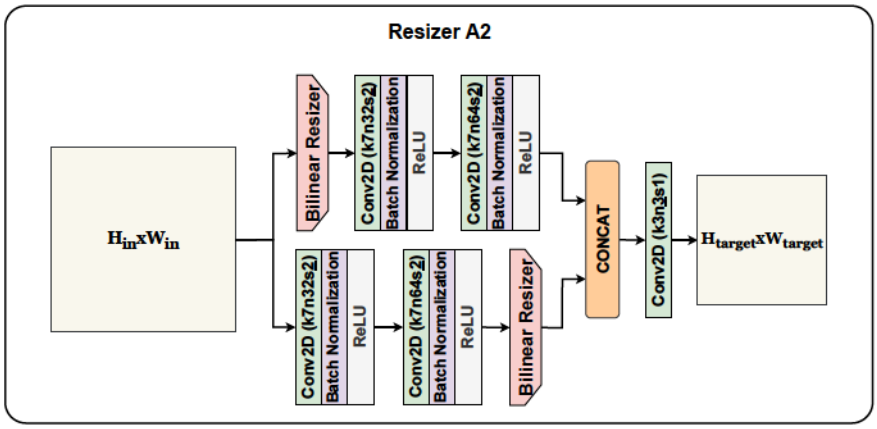

Figure 3.5. Resizer A2 architecture

Resizer SK: This architecture, shown in Figure 3.6, builds upon the Minimal V2 baseline by integrating vision attention mechanisms. Specifically, the Selective Kernel (SK) attention mechanism is employed. Resizer SK has three branches: the first is the Minimal V2 pipeline; the second performs direct bilinear resizing of the input to the target dimensions; and the third auxiliary branch first bilinearly resizes the image to the target dimensions, then expands it into a 32-channel tensor, and subsequently processes it through an SK attention module. The outputs from these three branches are concatenated and then projected to a 3-channel tensor via a final convolution.

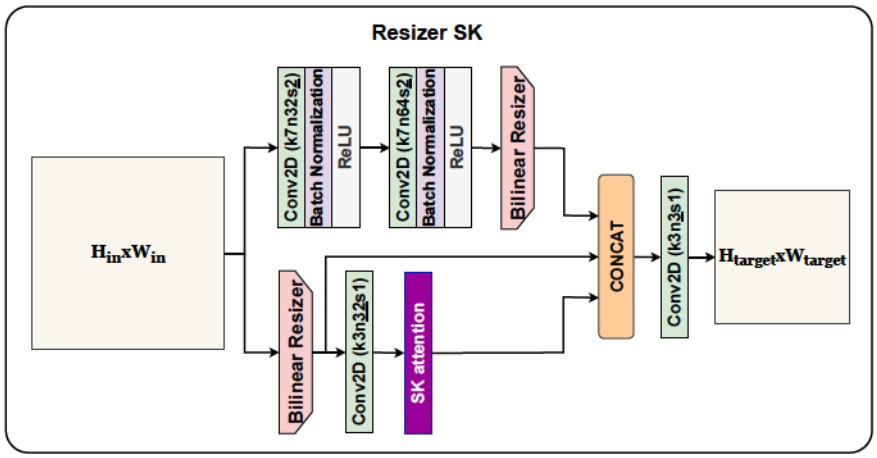

Figure 3.6. Resizer SK architecture

Resizer MFY: Depicted in Figure 3.7, is a three-branch architecture that extends the Resizer A design. Two branches follow the Resizer A structure (derived from Minimal V1 and Minimal V2). The third, new branch takes the input tensor, brings it to the target resolution via bilinear resizing, and then processes it with a bank of fixed 3×3 filters (e.g., Sobel, Prewitt, Laplacian, emboss, Schaar, edge detection, strong sharpen, vertical lines, and horizontal lines). Each of these fixed filters operates on the 3-channel image to produce a 3-channel output; their concatenation results in a tensor with $3 \times N_{filters}$ channels (where $N_{filters} = 12$ in this design, leading to a 36-channel tensor). The outputs from all three branches are concatenated and then convolved to produce the final 3-channel output image.

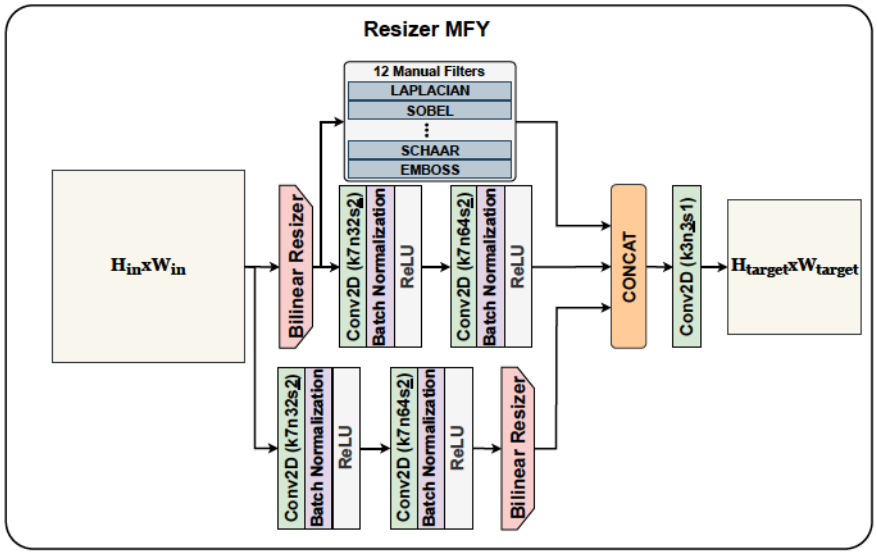

Figure 3.7. Resizer MFY architecture

3.3.6. Experimental Designs for Resizer Evaluation

To properly evaluate and compare the resizer architectures, we designed meticulous experimental setups for both segmentation and classification. These setups specify the data handling, model training, and evaluation strategies for each task. For segmentation on the HRF dataset, a standardized pipeline was followed from image preprocessing to model training and testing. For classification, we adopted a multi-faceted approach

using the IDRID dataset across different input and output resolutions to ensure a rigorous and comprehensive assessment of the resizers' capabilities.

3.3.6.1. Segmentation Task Experimental Setup

Specifically addressed to the task of semantic segmentation of retinal vessels using the High-Resolution Fundus (HRF) image dataset, this part describes the experimental methods used to rigorously evaluate the effectiveness of several proposed adaptive resizer architectures against conventional methods. Figure 3.8 shows the brief overview of the model training strategy.

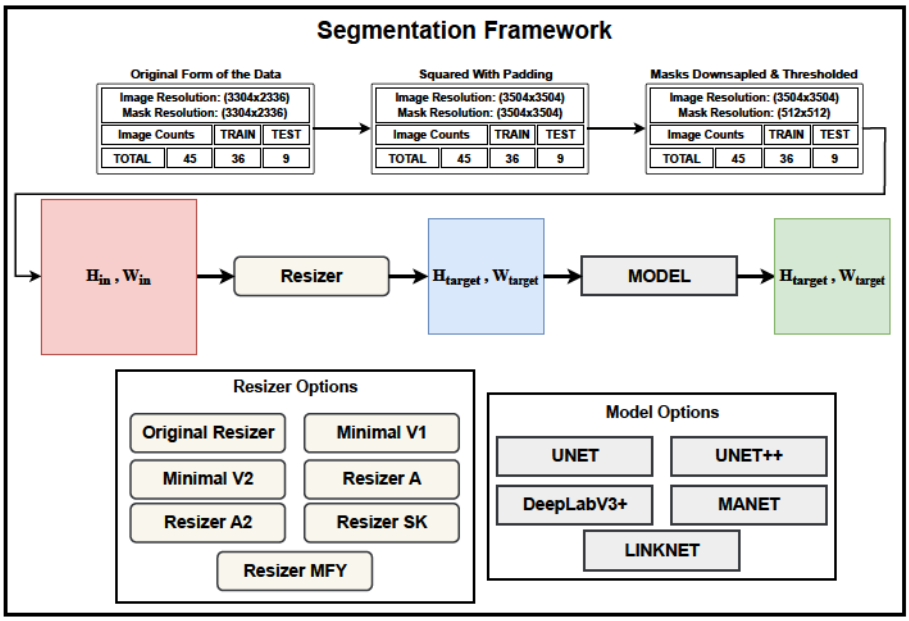

Figure 3.8. An overview of the segmentation pipeline designed to evaluate and compare various image resizing techniques. High-resolution fundus images (padded to 3504x3504) are downsampled to 512x512 using a selection of resizers. These resized images then serve as input for several established segmentation models. This framework allows for a rigorous comparison of how different resizing methods affect the final segmentation accuracy for each model architecture.

Experimentation used the 45 high-resolution retinal fundus pictures in the HRF dataset, each originally measuring 3304x2336 pixels. Pictures were preprocessed using

padding to produce square pictures with dimensions of 3504x3504 therefore achieving homogeneity and eliminating distortions resulting from aspect ratio variations. The dataset was then split into a training set of 36 images and a testing set comprising 9 images.

The main goal of the resizing work was to downsample these 3504x3504 pixel preprocessed photos to a much smaller target resolution of 512x512 pixels. This significant decrease in resolution presents special difficulties for the preservation of delicate vascular structures needed for precise segmentation.

Five well-known semantic segmentation networks were chosen: U-Net, U-Net++, DeepLabV3+, MA-Net, and LinkNet (with a ResNet-50 backbone) to methodically assess the efficacy of each resizing technique across various model topologies.

Eight different resizing techniques were used: Minimal V1, Minimal V2, Resizer A, Resizer A2, Resizer SK, and Resizer MFY; one traditional interpolation method (bilinear interpolation); one previously established learnable resizer architecture (Original Resizer) introduced by Talebi & Milanfar.

Corresponding ground truth segmentation masks were downsized in parallel with their respective input pictures (from 3504x3504 to 512x512 pixels) using identical scaling techniques to ensure consistency in evaluation. Resizing caused interpolation, hence masks had grayscale values between 0 and 255. Higher threshold values degraded the portrayal of fine vascular structures, according empirical testing with thresholds of 128, 100, and 70. Therefore, a threshold of 70 was found as ideal to efficiently binarize the masks into a distinct segmentation target while maintaining thorough vessel information.

Every combination of segmentation models and resizing techniques, resulting in a total of 40 unique combinations, calculated as five models multiplied by eight resizers, underwent training for one hundred epochs. Using the intersection over union (IoU) metric as the criterion for model selection, the model performance was tracked using a validation subset (formed from the training set). Saved and subsequently used for testing were model weights producing the best IoU on this validation set.

Every model-resizer configuration was trained independently five times to guarantee strong resistance against variability resulting from random initializations and stochastic optimization methods. By averaging the results of these five independent runs, final reported measurements improve the statistical dependability of the obtained data.

Especially with regard to their potential to preserve segmentation accuracy when downsampling high-resolution retinal pictures crucial for medical diagnosis, this thorough experimental framework offers an in-depth comparative examination of the suggested adaptive resizer architectures.

3.3.6.2. Classification Task Experimental Setup

Using the Indian Diabetic Retinopathy Image Dataset (IDRiD), this part describes the experimental methodologies used to assess the suggested adaptive resizer structures versus baseline approaches in a multi-class classification problem, namely diabetic retinopathy (DR) grading. The brief overview of the experimental framework for training can be seen in Figure 3.9.

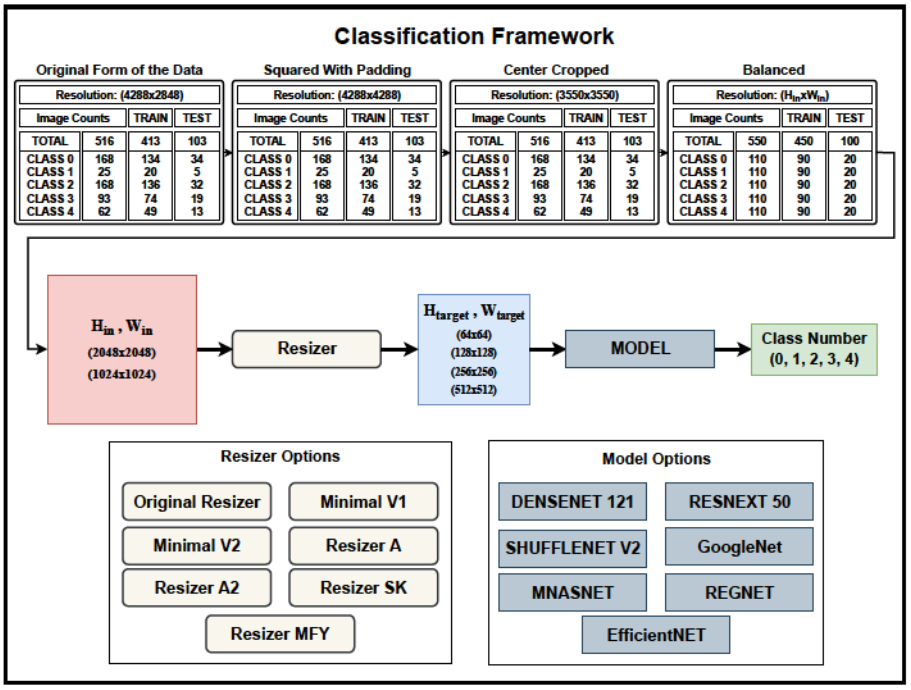

Figure 3.9. The comprehensive experimental framework for Diabetic Retinopathy (DR) classification. The top panel illustrates the four-stage data preparation process: starting with the original imbalanced IDRiD dataset, followed by padding, center cropping, and a balancing procedure to create uniform class distributions for training (450 images) and testing (100 images). The main pipeline then processes these images by downsampling them from a given input resolution (e.g., 2048x2048) to a target resolution (e.g., 256x256) using one of the available resizer options. Finally, the resized image is classified into one of five DR grades by one of seven different CNN models.

We used the 516 retinal fundus images from the IDRiD collection, originally 4288x2848 pixels. With 413 training images and 103 testing images, the dataset first comprised five different DR grades (Classes 0 through 4). A two-step balancing approach

was followed considering the underlying class imbalance in the initial distribution. First, underrepresented classes underwent data augmentation in order to synthetic increase their sample count. Second, selectively reduced photos from overrepresented classes. As so, this balancing produced a balanced testing dataset of 100 photos (20 images per class) and a homogeneous training dataset comprising 450 images (90 images per class). Images then underwent required padding and cropping to generate uniform square dimensions of 3504x3504 pixels.

The main experimental phase of first classification was two-stage downsampling. The standardized 3504x3504 photos were first scaled to an intermediary resolution of 2048x2048 pixels then further down-sized to a target level of 256x256 pixels. Six new adaptive resizer architectures proposed in this work, Minimal V1, Minimal v2, Resizer A, Resizer A2, Resizer SK, and Resizer MFY; one traditional baseline method (bilinear interpolation) and one established learnable resizer (Original Resizer) were tested. DenseNet121, EfficientNetB1, GoogLeNet, MnasNet1.0, ResNeXt50, RegNet, and ShuffleNetV2.0 were among seven different convolutional neural network (CNN) designs chosen for the classification challenge. This produced 56 total unique model-resizer combinations (8 resizing techniques $\times$ 7 classification models).

Expanding the experimental setting helped to methodically evaluate the impacts of different input and target resolutions on model correctness and computational complexity. Two alternative input resolutions—1024x1024 and 2048x2048 pixels—as well as four different target resolutions—64x64, 128x128, 256x256, and 512x512—were used. Each of the previously defined 56 model-resizer combinations thus underwent training and evaluation over these eight resolution configurations (2 input resolutions $\times$ 4 target resolutions).

Training and Evaluation Strategy: Every classification experiment was run consistently throughout one hundred training epochs. The model maintaining the best F1-score on a validation subset—derived from the balanced training set—was kept as the optimal model over training. This decision motivated by validation guaranteed dependability and strength of the outcomes. Specifically selected was the F1-score measure since it fit multi-class situations with balanced class distributions following augmentation.

This all-encompassing experimental design helps one to closely investigate the interactions among resizer designs, resolution strategies, and classification accuracy. It also offers new perspectives on computational accuracy and efficiency trade-offs in the framework of retinal disease categorization activities by using adaptive resizers.

3.3.7. Evaluation Metrics: Performance and Complexity

A comprehensive set of metrics was employed to assess the effectiveness of the different resizing strategies across classification and segmentation tasks, as well as to evaluate their computational efficiency. Although only selected metrics, the F1-score for classification and the intersection over union (IoU) for segmentation, are presented in the results section for clarity and conciseness, all listed metrics were computed during experimentation to ensure a thorough and reliable performance comparison.

For the classification tasks, several evaluation metrics were considered. Accuracy, which represents the overall proportion of correctly classified samples, is defined by Equation 3.2. where TP, TN, FP, and FN represent the number of True Positives, True Negatives, False Positives, and False Negatives, respectively.

While accuracy provides a general measure of performance, it may be misleading in class-imbalanced datasets. To provide deeper insight, precision, recall, and F1-score were also computed. Precision, quantifying the proportion of predicted positive instances that are truly positive, is given by Equation 3.3.

Recall (also known as sensitivity), which measures the proportion of actual positive instances that were correctly predicted, is calculated as shown in Equation 3.4.

The F1-score combines precision and recall into a single metric using the harmonic mean, as expressed in Equation 3.5.

This score is particularly informative for imbalanced class distributions. In the multi-class classification task conducted on the IDRiD dataset, these metrics were computed for each class individually and then averaged using macro or weighted averaging strategies.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}} \tag{3.2}$$

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} \tag{3.3}$$

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} \tag{3.4}$$

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}} \tag{3.5}$$

For the segmentation tasks, the primary metric used was the Intersection-over-Union (IoU), also known as the Jaccard Index. IoU, which measures the overlap between the predicted segmentation mask and the ground truth mask, is formulated in Equation 3.6.

Here, for segmentation, TP represents the area of true positive predictions, FP the area of false positives, and FN the area of false negatives. A higher IoU value indicates a better overlap between predicted and actual segmentation regions, with a perfect segmentation yielding an IoU of 1. Additionally, the Dice Similarity Coefficient (DSC), also referred to as the F1-score for segmentation, was calculated to measure spatial overlap between binary masks. The Dice score is given by Equation 3.7.

It is closely related to IoU, and this relationship is detailed in Equation 3.8.

Although the segmentation task in this study was binary (vessel vs. background), for multi-class segmentation problems, mean IoU (mIoU) can be computed by averaging the IoU values across all classes.

$$\text{IoU} = \frac{\text{Area of Overlap}}{\text{Area of Union}} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}} \tag{3.6}$$

$$\text{DSC} = \frac{2 \times \text{Area of Overlap}}{\text{Area of Prediction} + \text{Area of Ground Truth}} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}} \tag{3.7}$$

$$\text{DSC} = \frac{2 \times \text{IoU}}{\text{IoU} + 1} \tag{3.8}$$

To evaluate computational efficiency, a set of resource-aware metrics was also considered. The number of parameters quantifies the total trainable weights in the resizer module or the combined model, providing an estimate of memory usage. FLOPs (Floating Point Operations) denote the number of arithmetic computations (additions and multiplications) required during a single forward pass, serving as a hardware-agnostic measure of computational cost. Additionally, MACs (Multiply-Accumulate Operations) were monitored, which represent a fundamental operation in convolutional layers. Their relationship to FLOPs is often approximated by Equation 3.9.

$$\text{FLOPs} \approx 2 \times \text{MACs} \tag{3.9}$$

Lastly, inference time was measured as the average wall-clock time required to process a single image during testing, reported in milliseconds and converted to frames per second (FPS) where appropriate. Inference time was evaluated under both GPU and CPU environments to assess the real-world deployment feasibility of different resizer and model configurations.

This multi-faceted evaluation framework ensures a rigorous comparison of task performance, localization precision, and computational demands across all examined resizing strategies and model architectures.

3.4. Results

Here, we present the empirical findings from our extensive experiments evaluating the novel adaptive resizer architectures. The results are organized into two key areas. First, we provide a comparative performance analysis of the resizers for both segmentation and classification, using metrics like IoU and F1-score to quantify accuracy. Second, our analysis delves into the computational overhead introduced by each resizing method, offering a detailed breakdown of its impact on parameters, FLOPs, and processing time.

3.4.1. Comparative Results

The performance outcomes of the various resizing strategies are compared in detail for both segmentation and classification tasks. Our analysis is driven by quantitative metrics—primarily the Intersection over Union (IoU) for segmentation and the F1-score for classification. The results are systematically broken down to highlight how each resizer, from traditional numerical methods to the novel adaptive architectures, impacts the accuracy of different deep learning models under various experimental conditions.

3.4.1.1. Performance Comparison for Segmentation

The results of evaluating various image resizing methods for a binary semantic segmentation task on the HRF dataset are presented in Table 3.1. These experiments involved a consistent downsampling of input images from an original resolution of 3504x3504 pixels to an output resolution of 512x512 pixels prior to model processing.

The table details the performance of five distinct segmentation models—U-Net, U-Net++, DeepLabV3+, MA-Net, and LinkNet—when preceded by different image resizing techniques.

Table 3.1. Comparison of Segmentation Performance by Resizer and Model Architecture. The values indicate the Intersection over Union (IoU) score, with the percentage increase shown in parentheses relative to the Bilinear baseline.

Resizers	Metrics	UNET	UNET++	DeepLabV3+	MANET	LINKNET	Average Increase (%)
Bilinear	IoU Score	0.5617	0.5787	0.4665	0.5482	0.5401	(0.00%)
	Increase (%)						
Bicubic	IoU Score	0.5746	0.6031	0.4869	0.5835	0.5683	(4.51%)
	Increase (%)	2.29%	4.21%	4.37%	6.43%	5.22%	
Lanczos	IoU Score	0.5916	0.6075	0.4929	0.5924	0.5723	(5.99%)
	Increase (%)	5.32%	4.98%	5.66%	8.06%	5.96%	
Gaussian	F1 Score	0.5973	0.6105	0.4869	0.5909	0.5662	(5.77%)
	Increase (%)	6.34%	5.49%	4.37%	7.79%	4.83%	
Original Resizer	IoU Score	0.6601	0.6602	0.5256	0.6413	0.6121	(14.92%)
	Increase (%)	17.52%	14.09%	12.67%	16.99%	13.33%	
Minimal V1	IoU Score	0.6835	0.6923	0.5229	0.6775	0.6581	(19.77%)
	Increase (%)	21.68%	19.63%	12.09%	23.59%	21.84%	
Minimal V2	IoU Score	0.6792	0.6916	0.5338	0.6734	0.6546	(19.78%)
	Increase (%)	20.93%	19.51%	14.43%	22.84%	21.20%	
Resizer A	IoU Score	0.6887	0.6919	0.5371	0.6779	0.6487	(20.22%)
	Increase (%)	22.62%	19.57%	15.14%	23.67%	20.11%	
Resizer A2	IoU Score	0.6877	0.6971	0.5348	0.6751	0.6668	(20.83%)
	Increase (%)	22.43%	20.46%	14.64%	23.16%	23.46%	
Resizer SK	IoU Score	0.6907	0.6951	0.5277	0.6766	0.6474	(19.90%)
	Increase (%)	22.97%	20.12%	13.12%	23.43%	19.87%	
Resizer MFY	IoU Score	0.6896	0.6931	0.5382	0.6833	0.6625	(21.04%)
	Increase (%)	22.77%	19.75%	15.37%	24.65%	22.66%	

The rows delineate the resizing methods, encompassing standard numerical approaches such as Bilinear (serving as the baseline), Bicubic, Lanczos, and Gaussian, alongside a suite of advanced and adaptive resizers including the Original Resizer, Minimal V1, Minimal V2, Resizer A, Resizer A2, Resizer SK, and Resizer MFY. For each combination of a resizing method and a segmentation model, the table primarily reports the intersection over union (IoU) Score; notably, these IoU values in each cell for the

models are obtained by the average of 5 runs, providing a robust measure of performance. Crucially, the table also provides the percentage increase in this average IoU score relative to the bilinear resizer's performance with the same model. The rightmost column aggregates these improvements, showing the average percentage increase in IoU score for each resizing method across all five segmentation models, offering a clear overview of their general efficacy. While the gaussian method's primary scores are listed as F1 Scores, its percentage increase and average increase are calculated relative to the bilinear method's IoU scores, maintaining consistency for comparative analysis of improvements.

Analyzing the overall average performance, as indicated by the "Average Increase (%)" in IoU score, reveals a clear hierarchy among the resizing techniques. The Resizer MFY method demonstrated the most significant improvement, achieving the highest average increase of +21.04% over the bilinear baseline. Closely following was Resizer A2, with an average IoU gain of +20.83%. Other adaptive methods also delivered substantial enhancements: Resizer A showed an increase of +20.22%, Resizer SK provided a +19.90% gain, and both Minimal V2 and Minimal V1 yielded strong improvements of +19.78% and +19.77%, respectively. This pattern highlights a consistent advantage for these more sophisticated resizing approaches in the context of semantic segmentation on this dataset.

The Original Resizer also offered a considerable benefit, with an average IoU increase of +14.92%, positioning it as a strong performer, though moderately below the top tier of adaptive methods. Among the standard numerical resizing techniques, all showed an improvement over the Bilinear baseline. The Lanczos filter provided the best performance in this category with an average IoU increase of +5.99%. The Gaussian method followed with a +5.77% average increase, and the Bicubic method resulted in a +4.51% average gain. These results suggest that even within classical numerical resizers, choices like Lanczos can offer a noticeable, albeit smaller, advantage over Bilinear for segmentation tasks.

Across the individual segmentation models, the advanced resizing methods generally maintained their superior performance. For instance, Resizer MFY consistently delivered high percentage increases for models like MA-Net (+24.65%) and U-Net (+22.77%). Similarly, Resizer A2 showed robust gains across models, such as with LinkNet (+23.46%) and U-Net (+22.43%). While there were minor variations in the exact ranking of the top methods for each specific model, the collective group of Resizer MFY, Resizer A2, Resizer A, Resizer SK, Minimal V2, and Minimal V1 consistently outperformed the bilinear baseline by a significant margin across all tested segmentation architectures.

The improvements were evident even for DeepLabV3+, which generally yielded lower absolute IoU scores compared to other models but still benefited proportionally from advanced resizing (e.g., +15.37% with Resizer MFY).

In summary, for the binary semantic segmentation task on the HRF dataset, involving a significant downsampling from 3504x3504 to 512x512 pixels, the application of advanced and adaptive resizing methods led to substantial improvements in average IoU scores (derived from 5 runs per model) across all tested deep learning models when compared to the standard Bilinear interpolation. Resizer MFY and Resizer A2 emerged as the most effective methods on average, underscoring the potential benefits of optimized resizing strategies for enhancing segmentation accuracy. Even common numerical methods like Bicubic and Lanczos offered modest improvements over Bilinear, suggesting that the choice of resizer is an important consideration in the image preprocessing pipeline for semantic segmentation.

3.4.1.2. Performance Comparison for Classification

The presented results in Table 3.2 detail the performance outcomes of various image resizing methods when applied to images subsequently used for classification by seven distinct deep learning models: DenseNet, ResNeXt, ShuffleNet, GoogLeNet, MnasNet, RegNet, and EfficientNet. The rows of the table delineate the different resizing configurations, including the Bilinear resizer which serves as a baseline, the Original Resizer (often associated with the Talebi & Milanfar style), adaptive methods such as Minimal V1, Minimal V2, Resizer A, Resizer A2, Resizer SK, and Resizer MFY. The columns represent different downsampling scenarios, pairing input resolutions of 1024x1024 and 2048x2048 with output resolutions of 64x64, 128x128, 256x256, and 512x512. Each cell within this matrix contains the average F1 score achieved by the ensemble of models for that specific resizing method and resolution pair, alongside the percentage increase this F1 score represents over the Bilinear baseline. The rightmost column provides the overall average F1 score and percentage gain for each method across all tested resolution configurations.

An analysis of the overall average performance, as indicated in the final column, reveals Resizer A2 as the top-performing method, achieving an average F1 score of 0.45 and a substantial average percentage gain of +22.39% over the Bilinear baseline. Minimal V2 demonstrated nearly identical efficacy with an average F1 score of 0.45 and a +21.43% increase. Other methods delivering strong performance included Resizer A (0.43 F1;

+15.97%), Resizer SK (0.42 F1; +15.34%), and Minimal V1 (0.42 F1; +15.08%). The Original Resizer, noted for its classic approach, also outperformed the baseline with an F1 score of 0.39 and a +6.64% gain. The Bilinear resizer itself established the baseline with an average F1 score of 0.37. Notably, Resizer MFY was the only method to underperform the Bilinear resizer on average, yielding an F1 score of 0.36, which translates to a -5.84% decrease in performance.

Table 3.2. F1 metric increase and decrease percentages in terms of different input-output resolutions ((1024x1024 2048x2048)-(64x64, 128x128, 256x256, 512x512)). Each cell indicates the percentage of an average for all models (DenseNet, ResNext, ShuffleNet, GoogLeNet, MnasNet, RegNet, EfficientNet).

Resizer/Config	Metric	1024-64	2048-64	1024-128	2048-128	1024-256	2048-256	1024-512	2048-512	Average
Bilinear Resizer	F1 Avg	0.2551	0.2752	0.3249	0.3139	0.4207	0.4266	0.4929	0.4842	0.37
	% ↑	-	-	-	-	-	-	-	-	**0.00**
Original Resizer	F1 Avg	0.3145	0.3132	0.3547	0.3414	0.4391	0.4397	0.4436	0.4872	0.39
	% ↑	23.32	13.81	9.17	8.75	4.38	3.07	-10.00	0.61	**6.64**
Minimal V1	F1 Avg	0.3289	0.3309	0.4188	0.3786	0.4537	0.4680	0.4776	0.5207	0.42
	% ↑	28.94	20.23	28.89	20.61	7.84	9.71	-3.10	7.52	**15.08**
Minimal V2	F1 Avg	0.3738	0.3358	0.3983	0.4099	0.5032	0.5009	0.5231	0.5158	0.45
	% ↑	46.56	22.02	22.56	30.60	19.63	17.41	6.14	6.52	**21.43**
Resizer A	F1 Avg	0.3476	0.3163	0.3541	0.4267	0.5018	0.4400	0.5392	0.4834	0.43
	% ↑	36.28	14.94	8.96	35.94	19.28	3.14	9.39	-0.18	**15.97**
Resizer A2	F1 Avg	0.3804	0.3835	0.3806	0.4064	0.5050	0.4836	0.4893	0.5391	0.45
	% ↑	49.13	39.35	17.12	29.46	20.05	13.38	-0.72	11.33	**22.39**
Resizer SK	F1 Avg	0.3243	0.3464	0.4362	0.3882	0.4392	0.4755	0.4581	0.4987	0.42
	% ↑	27.12	25.86	34.23	23.67	4.40	11.46	-7.05	2.99	**15.34**
Resizer MFY	F1 Avg	0.2428	0.2175	0.3039	0.2962	0.4255	0.4241	0.4412	0.4897	0.36
	% ↑	-4.80	-20.97	-6.47	-5.64	1.15	-0.59	-10.49	1.12	**-5.84**

A consistent trend across most resizing methods was the general improvement in F1 scores as the output resolution increased, indicating that retaining more pixel information is generally beneficial for classification accuracy. For instance, the Bilinear resizer applied to a 1024x1024 input saw its F1 score rise from 0.2551 for a 64x64 output to 0.4929 for a 512x512 output. However, the relative percentage improvement offered by the advanced resizers over the Bilinear method was often most significant at the lower output resolutions (e.g., 64x64, 128x128), where the Bilinear method inherently struggled more.

Under conditions of substantial downsampling, such as to a 64x64 output, adaptive

methods showcased their largest advantages. For the 1024x1024 to 64x64 scenario, Resizer A2 led with a +49.13% gain (F1 = 0.3804), closely followed by Minimal V2 with a +46.56% gain (F1 = 0.3738). When downsampling from 2048x2048 to 64x64, Resizer A2 remained the strongest performer (+39.35%, F1 = 0.3835), while Resizer MFY exhibited a significant performance drop, particularly from the 2048x2048 input (-20.97%). As output resolution increased to 128x128, for 1024x1024 inputs, Resizer SK achieved the highest gain (+34.23%, F1 = 0.4362), while for 2048x1024 inputs, Resizer A performed best (+35.94%, F1 = 0.4267).

With medium output resolutions like 256x256, the percentage gains over bilinear, while still meaningful, began to diminish. For 1024x1024 inputs, Resizer A2 (+20.05%, F1 = 0.505), Minimal V2 (+19.63%, F1 = 0.5032), and Resizer A (+19.28%, F1 = 0.5018) were prominent. For 2048x2048 inputs downsampled to 256x256, Minimal V2 recorded the highest increase (+17.41%, F1 = 0.5009). At the highest tested output resolution of 512x512, when downsampling from a 1024x1024 input, only Resizer A (+9.39%, F1 = 0.5392) and Minimal V2 (+6.14%, F1 = 0.5231) provided improvements over the Bilinear baseline (F1 = 0.4929); several other advanced methods actually underperformed Bilinear in this specific scenario. However, when downsampling from a 2048x2048 input to 512x512, Resizer A2 reasserted its effectiveness with an +11.33% gain (F1 = 0.5391), and Minimal V1 and Minimal V2 also showed robust positive gains, suggesting that higher input resolutions can better leverage the capabilities of certain advanced resizers at larger output sizes.

In summary, adaptive resizer architectures generally deliver their most significant benefits under conditions of strong downsampling, particularly to output resolutions like 64x64 and 128x128, with Resizer A2 and Minimal V2 frequently leading in performance. As the target output resolution increases and the downsampling ratio becomes less extreme, the absolute gains from these advanced methods tend to decline. Indeed, for mild downsampling (e.g., 1024x1024 to 512x512), some sophisticated methods may not outperform simple Bilinear interpolation. The results also suggest that higher input resolutions (e.g., 2048x2048) can enhance the effectiveness of certain resizing methods, especially when producing larger output images. Overall, Resizer A2 emerges as a particularly robust choice across a variety of scenarios, while Resizer MFY generally failed to offer improvements over the baseline in this task.

3.4.2. Computational Overhead by Adaptive Resizing

A detailed quantitative analysis of the computational costs associated with each adaptive resizing method is presented here. We closely examine the trade-offs between performance gains and the increased resource demands in terms of FLOPs, parameters, inference time, memory usage, and total training duration. This overhead is evaluated separately for both the segmentation and classification experimental setups, offering a clear perspective on the practical deployment feasibility of each resizer architecture.

3.4.2.1. Segmentation Experiments

An evaluation of the computational overhead introduced by various adaptive image resizing methods, when integrated with five distinct semantic segmentation models (U-Net, U-Net++, DeepLabV3+, MA-Net, and LinkNet), is presented in Table 3.3. These segmentation experiments involved a consistent downsampling of input images from an original resolution of 3504x3504 pixels to an output of 512x512 pixels. The table quantifies this overhead through several metrics: the number of Parameters (M), Floating Point Operations (FLOPs, G), Inference Time (ms), Training GPU memory usage (GB), and Training Duration (Mins) for each model-resizer combination. In this comparison, the Bilinear resizer represents the most basic resizing approach, the "Original Resizer" serves as a non-proprietary adaptive baseline, and the remaining methods (Minimal V1, Minimal V2, Resizer A, Resizer A2, Resizer SK, Resizer MFY) constitute the proposed adaptive resizers under evaluation for their computational efficiency.

A consistent observation across all segmentation models and metrics is that the adoption of adaptive resizers, encompassing both the Original Resizer and all proposed methods, universally leads to an increased computational load compared to the elementary Bilinear resizer. While the additional parameters introduced by the adaptive resizers are relatively small, typically ranging from approximately 0.04% to 0.85% of the segmentation model's total parameters (e.g., for U-Net), the impact on operational complexity is far more substantial. For instance, FLOPs increased significantly, with proposed resizers elevating them by approximately 74% (Minimal V1 with U-Net) to over 730% (Resizer SK with DeepLabV3+) compared to their respective Bilinear configurations. This surge in operational demand directly translates to longer inference times. The Original Resizer, for example, increased U-Net's inference time by a substantial 214.6%, while proposed

methods extended these times by a range of approximately 20% (Minimal V1 with U-Net) to about 168% (Resizer SK with DeepLabV3+). Training GPU memory requirements also escalated; while U-Net with Bilinear reported 7.6 GB training GPU, other models like DeepLabV3+ with Bilinear were in the 2.3 GB range. Adaptive resizers increased this demand considerably, often by 100% to over 600% depending on the model and resizer (e.g., Minimal V1 increased DeepLabV3+'s training GPU usage by 221.7%, while Resizer SK increased it by 678.3%). Consequently, training durations were dramatically extended, with adaptive resizers typically causing an increase of 700% to over 1100% – essentially 8 to 12 times longer – compared to the much shorter training periods observed with bilinear resizing.

When comparing the proposed adaptive resizers against the Original Resizer, it is generally evident that the Original Resizer imposes a very high, often prohibitively so, computational cost. For example, with the U-Net model, the Original Resizer increased FLOPs by 82.4% over Bilinear, while its inference time surged by 214.6%, training GPU usage by 177.6%, and training duration by an exceptional 1162%. Many proposed resizers, while also adding overhead, presented a more favorable computational profile. For instance, MinimalV1 with U-Net increased FLOPs by a lesser 74.5% and inference time by only 20.8%. Even heavier proposed methods like Resizer SK for U-Net, which increased FLOPs by 629.6%, still had a lower inference time increase (+148.5%) than the Original Resizer. Across models, proposed resizers frequently offered substantial reductions in inference time (e.g., MinimalV1 for U-Net was approximately 61% faster in inference than Original Resizer) and training GPU usage (e.g., MinimalV1 for DeepLabV3+ used about 65% less training GPU memory than Original Resizer), leading to more manageable, albeit still increased, training durations compared to the often excessive times required by the Original Resizer.

Among the suite of proposed adaptive resizers, Minimal V1 consistently distinguishes itself as the most computationally lightweight option. This efficiency is apparent across multiple dimensions of computational cost when compared to its proposed peers. Minimal V1 generally adds the fewest parameters and, more critically, exhibits the lowest or among the lowest FLOP counts; for U-Net, its 74.5% increase in FLOPs over Bilinear was considerably less than the 290% to 630% increases seen with Resizers A, A2, SK, and MFY.

Table 3.3. A detailed breakdown of the computational overhead for each segmentation model and resizer combination. The table reports on five key metrics: the number of trainable parameters, FLOPs, GPU memory consumption for inference and training, and the total training duration in minutes.

General Context: 3504x3504 to 512x512 input/output resolution						
Model	**Resizer**	**Params (M)**	**FLOPs (G)**	**Inference GPU (ms)**	**Training GPU (GB)**	**Training Duration (Mins)**
U-Net	Bilinear Resizer	32.521	85.661	10.631	2.3	2.75
	Original Resizer	32.533	156.226	33.447	21.1	34.71
	MinimalV1	32.628	149.442	12.845	7.6	24.66
	MinimalV2	32.628	270.633	19.473	16.1	25.88
	Resizer A	32.735	334.505	21.269	18.1	24.33
	Resizer A2	32.735	334.421	21.541	18.2	24.54
	Resizer SK	32.922	624.978	26.414	18.1	25.29
	Resizer MFY	32.798	367.481	23.103	17.6	25.37
U-Net++	Bilinear Resizer	48.991	460.491	17.631	2.6	3.37
	Original Resizer	48.898	531.063	40.594	25.2	38.35
	MinimalV1	49.093	524.279	21.432	12.9	24.68
	MinimalV2	49.093	645.471	25.247	21.7	27.57
	Resizer A	49.211	709.341	28.603	22.9	28.19
	Resizer A2	49.211	709.257	28.783	21.2	24.71
	Resizer SK	49.387	999.814	32.877	23.6	27.99
	Resizer MFY	49.263	742.317	30.322	21.8	26.81
DeepLabV3+	Bilinear Resizer	26.681	73.811	9.700	2.3	1.97
	Original Resizer	26.691	144.379	32.641	21.1	32.45
	MinimalV1	26.785	137.595	12.441	7.4	23.47
	MinimalV2	26.785	258.786	18.061	15.8	23.33
	Resizer A	26.892	322.658	20.402	17.9	23.01
	Resizer A2	26.892	322.573	20.712	17.6	23.72
	Resizer SK	27.079	613.131	25.979	17.9	24.31
	Resizer MFY	26.955	355.633	22.276	17.2	23.31
MA-Net	Bilinear Resizer	147.441	149.231	13.911	2.5	4.14
	Original Resizer	147.452	219.801	35.821	21.6	50.32
	MinimalV1	147.547	213.016	16.748	10.4	39.07
	MinimalV2	147.547	334.207	20.634	17.1	37.08
	Resizer A	147.654	398.079	23.698	19.2	31.39
	Resizer A2	147.654	397.994	24.014	19.3	35.63
	Resizer SK	147.841	688.552	28.812	18.9	37.18
	Resizer MFY	147.717	431.054	25.542	17.4	35.65
LinkNet	Bilinear Resizer	31.181	86.281	10.781	2.7	2.12
	Original Resizer	31.191	156.855	32.925	20.1	35.58
	MinimalV1	31.285	150.071	13.059	8.1	25.76
	MinimalV2	31.285	271.261	18.103	15.8	25.91
	Resizer A	31.392	335.133	20.718	17.7	24.71
	Resizer A2	31.392	335.049	21.018	18.4	24.46
	Resizer SK	31.579	625.606	26.572	19.3	24.34
	Resizer MFY	31.455	368.108	22.602	17.2	27.01

This lower operational count directly translates to Minimal V1 typically offering the shortest inference times within the proposed group – for example, its 20.8% increase for U-Net was significantly less than the 83% to 148% increases from other proposed methods. Furthermore, Minimal V1 tends to be more economical in terms of training GPU usage. For the U-Net model, it uniquely reported no increase (0%) in training GPU memory over the Bilinear version's 7.6 GB, whereas other proposed methods for U-Net increased GPU usage by 111% to 139%. For DeepLabV3+, Minimal V1's 221.7% increase in training GPU usage over Bilinear was still at the lower end compared to other proposed methods like Resizer SK (+678.3%). Consequently, while its training durations represented a significant increase over Bilinear (e.g., approximately +797% for U-Net), they were often 5-10% shorter than some of the more computationally intensive proposed resizers like Resizer MFY or Resizer SK for the same model.

In conclusion, the integration of adaptive resizing techniques for the substantial downsampling (3504x3504 to 512x512 pixels) in these segmentation experiments brings a clear trade-off in the form of increased computational burden, with FLOPs, processing times, and memory usage often escalating by several hundred percent compared to the simple Bilinear approach. The non-proprietary Original Resizer was found to be particularly resource-intensive, often magnifying these overheads to extreme levels (e.g., training time increases exceeding 1000%). Within the set of proposed adaptive resizers, a clear gradient of computational costs exists. Minimal V1 consistently emerges as the most computationally parsimonious, offering increases in FLOPs and inference times that are often 2 to 5 times less than those of heavier proposed methods like Resizer SK, alongside more modest demands on GPU memory. This detailed quantitative analysis of computational overheads underscores the importance of selecting a resizing method that not only enhances segmentation performance but also aligns with available computational resources, with MinimalV1 demonstrating particular promise for scenarios where efficiency is a key practical consideration.

3.4.2.2. Classification Experiments

Table 3.4 details the computational overhead associated with various image resizing techniques when applied to the GoogLeNet classification model, specifically examining scenarios with input image sizes of 2048x2048 and 1024x1024 pixels, downscaled to target resolutions of 64x64, 128x128, 256x256, and 512x512 pixels. The evaluated metrics include Multiply-Accumulate operations (MACs, G), Floating Point Operations

(FLOPs, G), number of Parameters (M), Inference Time on both CPU and GPU (ms), and Training Duration (Min). The analysis compares the baseline bilinear resizer against the non-proprietary Original Resizer and a suite of proposed adaptive resizers: Minimal V1, Minimal V2, Resizer A, Resizer A2, Resizer SK, and Resizer MFY.

Table 3.4. Computational overheads of different resolution reduction ratios for one of the classification models (GoogLeNet).

Input Size	Target Size	Resizer	FLOPs (G)	Params (M)	Inference (ms) (CPU)	Inference (ms) (GPU)	Training Duration (min)
2048x2048	64x64	Bilinear Resizer	0.247	5.605	15.132	10.708	3.47
		Original Resizer	22.738	5.617	492.603	12.540	56.28
		Minimal V1	1.243	5.712	24.912	11.319	43.45
		Minimal V2	63.142	5.712	406.773	11.020	43.9
		Resizer A	64.140	5.819	407.070	12.665	44.01
		Resizer A2	64.138	5.819	407.915	11.059	43.67
		Resizer SK	169.092	6.006	775.578	12.731	44.04
		Resizer MFY	64.173	5.823	429.567	12.008	40.94
2048x2048	128x128	Bilinear Resizer	0.986	5.605	31.614	10.535	3.56
		Original Resizer	23.711	5.617	493.003	12.410	55.6
		Minimal V1	4.973	5.712	59.038	11.319	40.57
		Minimal V2	63.924	5.712	397.253	11.061	41.14
		Resizer A	67.916	5.819	405.380	12.075	40.81
		Resizer A2	67.911	5.819	408.088	11.706	40.63
		Resizer SK	172.045	6.006	738.165	12.741	41.4
		Resizer MFY	68.048	5.823	415.061	12.204	40.72
2048x2048	256x256	Bilinear Resizer	3.945	5.605	70.122	10.841	3.7
		Original Resizer	27.599	5.617	527.473	12.865	56.7
		Minimal V1	19.892	5.712	168.834	12.508	40.68
		Minimal V2	67.053	5.712	433.540	10.908	41.06
		Resizer A	83.021	5.819	521.023	11.435	40.75
		Resizer A2	83.000	5.819	528.613	12.002	40.48
		Resizer SK	183.858	6.006	817.451	12.755	41.09
		Resizer MFY	83.550	5.823	535.143	12.891	40.72
2048x2048	512x512	Bilinear Resizer	15.781	5.605	216.520	10.455	4.92
		Original Resizer	43.153	5.617	1085.431	16.152	61.31
		Minimal V1	79.568	5.712	559.786	11.336	40.96
		Minimal V2	79.568	5.712	586.499	11.499	43.1
		Resizer A	143.440	5.819	945.280	12.674	40.71
		Resizer A2	143.355	5.819	928.848	12.934	42.23

(cont. on next page)

Table 3.4 (cont.)

Input Size	Target Size	Resizer	FLOPs (G)	Params (M)	Inference (ms) (CPU)	Inference (ms) (GPU)	Training Duration (min)
		Resizer SK	231.109	6.006	1287.087	15.236	41.3
		Resizer MFY	145.556	5.823	1080.817	14.195	40.94
1024x1024	64x64	Bilinear Resizer	0.247	5.605	15.132	10.708	3.47
		Original Resizer	5.928	5.617	192.832	12.347	10.16
		Minimal V1	1.243	5.712	25.656	11.309	9.86
		Minimal V2	15.981	5.712	116.926	11.299	9.83
		Resizer A	16.979	5.819	119.972	12.567	9.89
		Resizer A2	16.978	5.819	112.265	12.072	9.91
		Resizer SK	43.011	6.006	224.761	12.658	10.01
		Resizer MFY	17.012	5.823	121.826	12.823	9.74
1024x1024	128x128	Bilinear Resizer	0.986	5.605	31.614	10.535	3.66
		Original Resizer	6.900	5.617	186.528	11.505	10.2
		Minimal V1	4.973	5.712	68.479	11.947	9.89
		Minimal V2	16.763	5.712	245.786	11.260	9.55
		Resizer A	20.755	5.819	166.634	13.369	9.73
		Resizer A2	20.750	5.819	166.515	12.109	9.45
		Resizer SK	45.965	6.006	266.464	13.468	10.08
		Resizer MFY	20.887	5.823	178.877	11.926	9.85
1024x1024	256x256	Bilinear Resizer	3.945	5.605	70.122	10.841	3.9
		Original Resizer	10.788	5.617	244.191	12.177	10.71
		Minimal V1	19.892	5.712	181.381	11.701	10.16
		Minimal V2	19.892	5.712	307.458	11.605	9.83
		Resizer A	35.860	5.819	302.868	11.940	9.98
		Resizer A2	35.839	5.819	324.115	12.582	9.85
		Resizer SK	57.777	6.006	406.266	13.706	10.68
		Resizer MFY	36.389	5.823	320.132	12.280	10.2
1024x1024	512x512	Bilinear Resizer	15.781	5.605	216.520	10.455	5.08
		Original Resizer	26.342	5.617	441.614	11.951	14.25
		Minimal V1	79.568	5.712	716.085	12.496	13.18
		Minimal V2	32.407	5.712	372.830	11.396	9.89
		Resizer A	96.279	5.819	909.406	12.383	15.61
		Resizer A2	96.194	5.819	931.680	12.061	15.87
		Resizer SK	105.028	6.006	974.522	13.001	16.43
		Resizer MFY	98.395	5.823	1085.608	12.256	17.36

The number of parameters for the GoogleNet model combined with each resizer remains constant across all resolution configurations, as parameters are inherent to the model and the resizer's architecture rather than the image dimensions. The bilinear resizer combination reports 5.605M parameters. The Original Resizer adds a minimal 0.012M

parameters (+0.21%). Among the proposed methods, Minimal V1 and Minimal V2 add 0.107M parameters (+1.91%), Resizers A, A2, and MFY add approximately 0.214M to 0.218M (+3.8% to +3.9%), and Resizer SK introduces the most, an additional 0.401M parameters (+7.15%) compared to the Bilinear setup. These increases are generally modest relative to the base model size.

The primary computational load from resizers is evident in MACs and FLOPs, which are heavily influenced by both the choice of resizer and the image resolutions. For the bilinear resizer, FLOPs scale predictably with the number of output pixels; for a 2048x2048 input, FLOPs increase from 0.247G for a 64x64 output to 15.781G for a 512x512 output, roughly a 4-fold increase with each doubling of output dimensions. Adaptive resizers exhibit significantly higher FLOPs than Bilinear. When downsampling a 2048x2048 input to 64x64, the FLOPs for Minimal V1 (1.243G) represent a 403% increase over Bilinear, while more complex proposed methods like Resizer SK (169.092G) show an exceedingly large percentage increase (over 68000%) due to Bilinear's very low baseline; the Original Resizer also shows a massive relative increase (+9106%). As output resolution increases, the absolute FLOPs for adaptive resizers also increase, often scaling with output pixel count (e.g., Minimal V1 FLOPs roughly quadruple with each output dimension doubling for a 2048x2048 input). However, their percentage increase over Bilinear can be more moderate at higher output resolutions where Bilinear's own FLOPs are higher; for instance, when resizing 2048x2048 to 512x512, Original Resizer's FLOPs (43.153G) are +173% over Bilinear, and MinimalV1 (79.568G) is +404% over Bilinear. An interesting observation for the 2048x2048 to 512x512 scenario is that Minimal V1 and Minimal V2 report identical MACs and FLOPs, a convergence not seen at lower output resolutions where Minimal V2 is significantly heavier. The input resolution also plays a critical role: processing a 2048x2048 input to 64x64 with Minimal V1 requires 1.243G FLOPs, while a 1024x1024 input to the same 64x64 output with Minimal V1 uses the same 1.243G FLOPs, suggesting the FLOPs for some resizers might be more dependent on the resizer's fixed operations or target resolution aspects rather than total input pixels once a certain threshold or operational mode is met, though this is not consistent across all resizers or metrics (e.g., Original Resizer FLOPs are much higher for 2048→64 than 1024→64). More typically, higher input resolutions lead to higher FLOPs for a fixed output; for Original Resizer targeting 64x64, FLOPs are 22.738G from 2048x2048 input versus 5.928G from 1024x1024.

Inference times reflect the trends in MACs/FLOPs, with CPU inference times showing greater sensitivity to computational load than GPU times. For the 2048x2048 to

64x64 downsampling, Bilinear CPU inference is 15.132 ms. Minimal V1 increases this by 64.6% to 24.912 ms, while Resizer SK dramatically slows it to 775.578 ms (a 50-fold increase). The Original Resizer also incurs substantial CPU inference overhead (e.g., 492.603 ms for 2048→64, a 3155% increase). On the GPU, the increases are far more contained: Bilinear takes 10.708 ms for 2048→64, Minimal V1 takes 11.319 ms (+5.7%), and even Resizer SK only increases GPU time to 12.731 ms (+18.9%). This highlights the GPU's efficiency in parallelizing image operations. As output resolution increases, both CPU and GPU inference times for all resizers tend to rise. For example, with a 2048x2048 input and Minimal V1, CPU inference time goes from 24.912 ms (64x64 output) to 559.786 ms (512x512 output), while GPU time increases from 11.319 ms to 11.336 ms (interestingly, GPU time for Minimal V1 remains very stable across output resolutions for this high input size). The Original Resizer shows very high CPU and significant GPU time increases across all settings. Comparing input sizes, using a 2048x2048 input generally results in longer CPU inference times than a 1024x1024 input for the same resizer and output resolution (e.g., Original Resizer 2048→64 takes 492.603 ms on CPU vs. 192.832 ms for 1024→64). GPU inference times are less affected by input size for some resizers like Minimal V1, but show increases for others like Original Resizer.

Training durations are significantly impacted by the choice of resizer and resolution, consistently being much longer for adaptive resizers than for Bilinear. For a 2048x2048 input downsampled to 64x64, Bilinear training takes 3.47 minutes. Minimal V1 extends this to 43.45 minutes (an increase of over 1150%), while Resizer SK takes 44.04 minutes (+1169%), and Original Resizer takes 56.28 minutes (+1522%). Generally, as output resolution increases, training duration for Bilinear configurations sees a modest rise (e.g., for 2048x2048 input, from 3.47 mins for 64x64 output to 4.92 mins for 512x512 output). Adaptive resizers, however, often exhibit less consistent scaling in training duration with output resolution, sometimes showing slight decreases or fluctuations, though they consistently remain much higher than Bilinear. For instance, with a 2048x2048 input, Minimal V1's training time varies from 43.45 mins (64x64) down to 40.96 mins (512x512). The choice of input resolution also affects training time; processing a 2048x2048 input generally leads to longer training durations than a 1024x1024 input for the same adaptive resizer and output (e.g., Original Resizer 2048→64 takes 56.28 mins vs. 10.16 mins for 1024→64).

Across the various resolutions and computational metrics evaluated for GoogleNet model, Minimal V1 frequently presents as the most computationally lightweight among the proposed adaptive resizers. While it consistently introduces more overhead than

bilinear resizing, for example, FLOPs increases of 400-500% over bilinear were common for higher output resolutions, and CPU inference times could be 2-3 times longer for smaller outputs from 1024x1024 inputs, its impact is generally much lower than other proposed methods like Minimal V2 (at lower resolutions), Resizers A/A2, and particularly Resizer SK and MFY, which often show substantially higher MACs/FLOPs and CPU inference times. For instance, when downsampling 2048x2048 to 256x256, Minimal V1 added 19.892G FLOPs (+404% over Bilinear), whereas Resizer SK added 183.858G FLOPs (+4560% over bilinear). The Original Resizer generally imposed a very high computational penalty, often exceeding that of most proposed methods, especially in CPU inference times and training duration. The efficiency of Minimal V1, particularly in GPU inference times which showed remarkable stability across output resolutions for a fixed large input, makes it a noteworthy candidate when balancing performance with computational cost.

In conclusion, Table 3.4 underscores that while adaptive resizers can potentially enhance classification accuracy, they introduce varying degrees of computational overhead compared to Bilinear resizing when applied to the GoogLeNet model across different resolution scaling tasks. This overhead is most pronounced in terms of FLOPs, CPU inference times, and training durations, often resulting in increases of several hundred to several thousand percent over Bilinear, particularly when Bilinear's baseline cost is very low (e.g., at small output resolutions). The impact on GPU inference times is generally more mitigated due to parallel processing capabilities. Higher input and output resolutions tend to increase computational demands for most methods. Among the proposed adaptive techniques, Minimal V1 consistently demonstrates a more favorable computational profile, offering a more lightweight alternative to other proposed resizers and the often very costly Original Resizer. These findings are critical for selecting an appropriate resizer, requiring a careful consideration of the trade-offs between potential accuracy gains and the significant implications for computational resources and processing times.

3.5. Discussion

The advent of adaptive image resizing marked a significant step forward from traditional static interpolation methods, with the Original Resizer, based on the work of Talebi & Milanfar, being a notable early architecture. While pioneering, this initial architecture is not without its drawbacks, exhibiting limitations in robustness and often falling short of optimal performance. This is primarily due to its reliance on bilinear re-

sizing after significant convolutional processing in one of its branches and the substantial computational overhead it introduces. Recognizing these gaps, this study introduced and comprehensively evaluated a suite of novel adaptive resizer architectures, Minimal V1, Minimal V2, Resizer A, Resizer A2, Resizer SK, and Resizer MFY, designed to offer improved performance and/or computational efficiency.

3.5.1. Architectural Impact on Task Performance

The effectiveness of the proposed resizers varied between segmentation and classification tasks, underscoring how architectural choices influence the preservation and learning of task-specific features.

3.5.1.1. Segmentation: Preserving Fine Details

For the retinal vessel segmentation task on the HRF dataset, the proposed adaptive resizers demonstrated a strong ability to preserve crucial image features necessary for delineating fine vascular structures, leading to substantial performance gains over traditional methods. Figure 3.10 provides a visual comparison of how different resizer architectures transform the input images from the HRF dataset. For instance, Resizer MFY's output might visually emphasize edge information due to its filter bank, while Minimal V1 or V2 might produce outputs that balance feature extraction with the downsampling process differently. These visual transformations directly impact how the subsequent segmentation model perceives and processes the image.

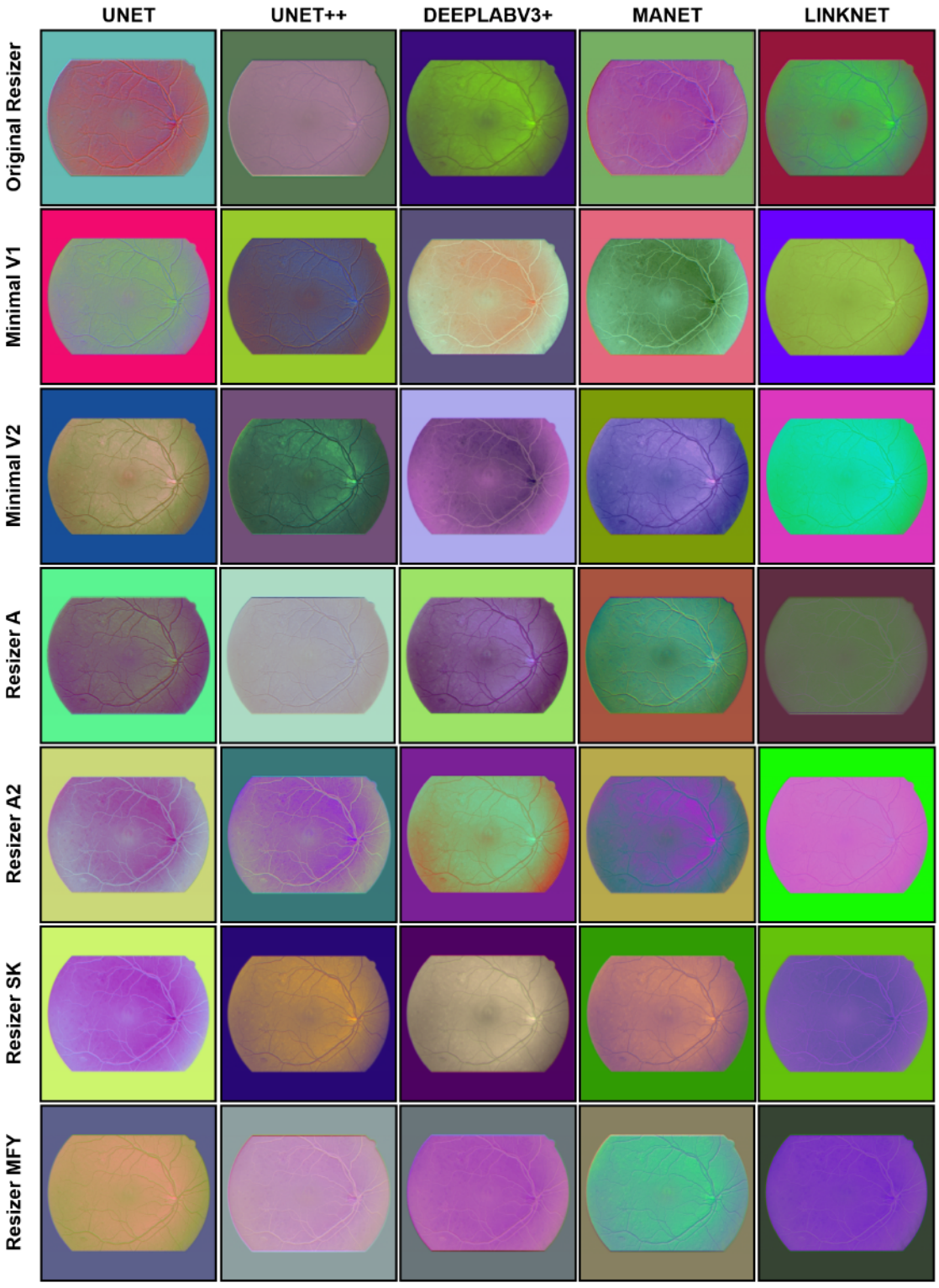

Figure 3.10. Visual comparison of the transformations produced by each resizer architecture on a sample image from the HRF dataset. Each row corresponds to a different resizer, while the columns indicate the subsequent segmentation model these images are fed into. The noticeable variations in color, contrast, and edge emphasis across the rows demonstrate how each resizing method uniquely prepares the image data before the segmentation task.

The superior performance of Resizer MFY in segmentation can be attributed to its unique branch incorporating manually designed filters (e.g., Sobel, Prewitt, Laplacian) explicitly aimed at edge detection and feature enhancement. This architectural feature appears highly beneficial for tasks requiring precise boundary delineation, such as vessel segmentation. Architectures like Minimal V2 and Resizer A2 (which builds upon Minimal V1 and V2) perform initial convolutional operations on the higher-resolution input before full downsampling via bilinear interpolation in parts of their pipeline. This strategy likely preserves more high-frequency information vital for pixel-wise segmentation. Figure 3.11 displays the training history plots for the LinkNet model when paired with different resizers. These plots can reveal differences in convergence speed or stability; for example, a resizer that provides richer, more relevant features might lead to faster convergence or a higher final validation IoU for the LinkNet model, reflecting a more effective learning process facilitated by that resizer's specific architectural design.

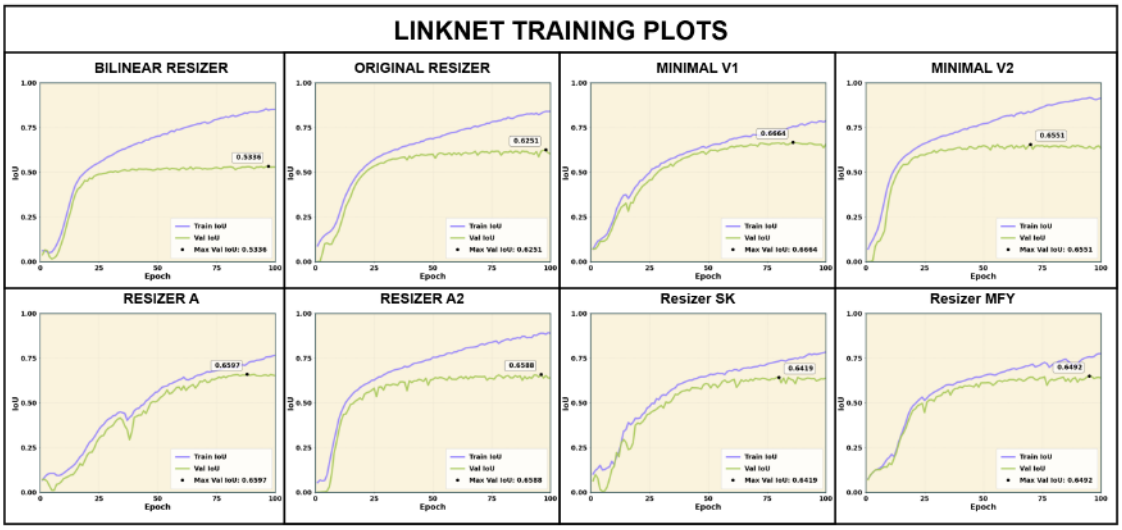

Figure 3.11. Training and validation history for the LinkNet model when paired with eight different resizer architectures. Each subplot displays the Intersection over Union (IoU) on both the training set (blue) and validation set (green) across 100 epochs. These curves allow for a direct comparison of convergence speed and stability, with the peak validation IoU score marked for each configuration.

Minimal V1, though simpler, leverages strided convolutions for adaptive resizing after an initial bilinear step, proving more effective than bilinear resizing alone or the Original Resizer. Resizer SK enhances Minimal V2 by integrating a Selective Kernel

(SK) attention mechanism. This allows the model to adaptively select kernel sizes and focus on relevant spatial and channel information, potentially improving the capture of intricate details in structures like retinal vessels. The consistent improvements across diverse segmentation models underscore the generalizability of these advanced resizing approaches for detail-oriented tasks.

3.5.1.2. Classification: Learning Discriminative Patterns

In the diabetic retinopathy classification task on the IDRiD dataset, Resizer A2 and Minimal V2 emerged as top performers. Figure 3.12 illustrates the resizer outputs for the IDRiD dataset across various input-output resolutions. These images show how different architectures handle the downsampling for classification. For instance, some resizers might smooth out finer details that are less relevant for classification while preserving broader textural information or lesion characteristics, which could explain their better performance in this task compared to segmentation-focused resizers. The visual outputs can hint at why certain resizers allow the classification models to learn more discriminative patterns.

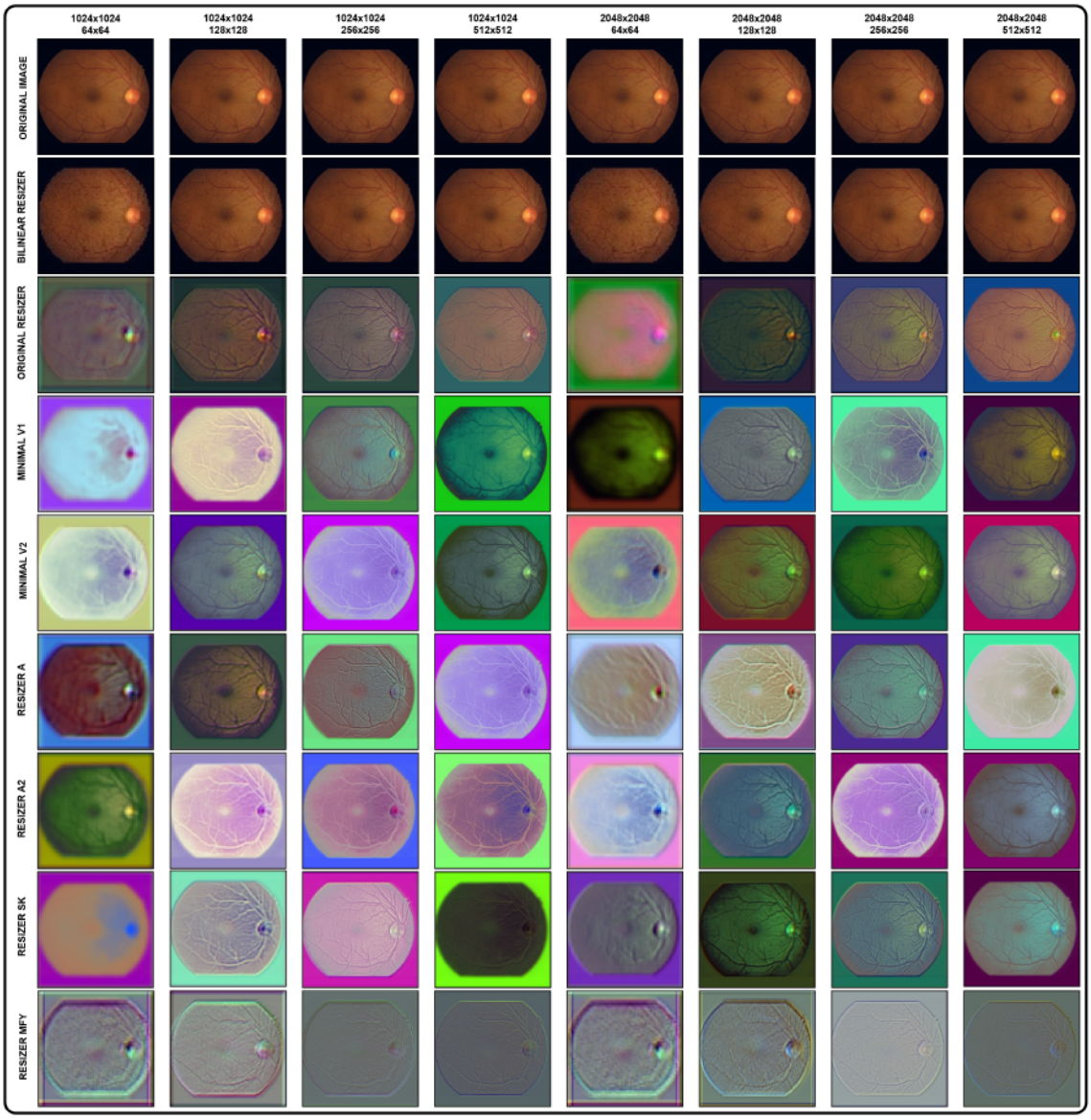

Figure 3.12. A comprehensive visual comparison of resizer outputs on the IDRiD dataset across a matrix of input and output resolutions. Each row displays the output from a specific resizing architecture, with the Bilinear Resizer serving as the baseline. The columns show the results for two different input sizes (1024x1024 and 2048x2048) downsampled to four different target sizes, illustrating how each resizer performs under varying degrees of compression.

The success of Resizer A2 and Minimal V2 in classification, particularly at aggressive downsampling rates, stems from their learned ability to retain task-relevant features. Minimal V2 applies convolutions before bilinear resizing, enabling it to encode important information from the higher-resolution input into feature maps prior to the final resizing step. Resizer A2, combining aspects of Minimal V1 and Minimal V2, suggests that a balanced approach to feature preservation and adaptive resizing is effective for classification. The removal of intermediate projection convolutions in Resizer A2 (compared to Resizer A), without significant performance degradation, indicates that extensive operations after reaching the target resolution might not always be beneficial for generalization in classification tasks.

Conversely, Resizer MFY, which excelled in segmentation, underperformed in classification. Its manually engineered filters, while adept at enhancing edges for segmentation, might not generalize well for classification, potentially hindering the learning of more abstract textural and contextual patterns crucial for disease grading. This highlights the task-specific nature of optimal resizing strategies. The largest performance increases for classification were often seen with significant downsampling (e.g., to 64x64 or 128x128 outputs), where adaptive resizers are particularly effective at preserving discriminative information that traditional methods would lose.

3.5.2. Architectural Design and Computational Cost

While adaptive resizers offer clear performance advantages, these benefits are universally accompanied by increased computational costs compared to simple bilinear interpolation.

The Original Resizer was found to be particularly resource-intensive, often imposing the highest computational burden in terms of FLOPs, inference time, GPU memory usage, and especially training duration. This is largely due to its initial large-kernel convolutional operations on the full high-resolution input in one of its branches.

Among the proposed architectures, Minimal V1 consistently stood out as the most computationally lightweight. Its efficiency stems from an architectural design that often involves an initial bilinear interpolation to an intermediate size (four times the target resolution) followed by a limited number of strided convolutional layers. This approach keeps the bulk of the convolutional computation on an already somewhat reduced tensor, although significant initial bilinear downscaling can risk information loss if the input is vastly larger than this intermediate size.

Minimal V2, which performs strided convolutions first before a final bilinear resizing step, generally incurs higher computational costs than Minimal V1, especially with large input resolutions, as its initial convolutions operate on larger feature maps. This can make Minimal V2 more memory-intensive in such scenarios. Resizer A and Resizer A2, being dual-branch architectures that draw principles from both Minimal V1 and Minimal V2, naturally exhibit higher complexity than Minimal V1 alone. Architectures like Resizer SK, with its additional SK attention mechanism and three branches, and Resizer MFY, with its bank of manual filters in a dedicated branch, are inherently more complex and thus computationally heavier than the simpler Minimal variants.

In summary, the proposed adaptive resizers, particularly architectures like Resizer MFY for segmentation and Resizer A2 for classification, demonstrate substantial performance improvements. However, these gains necessitate a trade-off with computational resources. The Original Resizer often represents an extreme in this cost. Among the novel architectures, Minimal V1 consistently emerges as the most computationally frugal, offering a viable balance when processing resources are constrained, even if its peak performance may sometimes be slightly lower than more complex proposed resizers. The selection of an optimal resizer is therefore a critical decision, hinging on a careful balance between the desired task accuracy, the specific nature of the visual task (detail-oriented vs. pattern-oriented), and the available computational budget.

3.6. Limitations of the Approach

It is important to remember that the proposed adaptive resizer architectures have shown big increases in performance, but they also have some limits. The main advantage of these resizers is that they may keep fine-grained details that are often lost when you downsample numbers normally. Because of this, their efficiency depends a lot on the type of dataset. If a dataset doesn't have any important, high-frequency information, the adaptive resizer may not work any better than simpler methods like bilinear or lanczos interpolation. The classification findings showed that some adaptive architectures did not fare better than the bilinear baseline when the downsampling ratios were lower (e.g., 1024x1024 to 512x512). This means that they are not useful in all situations.

The most important drawback is the balance between performance and cost. The results show that all adaptive resizers add a lot of additional overhead in terms of FLOPs, inference time, and memory use compared to standard approaches. This increase, which can make training periods more than 700% longer, makes the present architectures less

appropriate for applications where resources are very limited, including real-time processing or deployment on edge devices with limited VRAM and computing power. In fields where precision is more important than cost, though, this trade-off makes sense. In industries like medical analysis, where getting the most accurate diagnosis is the most important thing, these adaptive resizers' efficiency improvements can be quite important and may be worth the extra resources they need.

3.7. Conclusion

This study addressed the critical challenge of information loss during the downsampling of high-resolution medical images for deep learning applications. We introduced and comprehensively evaluated a suite of novel adaptive resizer architectures against traditional numerical methods and an established adaptive resizing technique. The investigation focused on two demanding medical imaging tasks: retinal vessel segmentation using the HRF dataset and diabetic retinopathy grading on the IDRiD dataset.

Our findings demonstrate that the proposed adaptive resizer architectures can significantly enhance model performance. For the retinal vessel segmentation task, Resizer MFY emerged as the most effective, yielding an average Intersection over Union (IoU) increase of +21.04% compared to the standard bilinear interpolation. In the diabetic retinopathy classification task, Resizer A2 proved to be the most successful, achieving an average F1 score improvement of +22.39% over the bilinear baseline. These results highlight the substantial benefits of employing adaptive resizers tailored to preserve task-relevant features, particularly when significant downsampling is required. The study also underscored that the optimal adaptive resizer can be task-dependent, as evidenced by Resizer MFY's differing performance between segmentation and classification.

While the proposed adaptive resizers offer considerable accuracy improvements, they also introduce additional computational overhead. The Original Resizer was found to be particularly resource-intensive. Among our novel architectures, Minimal V1 consistently presented the most favorable computational profile, offering a significant boost in performance over bilinear resizing with a comparatively lower increase in FLOPs, inference time, and training duration. This makes Minimal V1 a compelling option when computational resources are a primary constraint.

In conclusion, this research successfully demonstrates the potential of novel adaptive resizer architectures to significantly improve the accuracy of deep learning models in medical image analysis. The developed resizers, particularly Resizer MFY for seg-

mentation and Resizer A2 for classification, establish new benchmarks for performance. Furthermore, the identification of Minimal V1 as a computationally efficient yet effective adaptive resizer provides a practical solution for resource-constrained environments. This work underscores the importance of the resizing strategy as a critical component in the medical image analysis pipeline, paving the way for more accurate and efficient diagnostic tools. Future work could explore further refinements of these architectures and their applicability to a broader range of medical imaging modalities and tasks.

CHAPTER 4

CONCLUSION AND FUTURE PERSPECTIVES

In conclusion, this thesis addressed the critical challenge of adaptive image downsampling for deep learning tasks, particularly within the domain of medical image analysis. The core principle explored is the integration of an adaptive resizer module placed before a deep neural network, with both the resizer and the primary model being trained jointly. The objective of this approach is to transform input images in a task-based manner, conditioning them into a format that the deep learning model can more effectively interpret, thereby enhancing overall performance.

The research commenced with an initial study (Chapter 2) to ascertain the viability and effectiveness of the "Original Adaptive Resizer," proposed by Talebi and Milanfar in 2021, for semantic segmentation tasks. This investigation utilized the CRAG dataset for gland segmentation, employing three U-Net based architectures (U-Net, Attention U-Net, and U-Net 3+) to compare the adaptive resizer against the conventional bilinear resizing method. The findings of this first study were significant: it was observed that the adaptive image resizer could indeed be beneficially applied to segmentation tasks, yielding superior results compared to the numerical bilinear resizing approach. Specifically, the integration of the Original Adaptive Resizer led to notable improvements in the Intersection over Union (IoU) metric, with average gains of approximately +6.47% for U-Net, +6.67% for Attention U-Net, and +3.79% for U-Net 3+ over their bilinear counterparts when averaged across various backbones. However, these performance enhancements came with an increased computational burden. While the parameter count of the segmentation models increased by a mere 12,000 parameters due to the resizer, the training duration approximately doubled. The average total FLOPs for the U-Net 3+ models saw an increase of about 3.0% (from 433.55 billion to 446.58 billion), and the average GPU inference time (NVIDIA A100) increased by 7.2% (from 181.41 ms to 194.41 ms).

Building upon these initial insights and aiming to surpass the capabilities of the Original Adaptive Resizer, the second phase of this research (Chapter 3) focused on proposing and evaluating a suite of novel, more robust adaptive resizer architectures. These new architectures were rigorously tested on both segmentation and classification tasks using high-resolution medical image datasets—the High-Resolution Fundus (HRF) dataset for retinal vessel segmentation and the Indian Diabetic Retinopathy Image Dataset

(IDRiD) for diabetic retinopathy classification. The evaluation meticulously considered both performance metrics and computational overhead. After experimenting with numerous module designs (over 60), six architectures were proposed as the most effective: Minimal V1, Minimal V2, Resizer A, Resizer A2, Resizer SK, and Resizer MFY. The experiments were also expanded to encompass various input and output resolutions to determine the conditions under which these adaptive resizing modules yield the most significant benefits.

The results from Chapter 3 demonstrated that the proposed adaptive resizers, particularly in scenarios with high reduction ratios, are more adept at preserving critical information. For the segmentation task (HRF dataset, downsampling from 3504 $\times$ 3504 to 512 $\times$ 512 pixels), 'Resizer MFY' yielded the highest average IoU increase of +21.04% over bilinear interpolation across five different segmentation models. For the five-class classification task (IDRiD dataset), 'Resizer A2' proved most effective, achieving an average F1 score increase of +22.39% over bilinear interpolation across seven classification models and various resolution settings.

Crucially, in terms of computational efficiency, the 'Minimal V1' architecture consistently emerged as the most lightweight among the proposed adaptive resizers. It offered substantial performance improvements with significantly lower computational overhead compared to the Original Resizer. For instance, in segmentation experiments with U-Net, Minimal V1 increased FLOPs by only +74.5% and GPU inference time by +20.8% over bilinear, whereas the Original Resizer increased FLOPs by +82.4% and GPU inference time by +214.6% (comparing Bilinear Resizer U-Net: 85.6G FLOPs, 10.6ms; Original Resizer U-Net: 156.2G FLOPs, 33.4ms; MinimalV1 U-Net: 149.4G FLOPs, 12.8ms from Table 3.3). Similarly, for classification tasks with GoogLeNet (2048 $\times$ 2048 input to 64 $\times$ 64 output), Minimal V1 increased FLOPs by +403% (1.243G vs 0.247G for Bilinear) and CPU inference time by +64.6% (24.912ms vs 15.132ms for Bilinear) over bilinear, substantially less than the Original Resizer's +9106% FLOPs (22.738G) and +3155% CPU inference time (492.603ms) increase. Furthermore, more complex proposed architectures like 'Resizer A2' and 'Resizer MFY', while achieving top-tier performance, also demonstrated advantages over the Original Resizer in certain aspects of computational load. For example, when combined with U-Net for segmentation, both Resizer A2 and Resizer MFY had higher FLOP counts (A2: 334.4G, MFY: 367.4G) than the Original Resizer (156.2G), but they exhibited lower GPU inference times (A2: 21.5 ms, MFY: 23.1 ms, vs. Original: 33.4 ms), lower training GPU memory usage (A2: 18.2 GB, MFY: 17.6 GB, vs. Original: 21.1 GB), and shorter training durations (A2: 24.54

mins, MFY: 25.37 mins, vs. Original: 34.71 mins).

A possible future direction is to include frequency-domain analysis directly in the resizer structures. The convolutional methods in this thesis learn feature hierarchies without meaning to. However, if the resizer used transformations like Wavelet or Fourier transforms, it could learn a better way to deal with high-frequency details, which are often the first things to go when you downsample the image. This would let the model choose which frequency bands to keep or get rid of based on how important they are to the diagnostic task. In addition, future architectures could go from static designs to dynamic, content-aware models. A resizer could use a gating mechanism or a mixture-of-experts (MoE) framework to choose multiple ways to downsample or compute dependent on how complicated or statistically interesting the input image is. This would let us make a more flexible trade-off, using a more resource-intensive method for preserving features only on difficult photos and a lighter method for easier ones. This would optimize the performance-cost balance for each instance.

This work's laborious design and evaluation of more than 60 architectures shows that we need more automated ways to find new designs. Neural Architecture Search (NAS) could be used in future research to systematically look at the huge design space of learnable resizers. By establishing a search space based on the effective components identified in this thesis (e.g., strided convolutions, multi-branch designs, attention mechanisms) and employing a multi-objective optimization goal that reconciles task performance (IoU or F1-score) with computational limitations (e.g., latency, FLOPs), NAS could reveal innovative, Pareto-optimal resizer architectures customized for particular hardware or clinical deployment contexts. Lastly, making these resizers easier to understand is an important next step. This work employed Grad-CAM; however, the creation of more specialized visualization approaches could clarify the specific semantic or textural elements that the resizer learns to retain. Understanding this taught "visual priority" could help doctors a lot, make them trust the models more, and help them come up with even better ways to resize images.

Reflecting on the broader applicability and architectural lessons from this work, several key insights emerge. First, the efficacy of adaptive resizing is highly dataset-dependent. For datasets lacking the fine, high-frequency features characteristic of medical and histopathological images, the performance benefits of these advanced resizers diminish, and they may not offer a significant advantage over simpler methods like bilinear interpolation. Therefore, the choice to employ an adaptive resizer should be preceded by an analysis of the dataset's feature complexity. Second, the architectural experiments

underscored a crucial design principle: for resizing to be truly "adaptive," the downsampling process itself must be learnable. Architectures that applied strided convolutions to simultaneously reduce spatial dimensions and learn resizing parameters proved more effective at preserving information than those relying on standard convolutions after a fixed resizing step. This indicates that integrating learnability directly into the size-reduction operation is paramount. Finally, while vision attention mechanisms—including channel, spatial, and self-attention—are powerful in larger networks, their integration into these compact resizer models did not yield significant performance gains. For the specific task of drastically downsampling high-resolution images, the computational overhead of these attention mechanisms outweighed their marginal benefits, suggesting that lightweight, well-designed convolutional structures are more efficient and effective for this purpose.

In summary, this thesis has successfully introduced and validated a series of novel adaptive resizer architectures that not only enhance the performance of deep learning models in critical medical image analysis tasks but also offer improved computational characteristics compared to previously established adaptive methods. The findings highlight that tailored adaptive resizers can play a pivotal role in preserving vital information during downsampling, leading to more accurate and reliable outcomes. The development of efficient architectures like 'Minimal V1' provides a practical pathway for leveraging these benefits even in resource-constrained environments. This work underscores the significance of the image resizing step within the deep learning pipeline and contributes valuable tools and insights for the continued advancement of medical image computing. Future research could focus on further refining these architectures, exploring their applicability to a wider array of imaging modalities and tasks, and investigating dynamic selection mechanisms for optimal resizer configuration based on input data characteristics.

REFERENCES

Awan, R., K. Sirinukunwattana, D. Epstein, S. Jefferyes, U. Qidwai, Z. Aftab, I. Mujeeb, D. Snead, and N. Rajpoot. 2017. "Glandular morphometrics for objective grading of colorectal adenocarcinoma histology images." *Scientific reports 7*(1): 16852.

Badrinarayanan, V., A. Kendall, and R. Cipolla. 2017. "Segnet: A deep convolutional encoder-decoder architecture for image segmentation." *IEEE transactions on pattern analysis and machine intelligence 39*(12): 2481–2495.

Bakhtiarnia, A., Q. Zhang, and A. Iosifidis. 2024. "Efficient high-resolution deep learning: A survey." *ACM Computing Surveys 56*(7): 181:1–181:37.

Budai, A., T. Kocka, R. Bock, A. Maier, J. Hornegger, and G. Michelson. 2013. "Robust vessel segmentation in fundus images." *International Journal of Biomedical Imaging 2013*: 154860.

Chaurasia, A. and E. Culurciello. 2017. "LinkNet: Exploiting encoder representations for efficient semantic segmentation." *2017 IEEE Visual Communications and Image Processing (VCIP)*: 1–4.

Chen, J., Y. Lu, Q. Yu, X. Luo, E. Adeli, Y. Wang, L. Lu, A. L. Yuille, and Y. Zhou. 2021. "Transunet: Transformers make strong encoders for medical image segmentation." *arXiv preprint arXiv:2102.04306*.

Chen, L.-C., G. Papandreou, F. Schroff, and H. Adam. 2017. "Rethinking atrous convolution for semantic image segmentation." *arXiv preprint arXiv:1706.05587*.

Chen, L.-C., Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. 2018. "Encoder-decoder with atrous separable convolution for semantic image segmentation." *Proceedings of the European Conference on Computer Vision (ECCV)*: 801–818.

Duzyel, O., M. S. Catal, C. E. Kayan, A. Sevinc, and A. Gumus. 2023. "Adaptive resizer-based transfer learning framework for the diagnosis of breast cancer using histopathology images." *Signal, Image and Video Processing 17*(8): 4561–4570.

Elharrouss, O., S. Al-Maadeed, N. Subramanian, N. Ottakath, N. Almaadeed, and Y. Himeur. 2021. "Panoptic segmentation: A review." *arXiv preprint arXiv:2111.10250*.

Graham, S., H. Chen, J. Gamper, Q. Dou, P.-A. Heng, D. Snead, Y. W. Tsang, and N. Rajpoot. 2019. "Mild-net: Minimal information loss dilated network for gland instance segmentation in colon histology images." *Medical image analysis 52*: 199–211.

Gu, W., S. Bai, and L. Kong. 2022. "A review on 2d instance segmentation based on deep neural networks." *Image and Vision Computing 120*: 104401.

Guo, M.-H., T.-X. Xu, J.-J. Liu, Z.-N. Liu, P.-T. Jiang, T.-J. Mu, S.-H. Zhang, R. R. Martin, M.-M. Cheng, and S.-M. Hu. 2022. "Attention mechanisms in computer vision: A survey." *Computational visual media 8*(3): 331–368.

Han, X. and Y. Chen. 2021. "Covid-19 classification using ct scan images with resize-mobilenet." *2021 International Conference on Intelligent Computing, Automation and Systems (ICICAS)*: 286–289.

He, K., X. Zhang, S. Ren, and J. Sun. 2016. "Deep residual learning for image recognition." *Proceedings of the IEEE conference on computer vision and pattern recognition*: 770–778.

Hesse, R., S. Schaub-Meyer, and S. Roth. 2023. "Content-adaptive downsampling in convolutional neural networks." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*: 4305–4314.

Huang, G., Z. Liu, L. van der Maaten, and K. Q. Weinberger. 2017. "Densely connected convolutional networks." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*: 4700–4708.

Huang, H., L. Lin, R. Tong, H. Hu, Q. Zhang, Y. Iwamoto, X. Han, Y.-W. Chen, and J. Wu. 2020. "Unet 3+: A full-scale connected unet for medical image segmentation." *ICASSP 2020-2020 IEEE international conference on acoustics, speech and signal processing (ICASSP)*: 1055–1059.

Jin, C., R. Tanno, T. Mertzanidou, E. Panagiotaki, and D. C. Alexander. 2022. "Learning to downsample for segmentation of ultra-high resolution images." *International Conference on Learning Representations (ICLR)*.

Khan, M. B., M. Ahmad, S. B. Yaakob, R. Shahrior, M. A. Rashid, and H. Higa. 2023. "Automated diagnosis of diabetic retinopathy using deep learning: On the search of segmented retinal blood vessel images for better performance." *Bioengineering 10*(4): 413.

Lateef, F. and Y. Ruichek. 2019. "Survey on semantic segmentation using deep learning techniques." *Neurocomputing 338*: 321–348.

Li, H., L. Li, Y. Huang, N. Li, and Y. Zhang. 2023. "An adaptive plug-and-play network for few-shot learning." *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*: 1–5.

Li, R., S. Zheng, C. Zhang, C. Duan, J. Li, and X. Lu. 2020. "MA-Net: A multi-scale attention network for remote sensing image segmentation." *IEEE Transactions on Image Processing 29*: 9372–9384.

Ma, N., X. Zhang, H.-T. Zheng, and J. Sun. 2018. "ShuffleNet V2: Practical guidelines for efficient CNN architecture design." *Proceedings of the European Conference on Computer Vision (ECCV)*: 116–131.

Minaee, S., Y. Boykov, F. Porikli, A. Plaza, N. Kehtarnavaz, and D. Terzopoulos. 2021. "Image segmentation using deep learning: A survey." *IEEE transactions on pattern analysis and machine intelligence 44*(7): 3523–3542.

Mo, Y., Y. Wu, X. Yang, F. Liu, and Y. Liao. 2022. "Review the state-of-the-art technologies of semantic segmentation based on deep learning." *Neurocomputing 493*: 626–646.

Oktay, O., J. Schlemper, L. L. Folgoc, M. Lee, M. Heinrich, K. Misawa, K. Mori, S. McDonagh, N. Y. Hammerla, B. Kainz, et al. 2018. "Attention u-net: Learning where to look for the pancreas." *arXiv preprint arXiv:1804.03999*.

Paul, S. 2021. "Learning to resize in computer vision." *Keras Documentation*. Retrieved

from Keras Documentation.

Porwal, P., S. Pachade, R. Kamble, M. Kokare, G. Deshmukh, V. Sahasrabuddhe, and F. Meriaudeau. 2018. "Indian diabetic retinopathy image dataset (idrid): A database for diabetic retinopathy screening research." *Data 3*(3): 25.

Qin, X., Z. Zhang, C. Huang, M. Dehghan, O. R. Zaiane, and M. Jagersand. 2020. "U2-net: Going deeper with nested u-structure for salient object detection." *Pattern recognition 106*: 107404.

Qiu, Q., M. Wang, J. Guo, Z. Liu, and Q. Wang. 2022. "An adaptive down-sampling method of laser scan data for scan-to-bim." *Automation in Construction 135*: 104135.

Radosavovic, I., R. P. Kosaraju, R. Girshick, K. He, and P. Dollár. 2020. "Designing network design spaces." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*: 10428–10436.

Rahman, M. 2023. "Empirical analysis of learnable image resizer for large-scale medical classification and segmentation." *Master's thesis*, University of Dayton.

Rastogi, P., K. Khanna, and V. Singh. 2022. "Gland segmentation in colorectal cancer histopathological images using u-net inspired convolutional network." *Neural Computing and Applications 34*(7): 5383–5395.

Ronneberger, O., P. Fischer, and T. Brox. 2015. "U-net: Convolutional networks for biomedical image segmentation." *Medical image computing and computer-assisted intervention–MICCAI 2015: 18th international conference, Munich, Germany, October 5-9, 2015, proceedings, part III 18*: 234–241.

Sha, Y. 2021. "Keras-unet-collection: The tensorflow, keras implementation of u-net, v-net, u-net++, unet 3+, attention u-net, r2u-net, resunet-a, u^2-net, transunet, and swin-unet with optional imagenet-trained backbones." `https://github.com/yingkaisha/keras-unet-collection`.

Simonyan, K. and A. Zisserman. 2014. "Very deep convolutional networks for large-scale image recognition." *arXiv preprint arXiv:1409.1556*.

Stergiou, A. and R. Poppe. 2023. "AdaPool: Exponential adaptive pooling for information-retaining downsampling." *IEEE Transactions on Image Processing 32*: 251–264.

Szegedy, C., W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. 2015. "Going deeper with convolutions." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*: 1–9.

Talebi, H. and P. Milanfar. 2021. "Learning to resize images for computer vision tasks." *Proceedings of the IEEE/CVF international conference on computer vision*: 497–506.

Tan, M., B. Chen, R. Pang, V. Vasudevan, M. Sandler, A. Howard, and Q. V. Le. 2019. "MnasNet: Platform-aware neural architecture search for mobile." *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*: 2820–2828.

Tan, M. and Q. V. Le. 2019. "EfficientNet: Rethinking model scaling for convolutional neural networks." *Proceedings of the 36th International Conference on Machine Learning (ICML) 97*: 6105–6114.

Wang, R., T. Lei, R. Cui, B. Zhang, H. Meng, and A. K. Nandi. 2022. "Medical image segmentation using deep learning: A survey." *IET Image Processing 16*(5): 1243–1267.

Xie, S., R. Girshick, P. Dollár, Z. Tu, and K. He. 2017. "Aggregated residual transformations for deep neural networks." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*: 1492–1500.

Xu, G., W. Liao, X. Zhang, C. Li, X. He, and X. Wu. 2023. "Haar wavelet downsampling: A simple but effective downsampling module for semantic segmentation." *Pattern Recognition 143*: 109819.

Yang, Z., X. Wang, J. Xiang, J. Zhang, S. Yang, X. Wang, W. Yang, Z. Li, X. Han, and Y. Liu. 2023. "The devil is in the details: a small-lesion sensitive weakly supervised

learning framework for prostate cancer detection and grading." *Virchows Archiv 482*(3): 525–538.

Zhang, A., S. Zheng, Y. He, X. Tan, H. Dang, and Q. Yuan. 2022. "Mini-resizer: a minimalist learnable resizer for image geolocation." *CIBDA 2022; 3rd International Conference on Computer Information and Big Data Applications*: 1–4.

Zhang, J., Y. Zhang, S. Zhu, and X. Xu. 2020. "Constrained multi-scale dense connections for accurate biomedical image segmentation." *2020 IEEE international conference on bioinformatics and biomedicine (BIBM)*: 877–884.

Zhou, Z., M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang. 2018. "UNet++: A nested U-Net architecture for medical image segmentation." *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support (DLMIA 2018, held in conjunction with MICCAI 2018)*: 3–11.

Zou, L., H. F. Lam, and J. Hu. 2023. "Adaptive resize-residual deep neural network for fault diagnosis of rotating machinery." *Structural Health Monitoring 22*(4): 2193–2213.