National Technical University of Athens

School of Electrical and Computer Engineering

Data Science and Machine Learning

Study of image segmentation models on different datasets

DIPLOMA THESIS

Georgios Nanos

Supervisor: Stefanos Kollias
Professor NTUA

Artificial Intelligence and Learning Systems Laboratory

Athens, September 2023

National Technical University of Athens
School of Electrical and Computer Engineering
Data Science and Machine Learning
Artificial Intelligence and Learning Systems Laboratory

Study of image segmentation models on different datasets

DIPLOMA THESIS

Georgios Nanos

Supervisor: Stefanos Kollias
Professor NTUA

Approved by the three-member committee of examiners on 28th September, 2023.

.....	.....	.....
Stefanos Kollias	Athanasios Voulodimos	Giorgos Stamou
Professor NTUA	Professor NTUA	ProfessorNTUA

Athens, September 2023

.....
NANOS GEORGIOS
Electrical Engineer &
Computer Engineer NTUA
Machine Learning Engineer NTUA

Abstract

The process of segmenting an image into different regions or objects is a critical step in computer vision and image analysis. In recent years, deep learning-based image segmentation models have demonstrated remarkable performance on various benchmark datasets. In this study, we have conducted a comprehensive analysis of several state-of-the-art image segmentation models on different benchmark datasets, including the Corsican Fire dataset. The models investigated in this thesis include DeepLabv3+, HRNet, HRNet+OCRNet, DDRNet, PIDNet, Swin Transformer, BASNet, BISNet, UNET3+, Attention U-Net, and EMANet. Each model has a unique architectural design and distinct features, which are tested on several benchmark datasets to evaluate their performance. The Corsican Fire dataset is a challenging dataset that consists of high-resolution images of forests captured during wildfire outbreaks. To test the model's robustness, we have also included other benchmark datasets, which are widely used in the field of image segmentation. Our experimental results reveal that the Attention U-Net, EMANet and Swin Transformer outperform the other models in terms of overall accuracy segmentation and speed on the Corsican fire dataset. In addition, we performed an extensive comparative analysis of the models on other datasets and the results show that each model has its strengths and weaknesses depending on the characteristics of the dataset and the task segmentation task. Therefore, in the remainder of this study, it was shown that different models perform better in the transfer learning technique for its evaluation performance on other datasets. Overall, our study provides a comprehensive evaluation of state-of-the-art image segmentation models on different benchmark datasets, highlighting their performance, strengths, and limitations. The findings of this study can serve as a benchmark for researchers in the field of computer vision and image analysis and can aid in the development of more efficient and accurate image segmentation models.

Keywords — Image Segmentation, Semantic Segmentation, Instance Segmentation, Deep Learning, Fully Convolutional Networks (FCNs), Convolutional Neural Networks (CNNs), Performance Evaluation, Attention Mechanisms, Loss Functions, Training, and Validation, Testing, and Evaluation

Acknowledgements

I would first like to thank Assistant Professor Kollias Stefanos for the supervision of this thesis and for the and for the opportunity he gave me to work on it at the Artificial Intelligence and Learning Systems Laboratory. Also, special thanks to Mrs. Tzouveli Paraskevi for her support and the excellent collaboration we had. Finally I would like to thank the parents my parents for the guidance and support they have offered me throughout these years.

Nanos Georgios
September 2023

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Introduction

The object of this thesis is to study image segmentation models on various reference datasets using a variety of state-of-the-art deep learning models. The purpose of this research is to investigate the performance of these models for the task of image segmentation on multiple datasets using metrics such as Intersection over Union (IoU), Precision, Dice Loss and Recall, along with some variations. Image segmentation has become a critical task in computer vision due to the exponential increase in the volume of image data and has numerous applications in various fields. The task of image segmentation involves dividing an image into multiple segments, where each segment represents a different object or region within the image.

In this thesis, we analyzed and compared the performance of several deep learning models, including Attention U-Net (AUNet), Boundary-Aware Salient Object Detection (BASNet), BiSeNet, Deep Dual-Resolution Network (DDRNet), DeepLabv3+, HRNet, HRNet+OCR, Efficient Multi-Scale Aggregation Network (EMANet), Progressive Instance Discrimination Network (PIDNet), Swin Transformer and U-Net 3+. Each of these models has its unique architecture, loss function and optimization algorithm. The reference datasets used in this study represent a wide range of real-world scenarios, including fire images of natural scenes, and various medical images.

To evaluate the performance of these models, we used several metrics such as IoU, Precision, Dice Loss and Recall, along with some variations of these metrics. We also analyzed the effect of various hyperparameters on the performance of each model, such as learning rate, batch size and number of epochs. At the same time we used the transfer learning technique and analyzed the advantages it offered.

The results of this study demonstrate that the performance of deep learning models varies significantly depending on the dataset and the specific image segmentation task. Our findings provide insights into which models perform well under different conditions and which models are more suitable for specific applications. Furthermore, this research contributes to the development of robust and efficient image segmentation models that can be used in a wide range of practical applications. Overall, this thesis provides a comprehensive analysis of deep learning models for image segmentation on multiple reference datasets using a variety of state-of-the-art models and metrics. The results of this study can serve as a valuable resource for researchers and practitioners in the field of computer vision, providing a deeper understanding of the performance of different models and their suitability for various applications.

1.2 Organization of the Thesis

This thesis is organized in several chapters, each of which covers different aspects of the topic of semantic segmentation.

The chapter 1 provides an introduction to the problem of semantic segmentation and its importance in various applications. It also presents the research objectives and the research questions that the thesis aims to answer.

Chapter 2 focuses on a more detailed description of the issue of semantic segmentation, including an overview of the three main types of semantic segmentation: semantic segmentation, instance segmentation and panoptic segmentation. In addition, this chapter provides motivation for research and a review of related work in the field.

Chapter 3 presents the theoretical background of the thesis, including machine learning and deep learning, neural networks of different types of convolution, transformers, attention mechanisms and encoder-decoder architectures.

Chapter 4 describes the reference datasets used in the experiments, the Corsical Fire Database, the COVID-QU-Ex Dataset, the Kvasir-Instrument medical image database and the Cell cell image database and discusses their characteristics. This chapter also covers the preprocessing steps performed on the dataset.

Chapter 5 explains the application of modern models for semantic segmentation to the Corsican Fire Database. In particular, this chapter covers and analyses in detail the models DeepLabv3+, HRNet, HRNet+OCRNet, DDRNet, PIDNet, Swin Transformer, BASNet, BISNet, UNET3+, Attention U-Net and EMANet. This chapter also presents the evaluation metrics used to compare and evaluate the models.

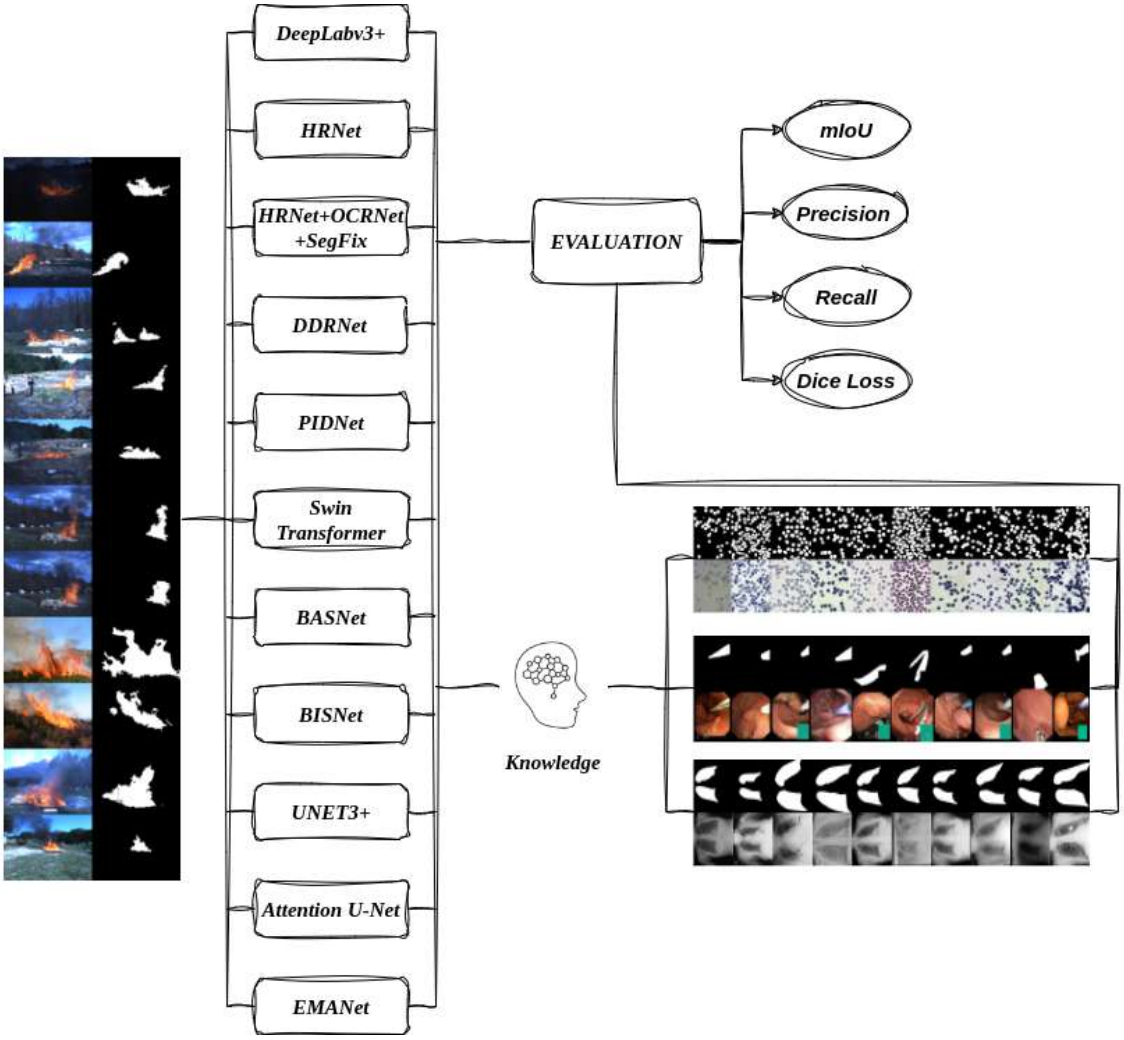

Figure 1.2.1: The pipeline of this thesis.

Chapter 6 provides a summary of the results obtained from the experiments and the knowledge gained from them. It introduces and analyzes the merits of the transfer learning technique and discusses the limitations of the research and provides recommendations for future work.

Finally, Chapter 7 presents the conclusions of the thesis, including a restatement of the research objectives and research questions, a summary of the main findings and the contribution of the research. This chapter also suggests possible applications of the research and suggests directions for future work.

Chapter 2

Description of the Problem

2.1 History of Image Segmentation

In computer vision, image segmentation refers to the process of dividing a digital image into multiple sets of pixels or segments. This has been a well-established challenge in computer vision for some time. However, semantic segmentation represents a more advanced technique that involves segmenting an image while having a comprehensive understanding of each pixel within the image. In essence, semantic segmentation involves the analysis and classification of each pixel into various categories. Numerous algorithms have been developed to tackle this complex task, such as the Watershed algorithm, Image Thresholding, K-means clustering and Conditional Random Fields.

2.1.1 Image Thresholding Image Thresholding

Image thresholding [SSW88] is a fundamental algorithm used to segment an image by dividing it into two or more classes of pixels - i.e., foreground and background. Typically, this is achieved by first converting the original image to grayscale and then applying the thresholding technique to obtain a binary image. Under this approach, the intensity value of each pixel is compared to a predefined threshold value. If the intensity is less than the threshold value, the pixel is assigned a value of 1 (white), and if it is greater, the pixel is assigned a value of 0 (black).

The main challenge in implementing this simple algorithm is determining the optimal threshold. For simple applications, the threshold value can be set manually by the algorithm designer. However, in real-world scenarios, it is important that the threshold is determined automatically. A popular technique to achieve this goal is the Otsu [Ots79] method, developed by Nobuyuki Otsu in 1979. This algorithm determines the threshold by minimizing the within-class intensity variance or, equivalently, by maximizing the between-class variance. Specifically, the algorithm exhaustively searches for the threshold that minimizes the weighted sum of the deviations of the two classes, where the weights are the probabilities of the two classes separated by the threshold. The resulting threshold corresponds to the minimum variance within the class.

The optimal threshold value is given by:

$$T = \underset{t}{\operatorname{argmin}}(\sigma_w^2(t)) \tag{2.1.1}$$

where T is the optimal threshold value, $\sigma_w^2(t)$ is the variance of the intensity within the class and the minimum is obtained for all possible threshold values t.

The variation within the class is defined as follows:

$$\sigma_w^2(t) = \omega_0(t)\sigma_0^2(t) + \omega_1(t)\sigma_1^2(t) \tag{2.1.2}$$

where $\omega_0(t)$ and $\omega_1(t)$ are the probabilities of the two classes separated by the threshold t, and $\sigma_0^2(t)$ and $\sigma_1^2(t)$ are the deviations of these classes. The optimal threshold is the one that minimizes the intraclass variance, which corresponds to the threshold that maximizes the intraclass variance. Figure 2.1.1 illustrates a thresholding case using the Otsu method. The left side shows the initial grayscale image, while the center shows the binary image after thresholding is applied. On the right side, the intensity histogram with the threshold marked is shown.

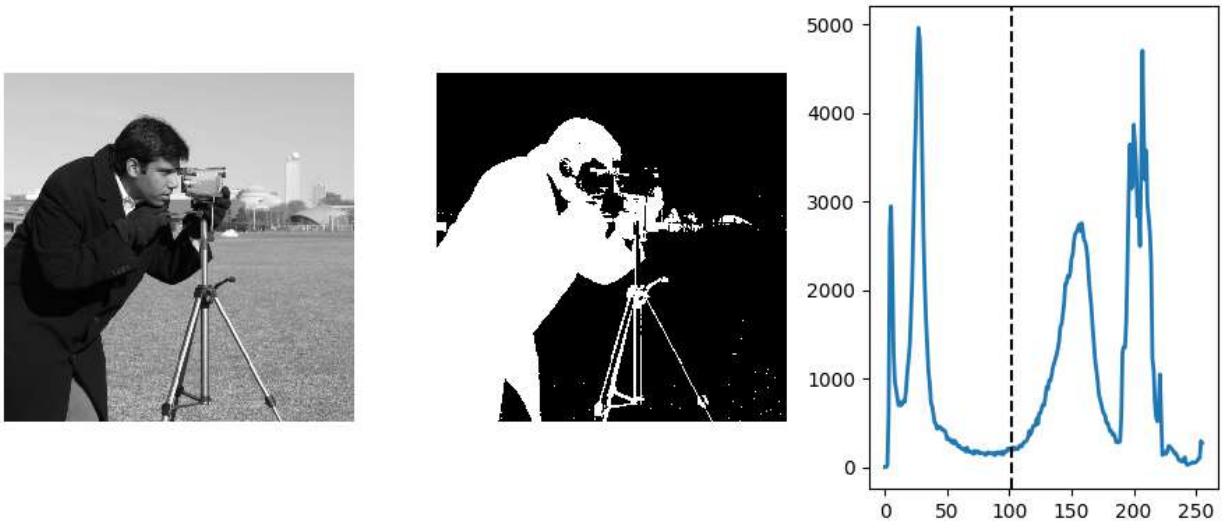

Figure 2.1.1: Demonstration of the threshold value through the use of the Otsu approach

2.1.2 Conditional Random Fields

Random field methods are widely used in image analysis to model spatial regularities. These methods are useful for tasks ranging from low-level noise reduction to high-level object recognition and segmentation. While previous work used generative models based on random Markov fields, Conditional Random Fields (CRFs) [TV07a] have become popular due to their ability to directly predict image segmentation given the observed image. CRFs are effective tools for various segmentation and labeling tasks, including visual scene interpretation, which divides images into semantic-level regions and assigns class labels to each region. Accurate labeling requires capturing both global and local information in the image.

Jakob Verbeek and Bill Triggs [TV07b] proposed a model that represents images as rectangular surfaces at a single scale, with each surface having a hidden class label. The CRF model incorporates 4−geometric links between patch labels, and texture, color, and position descriptors are used to encode the local content of the image. Texture is represented using a 128−dimensional SIFT descriptor quantized with a $ks = 1000$ text word dictionary learned by k-means clustering of all patches in the training set. Position is encoded by overlaying the image with a grid of $m \times m$ cells ($m = 8$) and using the index of the cell in which the patch falls as the position feature. Each patch is thus encoded by three binary vectors, each of which has a set of one bit corresponding to the observed visual word. The CRF observation functions are simple linear functions of these three vectors.

Verbeek et al. found that models based on visual word histograms were very successful for image categorization. They took into account the global context by including observation functions based on histograms of their visual word patches across the image. A conditional model for patch labels was defined that incorporates both local patch-level features and global aggregate features. The x_i label of patch i, y_i denoting the W dimensionalized convolved binary index vector of its three visual words ($W = k_s + h_h + k_p$) and h denoting the normalized histogram of all visual words in the image, i.e., the P patches, i y_i normalized for sum one. The conditional probability of the label x_i is modeled as follows:

$$p(xi = l|yi, h) \propto \exp(-\sum_{w=1}^{W}(\alpha_{wl}y_{iw} + \beta_{wl}h_w l)) \tag{2.1.3}$$

where α_{wl}, β_{wl} are $W \times C$ coefficient matrices to be learned. This is a multiplicative combination of a local classifier based on observation at the y_i patch level and a global framework or bias based on the h image-level histogram.

The performance of the segmentation models was measured on the Microsoft Research Cambridge (MSRC) dataset [AZ18], which consists of 240 images with partial pixel-level labels. The labels assign pixels to one of nine classes, but about 30% of the pixels are unlabeled. For model evaluation, 20 pixels with centers per 10 pixels were used to evaluate the models. To obtain patch labels, pixels were assigned to the nearest patch center. Patches were allowed to have any label they appeared between their pixels, while unlabeled pixels were allowed to have any label. To map the patch level segmentation back to the pixel level, each pixel was assigned the patch margin with the nearest center. The model that took into account both local and global contexts and did not exclude unlabeled pixels achieved a return of 84.9%.

The figure 2.1.2 shows various segmentation results from the MSRC dataset. A Gaussian filter was used to post-process the segmentation maps by smoothing the pixel margins and the scale was set to half the distance of the surfaces.

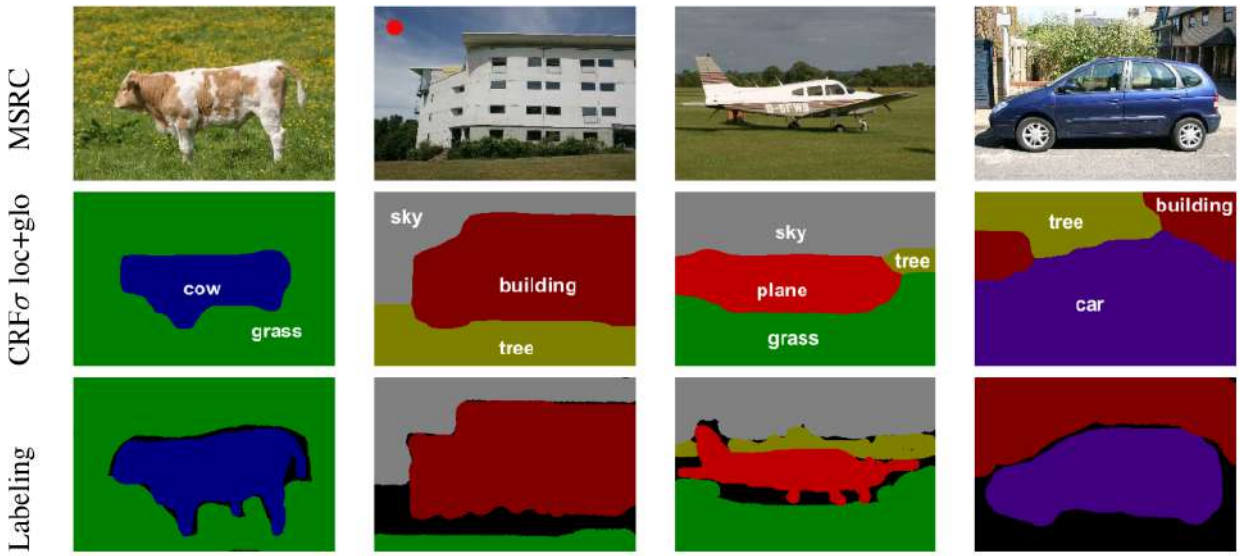

Figure 2.1.2: Examples taken from MSRC
, which include segmentation and tagging [AZ18].

2.2 Semantic Segmentation

Semantic segmentation [Tho16] is a computer vision task that involves assigning a label to each pixel in an image, where the label indicates the object or region to which the pixel belongs. In other words, the goal of semantic segmentation is to produce a pixel-by-pixel segmentation map that accurately describes the objects and regions in an image.

The task of semantic segmentation is challenging because objects can vary significantly in shape, size and appearance and may be partially occluded or covered by other objects in the image. In addition, images may contain multiple objects or regions that overlap or are adjacent to each other, making it difficult to accurately segment individual objects.

To address these challenges, deep learning-based approaches to semantic segmentation have been developed. These approaches typically use convolutional neural networks (CNNs) to learn a mapping from the input image to the output segmentation map. The CNN consists of a series of layers designed to extract increasingly abstract features from the input image. The output of the CNN is then processed by a decoder, which produces the final segmentation map.

One of the key advantages of using CNNs for semantic segmentation is their ability to capture high-level features that are useful for distinguishing between different classes of objects. For example, CNNs can learn to recognize the shapes and texture properties of different objects, as well as the spatial relationships between objects in the image.

For accurate segmentation of objects in the image, it is important to take into account both local and general contexts. The local context refers to the relationship between adjacent pixels, while the general context refers to the relationship between different regions of the image. Deep learning-based approaches to semantic segmentation often use a combination of techniques to capture both local and global contexts. These techniques include aggregation and convolutional layers that operate at different scales, skip links that allow information to be transmitted between different layers of the network, and spatial pyramid aggregation, which captures features at multiple scales.

The performance of semantic segmentation models is usually evaluated using metrics such as intersection on union (IoU), accuracy (Precision), Recall and F1-score. These metrics provide a quantitative measure of how well the model is able to segment objects in the image. IoU measures the overlap between the predicted segmentation and the actual segmentation (ground truth), while Precision measures the ratio of correctly predicted pixels to the total number of predicted pixels. Recall measures the ratio of correctly predicted pixels to the total number of ground truth pixels, and F1-score is the harmonic mean of Precision and Recall.

In general, semantic segmentation is a difficult computer vision task that involves assigning a label to each pixel of an image. Deep learning-based approaches have been developed for semantic segmentation using convolutional neural networks to capture high-level features and contexts. These approaches have achieved top performance on a range of benchmark datasets and have numerous applications in areas such as autonomous driving, robotics and medical imaging.

2.3 Instance Segmentation

Instance segmentation [HB20] is a complex computer vision task that aims to simultaneously classify and segment each individual instance of an object in an image. Unlike semantic segmentation, which assigns a common label to all pixels belonging to a particular object class, instance segmentation produces unique labels for each object instance present in the image. The output of object segmentation is a binary mask that accurately describes the boundaries of each object in the image.

The task of segmenting objects is quite difficult, as it requires the model to accurately distinguish between different instances of objects belonging to the same class. This means that the model must be able to segment individual objects that may have different sizes, shapes, orientations and postures, as well as objects that may be partially covered or overlapped by other objects.

Deep learning-based approaches have shown promise for solving the case segmentation problem. These approaches typically use a two-step process consisting of region proposal and segmentation. In the first stage, a Region Proposal Network (RPN) is used to identify candidate object regions in the image. The RPN generates a set of bounding box proposals that are likely to contain objects of interest. In the second stage, a segmentation network is used to classify and segment each candidate region. The segmentation network takes the candidate site proposals generated by the RPN and produces a case-level segmentation mask for each proposal.

To capture both local and general context, case segmentation models typically use a combination of techniques such as multi-scale feature extraction, skip links and feature pyramid networks. These techniques allow the model to extract high-level features that can help distinguish between different object instances and improve the accuracy of instance segmentation.

The performance of presentation segmentation models is usually evaluated using metrics such as Average Precision (AP) and Average Recall (AR). AP measures the accuracy of object detection and segmentation, while AR measures the ability of the model to detect all instances of a class of objects. These metrics are commonly used to evaluate the performance of object segmentation models on reference datasets, such as COCO [Lin+15], which contains a large number of object instances in complex scenes.

In general, instance segmentation is a difficult computer vision task that requires the model to accurately distinguish between different instances of objects in an image. Deep learning-based approaches have achieved significant performance on reference datasets. Case segmentation has numerous applications in areas such as autonomous driving, robotics and surveillance. Further research is needed to improve the accuracy and

efficiency of instance segmentation models and to extend their capabilities to handle more complex and dynamic scenes.

2.4 Panoptic Segmentation

Panoptic segmentation [Kir+18] is a demanding computer vision process that requires a model to simultaneously perform semantic and contextual segmentation on an image. The goal of panoptic segmentation is to create a single image-level segmentation mask that combines both semantic and case information. Panoptic segmentation models typically use a two-step process consisting of object detection and segmentation to accomplish this task. The first stage involves using an object detection network to detect and classify different object instances in the image, while the second stage involves using a segmentation network to generate case-level segmentation masks for each detected object instance.

As in presence segmentation, one of the main challenges in panoptic segmentation is to accurately distinguish between different instances of objects of the same category. Deep learning-based approaches involve techniques such as feature fusion and mask aggregation to combine the high-level semantic information captured by the semantic segmentation network with the instance-level information captured by the instance segmentation network.

To evaluate the performance of panoptic segmentation models, the Panoptic Quality (PQ) metric [Kir+18] is commonly used, which measures the accuracy of the model in generating segmentation masks at both the semantic and instance levels. The PQ metric is calculated as the product of two factors: the segmentation quality SQ, which measures the accuracy of the segmentation masks at the instance level, and the recognition quality RQ, which measures the accuracy of the semantic segmentation masks.

Panoptical segmentation has numerous applications in areas such as robotics, autonomous driving and surveillance. For example, panoptic segmentation can be used in robotics to allow robots to navigate and interact with their environment more efficiently by accurately detecting and localizing objects. In the context of autonomous driving, panoptical segmentation can be used to detect and track other vehicles and pedestrians in real-time, allowing autonomous vehicles to make safer and more informed decisions. In the field of surveillance, panoptical segmentation can be used to detect and track people and objects of interest in large-scale environments.

Despite the progress made in the area of panoptic segmentation, there is still much room for improvement. Future research is needed to develop more accurate and efficient panoptic segmentation models and to extend their capabilities to handle more complex and dynamic scenes. In addition, efforts should be made to make these models more accessible and applicable to real-world scenarios and to address the challenges associated with training and developing large-scale panoptic segmentation models on resource-constrained devices.

2.5 Motivation for Diploma Thesis

The study of image segmentation models on various reference data sets is a key research area in computer vision, with far-reaching implications for a wide range of applications. Accurate and efficient image segmentation has the potential to improve the performance of autonomous systems and advance the field of medical imaging, among other benefits.

The primary motivation behind this study is to gain a deeper understanding of the capabilities and limitations of state-of-the-art deep learning models for image segmentation. While deep learning has demonstrated tremendous potential in computer vision, image segmentation remains a challenging problem. Evaluating different models on different reference datasets can provide valuable insights into their strengths and limitations, enabling the development of more effective approaches for image segmentation that can be adapted to different types of images and tasks.

Advances in image segmentation can also drive progress in related areas such as robotics, autonomous driving and healthcare. Accurate and efficient image segmentation is crucial for robots to navigate and interact with their environment and for real-time object detection and tracking in the context of autonomous driving. In

healthcare, image segmentation is essential for detecting and diagnosing medical conditions, such as cancer, and for monitoring the progression of the disease over time.

Overall, by better understanding the capabilities and limitations of modern image segmentation models and developing more accurate and efficient algorithms, we can work to solve some of the most challenging computer vision problems of our time and make significant advances in related areas.

2.6 Related Work

In this extensive exploration, we delve into a rich body of existing research and literature that inspires and shapes our subject matter. Focusing at the intersection of image segmentation, fire dataset exploitation, and inter-domain transfer learning in medical image datasets, our research draws on a wide range of work that collectively contribute to our overall effort.

2.6.1 Fire Image Segmentation

In the field of fire image segmentation, researchers have developed advanced architectures and innovative strategies to address the challenges of fire detection and analysis. The pioneering work [Maj+22] introduces a CNN architecture, channeling spatial and spectral features to achieve real-time fire detection and segmentation in aerial imagery. Their contribution highlights the importance of incorporating contextual cues to accurately delineate fire areas. Complementing this, Gupta [NG18] curated a specialized fire segmentation dataset and devised a custom DeepLabv3+ [Che+18] model, demonstrating the power of domain-specific training to achieve accurate detection and refinement of fire area boundaries.

2.6.2 Medical Image Segmentation

Medical Image Segmentation: a Review of Modern Architecture [STK23] research involves identifying key regions in medical images to improve diagnosis. The study focuses on powerful computer vision algorithms to speed up diagnoses, reduce costs, and enable faster disease prevention and treatment. The research examines prevailing methods, including the U-type architecture and the UNet model. These use encoder-decoder frameworks with convolutional networks. The study also explores transformer architectures using attention mechanisms and residual learning to enhance information integration. This paper evaluates eleven models designed for medical image segmentation and extends to other areas. Augmentation addresses the limited medical data by enhancing the size of the dataset and the adaptability of the models. In addition, the research explores the interaction between data augmentation and outcomes, offering insights into its impact. The study highlights the potential of models, and describes pathways for enhancing medical image segmentation.

2.6.3 Machine Learning and Satellite Imagery

Satellite image super-resolution for forest localization [LTK23] examines the evolving contribution of satellite missions, Earth observation and machine learning methodologies for enhanced environmental monitoring. However, one obstacle lies in the limited spatial resolution of satellite imagery. To overcome this challenge, this study exploits Image Super Resolution Networks to improve the resolution of remote sensing images. The central objective is to perform semantic segmentation using Sentinel-1 and Sentinel-2 satellite images [Lan+18] to identify forested areas. The research uses three different deep neural network architectures and subjects them to rigorous training and testing. In addition, a deep super-resolution network is used to increase the resolution of the image, resulting in two datasets - original and super-resolution images. The architectures are trained on both datasets and evaluated on a test set that includes both original and super-resolved images. The study reveals that the Swin-Transformer architecture yields the most promising results, with training on super-resolved images enhancing modeling efficiency across all architectures. This research suggests the potential of combining advanced machine learning techniques and satellite imagery to address critical global challenges such as deforestation monitoring.

2.6.4 Transfer Learning in Medical Image Segmentation

The field of transfer learning is unfolding as a catalyst for bridging the gap between different areas of the medical picture. Lee and colleagues [Lee+20] orchestrate knowledge transfer mechanisms by architecting a pre-trained HRNet [Wan+19], cultured for lung segmentation, on a cell segmentation dataset. This orchestration highlights the seamless exchange of learned features between tasks, culminating in improved segmentation accuracy. Similarly, Chen et al.[Che+20] highlight the potential of the BiSeNet [Yu+18] architecture, orchestrating its harmonious adaptation for segmentation of various medical tools and lung structures. Their effort echoes the importance of architectural flexibility and domain knowledge to advance medical image segmentation.

2.6.5 Advanced Architectures for Image Segmentation

At the forefront of image segmentation, the place is shared by innovative architectures that are reshaping the field of precision and detailed analysis. Zhang and colleagues [Li+22] illuminate this field with HRNet, an architecture enriched with an optical character recognition (OCR) module. This innovation spawns fine-grained feature extraction, empowering the HRNet + OCR fusion to decipher complex segmentation challenges across diverse datasets. Swin Transformer [Liu+21] stands out, with a hierarchical self-attention mechanism. This fusion enriches Swin Transformer's ability to capture both general context and complex local specifics in complex medical segmentation tasks.

2.6.6 Hybrid and Combinatorial Approaches

The field of segmentation strategies is enriched by the mix of hybrid and ensemble approaches, where different methods are combined to produce accurate results. Liang et al.[Ala+23] presented UNet3+ [Hua+20], a modified version of the UNet [RFB15] design, enhanced with attention mechanisms and skip connections. This improvement helps the model capture both detailed and global features in segmentation. At the same time, the work, BASNet, brings a boundary-focused approach to segmentation. The sensitivity of BASNet [Qin+19] to specific structures highlights the importance of boundary information, a key element in achieving accurate results. Moving to the ensemble aspect, EmaNet [Li+19] combines the expertise of several segmentation approaches to achieve highly accurate results. This demonstrates the power of combining different methods for improved performance.

The paper, entitled "Deep neural architectures for prediction in healthcare", presents innovative deep neural architectures for disease diagnosis and personalised assessment in healthcare, highlighting their applicability to neurodegenerative diseases such as Parkinson's disease [Kol+18]. Building on these foundations, the paper "Assessment of Parkinson's Disease Based on Deep Neural Networks", which proposes a system for analysing medical imaging data related to Parkinson's disease and correlating it with clinical information, demonstrating its ability to accurately assess disease status [TKS17]. In addition, the publication "Machine Learning for Neurodegenerative Disorder Diagnosis - Survey of Practices and Launch of Benchmark Dataset" highlights the importance of early diagnosis and offers a comprehensive archive of medical tests related to neurodegenerative diseases, with a focus on Deep Learning. In parallel, a benchmark dataset on Parkinson's disease is presented, further highlighting the potential of research [Tag+18]. In "Predicting Parkinson's Disease using Latent Information extracted from Deep Neural Networks", an approach for medical diagnosis of neurodegenerative diseases using latent information extracted from deep neural networks is presented. The effectiveness of this method in predicting Parkinson's disease based on MRI and DaT scan data is demonstrated [KSK19]. The study, titled "A Unified Deep Learning Approach for Prediction of Parkinson's Disease", unifies community efforts into a unified framework for predicting Parkinson's disease using deep learning and medical imaging. This approach shows promise in various medical settings, further substantiating its effectiveness [Win+20].

In addition, a database and a deep learning approach for COVID-19 detection is presented, contributing to the pandemic response effort [Kol+21]. "A Large Imaging Database and Novel Deep Neural Architecture for Covid-19 Diagnosis", continues to address the COVID-19 challenge by developing a large annotated database for the diagnosis of COVID-19 based on chest CT scans. The RACNet architecture, which is superior to other deep neural networks, is also presented [AKK22]. "AI-MIA: COVID-19 Detection & Severity Analysis through Medical Imaging", which presents a basic approach for the 2nd Covid-19 competition and introduces an annotated database for COVID-19 detection and severity analysis, together with a deep learning model

for [KAK23b]. "Data-Driven Covid-19 Detection Through Medical Imaging", introduces a new COVID-19 database and a deep learning model for accurate classification of CT scans. This approach demonstrates superior performance compared to existing models [Ars+23].

The research "Adaptation and contextualization of deep neural network models", proposes innovative systems for the adaptation and contextualization of trained DNNs, addressing issues related to transparency and efficiency [Kol+]. In "Deep Transparent Prediction through Latent Representation Analysis", a new approach to extract latent information from DNNs for transparent and accurate predictions is presented. This approach finds applications in various fields, including medical imaging and disease prediction [Kol+20a]. The paper "A Deep Neural Architecture for Harmonizing 3-D Input Data Analysis and Decision Making in Medical Imaging" focuses on harmonizing 3-D medical data analysis and decision making using the RACNet architecture. This innovative approach enhances the capabilities of deep neural networks in medical imaging. [KAK23a]. "Transparent adaptation in deep medical image diagnosis" introduces a method for transparent adaptation of DNNs, enabling consistent and reliable predictions across different contexts and datasets. This approach is applied to the prediction of Parkinson's disease and COVID-19 [Kol+20b].

The study "Deep Bayesian Self-Training" and "Capsule routing via variational bayes", proposes methods for automatic data annotation and improving the efficiency of capsule networks, respectively [DLK20] [De +20] [RLK20]. In the field of nuclear reactor analysis, the papers "A deep learning approach to anomaly detection in nuclear reactors" and "Machine learning for analysis of real nuclear plant data in the frequency domain" contribute. These papers present innovative approaches for anomaly detection in nuclear reactors using deep learning and machine learning [Cal+18] [Kol+22]. Furthermore, the studies entitled "An autoencoder wavelet based deep neural network with attention mechanism for multi-step prediction of plant growth" and "Using deep learning to predict plant growth and yield in greenhouse environments" are explored, presenting applications of deep learning in predicting plant growth and yield in controlled environments [Alh+21] [Alh+19].

For facial expression recognition the paper "MixAugment & Mixup: Augmentation Methods for Facial Expression Recognition" presents MixAugment, a new data augmentation strategy that outperforms existing methods in handling different facial expressions [PK22]. Furthermore, "Abaw: Valence-arousal estimation, expression recognition, action unit detection & emotional reaction intensity estimation challenges", includes extensive corpora and challenges related to the analysis of emotional behaviour [Kol+23]. Finally implemented applications in image analysis problems and human-computer interaction [04; 99; Wal+03; AXK01; 96]. are new work in the field of Artificial Intelligence.

2.6.7 Conclusion and Future Horizons

By looking carefully at these different contributions, we gain a deeper understanding of how image segmentation evolves. These different insights help us in our efforts to explore and expand the possibilities of image segmentation. Our work builds on the foundation of knowledge gained from previous research.

Chapter 3

Theoretical Background

3.1 Machine learning & Deep learning

3.1.1 Machine Learning

Machine learning [Sim18] is a branch of artificial intelligence (AI) that allows computers to learn from data and make predictions or decisions without being explicitly programmed. The goal of machine learning is to develop algorithms and models that can automatically identify patterns and ideas in data and make predictions or decisions based on those patterns. There are three main types of machine learning: supervised learning, unsupervised learning, and reinforcement learning [HA19]. Supervised learning is the most common type of machine learning and involves training a model on a tagged dataset where the output variable or target is known. The model is then tested on new, unknown data to evaluate its performance. Examples of supervised learning include image classification, speech recognition, and natural language processing.

Unsupervised learning, on the other hand, involves training a model on an unlabeled data set where the output or target variable is unknown. The model is then used to identify patterns and information in the data, such as clustering or dimensionality reduction. Examples of unsupervised learning include anomaly detection, market segmentation and feature extraction.

Reinforcement learning is a type of machine learning that focuses on training models to make decisions based on feedback or rewards. It is often used in situations where the outcome of a decision is uncertain or the optimal decision may change over time. Examples of reinforcement learning include robotics, games and control systems.

The process of machine learning usually begins with data collection and cleaning. This step is crucial, as the quality of the data can significantly affect the performance of the model. After the data is cleaned and prepared, it is then divided into a training set and a test set. The training set is used to train the model, while the test set is used to evaluate the performance of the model.

The next step is to select a suitable model and algorithm for the task at hand. There are many different models and algorithms such as decision trees, random forests, neural networks and support vector machines. The choice of model and algorithm depends on the type of problem, the complexity of the data and the available resources.

Once the model is selected, it is then trained on the training set. The goal of training is to find the best parameters for the model that will minimize the error in the training set. The model is then evaluated on the test set to measure its performance and ensure that it does not overfit the training data. Once the model is trained and evaluated, it can then be used to make predictions on new, unseen data. This is often referred to as inference or development. The model can be deployed in various applications, such as image classification, speech recognition or natural language processing. Machine learning encompasses a wide range of applications and has been applied in a variety of industries, such as healthcare, finance, transportation and manufacturing. For example, in healthcare, machine learning has been used to develop models that can predict patient outcomes, identify potential health risks and assist in medical diagnosis. In finance, machine learning has been applied to identify unfair trades, predict stock prices and identify potential investment opportunities.

Machine learning has also been used to improve the efficiency of transport systems, such as self-driving vehicles, traffic prediction and optimisation, and logistics optimisation. In industry, machine learning has been exploited to optimise production processes, improve quality control and predict equipment failures. Overall, machine learning is a powerful tool that can help organisations make better decisions and improve their operations. As more and more data is generated, machine learning will continue to be a key tool for organisations looking to gain knowledge and improve their performance. However, it is important to note that machine learning is not without its challenges. Some of the challenges include privacy and data security, ethical concerns and the need for large amounts of high quality data.

One of the main challenges of machine learning is data privacy and security. As machine learning models are trained on large amounts of data, it is vital that this data is protected and kept confidential. This can be particularly challenging when dealing with sensitive data, such as personal information or financial data. Organisations need to ensure that they have appropriate data management procedures in place to protect against data breaches and unauthorised access to data. Another challenge of machine learning is ethical

considerations. As machine learning models are used to make decisions and predictions, it is important to ensure that these decisions and predictions are fair and unbiased. This can be particularly challenging when dealing with sensitive data, such as data related to race, gender or income level. Organisations need to ensure that their models do not inadvertently perpetuate existing biases and discrimination.

Finally, machine learning requires a large amount of high-quality data. The performance of machine learning models depends largely on the quality and quantity of the data on which they are trained. Without sufficient data, it may be problematic to train models that are accurate and reliable. In addition, data must be cleaned and pre-processed to ensure that it is free of errors and inconsistencies. This can be a time-consuming and resource-intensive process. Overall, machine learning is a powerful tool that can help organisations make better decisions and improve their operations. However, it is important to be aware of the challenges and limitations of machine learning and have appropriate processes in place to address these challenges. With the right approach and the right resources, organisations can successfully harness the benefits of machine learning while minimising the risks.

3.1.2 Deep Learning

Deep learning [Sch15] is a subfield of machine learning that uses deep neural networks to model and solve complex problems. The concept of deep learning has its roots in the field of artificial intelligence and has gained rapid popularity in recent years due to its ability to achieve top performance in a wide range of tasks such as image classification, speech recognition and natural language processing.

Deep neural networks, also known as deep learning models, consist of multiple layers of interconnected artificial neurons, where each layer receives input from the previous layer and produces output for the next layer. The architecture of deep neural networks allows them to automatically learn and extract features from raw input data and perform complex nonlinear transformations. This contrasts with traditional machine learning models, which typically require manual feature design. Deep learning models can be trained in different ways, depending on the application. As an example, supervised learning is the most common method used to train deep learning models, where the model is trained on a tagged dataset and the goal is to minimize the difference between the predicted output and the actual output. Other training methods include unsupervised learning, where the model is trained on an unlabeled dataset, and reinforcement learning, where the model learns to make decisions based on feedback or rewards.

In addition, deep learning has been applied to achieve top performance in a wide range of applications, such as image classification, speech recognition, and natural language processing. In image classification, for example, deep learning models have been used to automatically identify objects and scenes in images and to achieve performance that is comparable or even superior to human performance. In speech recognition, deep learning models have been used to automatically transcribe speech and to achieve performance that is comparable or superior to traditional speech recognition methods. In natural language processing, deep learning models have been used to automatically understand and produce natural language and to achieve performance comparable or superior to traditional natural language processing methods.

In addition, deep learning is a rapidly evolving field and new techniques and architectures are being developed at a rapid pace. In recent years, research has focused on the development of more complex and powerful architectures, such as convolutional neural networks, recurrent neural networks, and transformational networks. Furthermore, new techniques for training deep learning models, such as transfer learning and domain adaptation, have been developed to improve the performance of deep learning models on new tasks and domains.

In general, deep learning is a powerful tool that can be used to solve a wide range of problems and has the potential to revolutionize many disciplines. However, like machine learning it is important to note that deep learning is not without its challenges, such as the need for large amounts of high quality data, the complexity of the models and the computational resources required to train and develop deep learning models.

3.2 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) [ON15] is a type of deep neural network designed specifically for processing data that has a grid-like topology, such as images, videos and audio signals. CNNs consist of multiple layers, where each layer applies a set of filters to the input data and produces a feature map as output. The filters are learned during training and are designed to automatically extract significant features from the input data.

The first layer of an CNN, also known as the input layer, receives the raw input data, such as an image. Subsequent layers, known as convolutional analysis layers, apply a set of filters to the input data and produce a feature map as output. These filters are learned during training and are designed to automatically extract meaningful features from the input data. The filters are typically small and spatially local, which allows them to capture local patterns in the input data. Filters are applied to the input data using a process known as convolution, where the filter is moved to the input data and the dot product between the filter and the input data is computed at each location. The result of the convolution is a feature map, which is a representation of the input data that is more abstract and compact than the original input data.

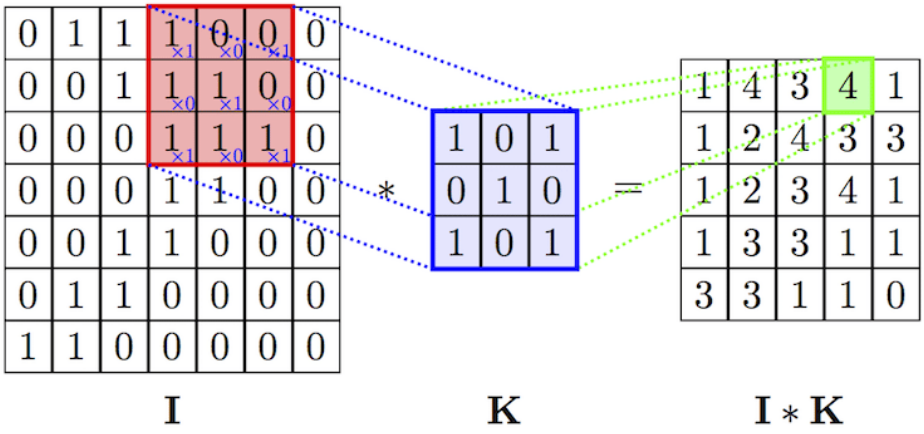

Figure 3.2.1: Two-dimensional Convolution

The convolutional resolution levels are followed by aggregation levels, which are used to reduce the spatial resolution of the feature maps and increase the robustness of the feature maps to small variations in input data. The most common pooling formula is the maximum pooling (max pooling), where the maximum value of a small subregion of the feature map is taken as the output value for that subregion. This feature effectively reduces the feature map samples and also makes the feature maps more robust to small changes in the input data.

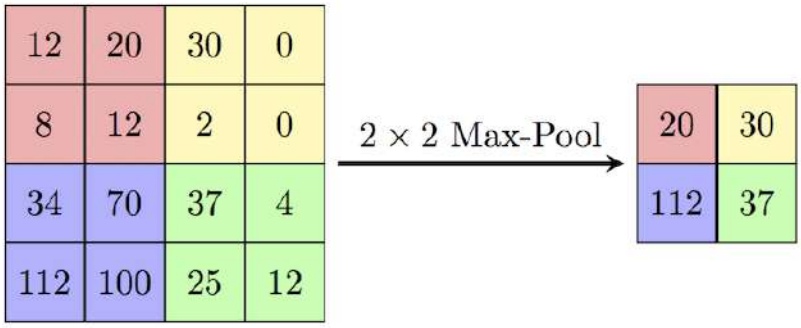

Figure 3.2.2: Max Pooling [Nag+11].

After the aggregation layers, the feature maps are then passed through fully connected layers, which are used to make the final predictions. These layers are similar to the fully connected layers in traditional neural

networks and are used to learn a nonlinear function that maps the feature maps to the output predictions.

Fully convolutional neural networks (FCNs) [LSD14] are an extension of CNNs, and are designed to process data of any size, not just a fixed size as in CNNs. This is achieved by replacing fully connected layers with convolutional layers, which allows the network to process data of any size and maintain a fixed number of parameters. This architecture is particularly useful for tasks such as semantic segmentation, where the goal is to assign a semantic label to each pixel of an image.

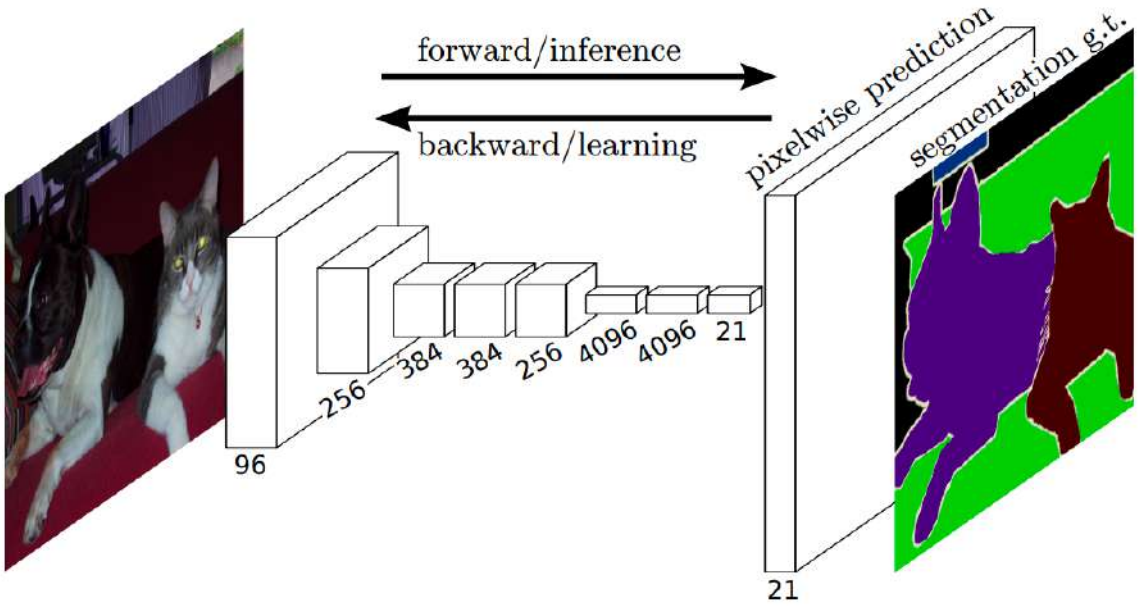

Figure 3.2.3: Fully convolutional networks are an architecture mainly used for semantic segmentation [LSD14].

In FCN, the last layers of CNN are replaced by 1×1 convolutional layers, which are used to reduce the number of feature maps and increase the spatial resolution of the feature maps. This is achieved by using a technique called post-convolution, also known as deconvolution, as shown in 3.2.4, which increases the spatial resolution of the feature maps using a set of filters learned during training.

Finally, the output of the last layer is passed through a softmax activation function [DSC21], which produces a probability map for each class. The class with the highest probability for each pixel is selected as the final prediction.

In summary, CNN and FCN are powerful architectures designed specifically for processing data that has a grid-like topology, such as images, video and audio signals. CNN consist of multiple layers, where each layer applies a set of filters to the input data and produces a feature map as output. The filters are learned during training and are designed to automatically extract meaningful features from the input data. FCN is an extension of CNN and is designed to process data of any size by replacing fully connected layers with convolutional layers, which allows the network to maintain a constant number of parameters while processing data of any size. These architectures have achieved top performance in a wide range of tasks such as image classification, object detection and semantic segmentation.

3.3 Recurrent Neural Networks

Recurrent neural networks (RNN) is a type of artificial neural network that can process successive data while retaining information from previous inputs. They have been successfully applied in various fields such as natural language processing, speech recognition and time series prediction.

At a high level, an RNN processes a sequence of inputs $\mathbf{x} = \mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T$ and produces a sequence of outputs $\mathbf{y} = \mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T$. The basic idea of an RNN is to use a hidden state vector $\mathbf{h}_t$ to capture information from

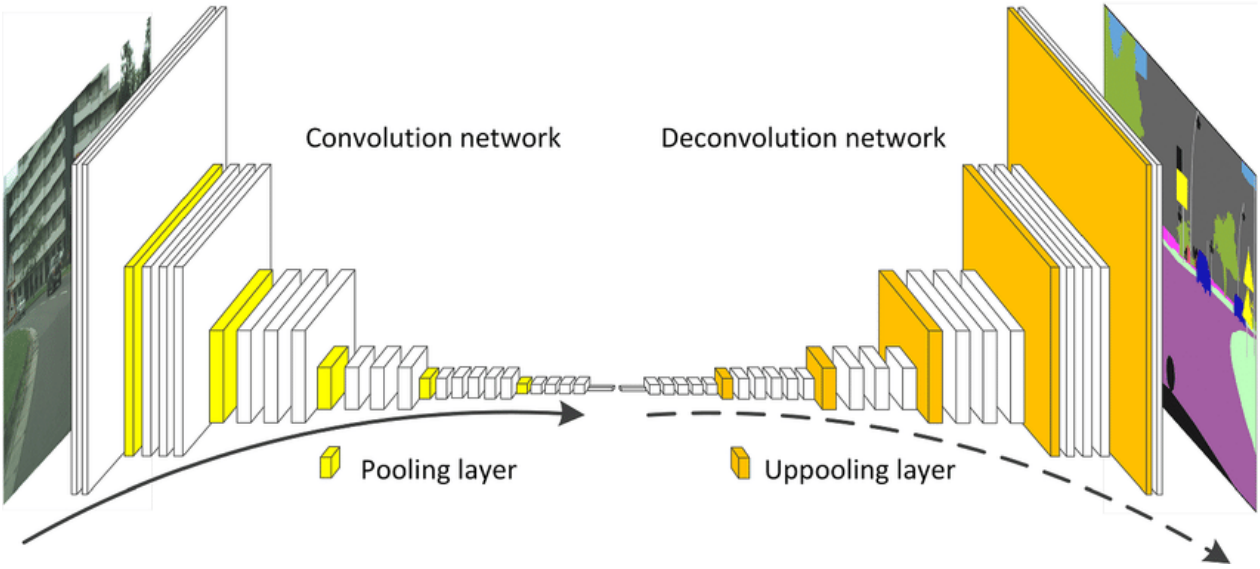

Figure 3.2.4: The architecture of fully convolutional networks for semantic segmentation together with the convolutional and deconvolutional components [LDY19].

the previous inputs in the sequence and use this hidden state to compute the next output in the sequence. The hidden state is updated at each time step based on the current input and the previous hidden state:

$$\mathbf{h}_t = f(\mathbf{x}t, \mathbf{h}t - 1) \tag{3.3.1}$$

where f is a nonlinear function that maps the input and the previous hidden state to a new hidden state. The output at each time step is then computed as a function of the current hidden state:

$$\mathbf{y}_t = g(\mathbf{h}_t) \tag{3.3.2}$$

where g is a nonlinear function that maps the hidden state to the output. One of the key advantages of RNNs is their ability to handle arbitrary-length input sequences, as the hidden state captures information from all previous inputs of the sequence. However, a major challenge in training RNNs is the problem of disappearing or gradient descent, which occurs when the gradients used to update network parameters become too small or too large, making it difficult to learn long-term dependencies. To address this problem, several variants of RNNs have been proposed, including Long Short-Term Memory (LSTM) and Gated Recurrent Unit (GRU) architectures, which use specialized gating mechanisms to control the flow of information within the network.

LSTM networks consist of memory cells that can store and retrieve information for long periods of time. At

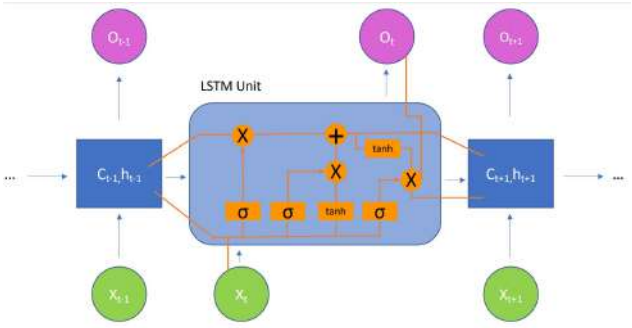

Figure 3.3.1: Diagram of a Long Short Term Memory Module (LSTM) [Nos21].

each time step, the contents of the cell are updated based on the current input, the previous hidden state,

and a set of gate vectors that control the flow of information in and out of the cell:

$$\mathbf{i}_t = \sigma(\mathbf{W}_{xi}\mathbf{x}_t + \mathbf{W}_{hi}\mathbf{h}_{t-1} + \mathbf{b}_i) \tag{3.3.3}$$

$$\mathbf{f}_t = \sigma(\mathbf{W}_{xf}\mathbf{x}_t + \mathbf{W}_{hf}\mathbf{h}_{t-1} + \mathbf{b}_f) \tag{3.3.4}$$

$$\mathbf{o}_t = \sigma(\mathbf{W}_{xo}\mathbf{x}_t + \mathbf{W}_{ho}\mathbf{h}_{t-1} + \mathbf{b}_o) \tag{3.3.5}$$

$$\tilde{\mathbf{c}}_t = \tanh(\mathbf{W}_{xc}\mathbf{x}_t + \mathbf{W}_{hc}\mathbf{h}_{t-1} + \mathbf{b}_c) \tag{3.3.6}$$

$$\mathbf{c}_t = \mathbf{f}_t \odot \mathbf{c}_{t-1} + \mathbf{i}_t \odot \tilde{\mathbf{c}}_t \tag{3.3.7}$$

where σ is the sigmoid function [DSC21], tanh is the hyperbolic tangent function and $\mathbf{i}_t$, $\mathbf{f}_t$ and $\mathbf{o}_t$ are the entry, exit and exit gates, respectively.

GRU networks are similar to LSTM, but use a smaller number of control vectors to control the flow of information within the network. At each time step, the network computes a set of update and reset gates, which determine how much of the previous hidden state and current input are used to compute the new hidden state.

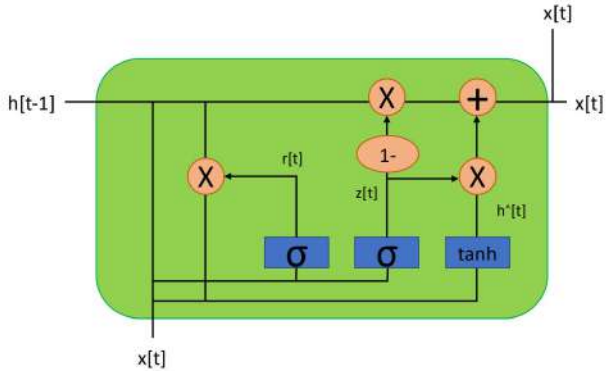

Figure 3.3.2: A Gated Recurrent Unit (GRU) [Nos21].

$$\mathbf{z}_t = \sigma(\mathbf{W}_{xz}\mathbf{x}_t + \mathbf{W}_{hz}\mathbf{h}_{t-1} + \mathbf{b}_z) \tag{3.3.8}$$

$$\mathbf{r}_t = \sigma(\mathbf{W}_{xr}\mathbf{x}_t + \mathbf{W}_{hr}\mathbf{h}_{t-1} + \mathbf{b}_r) \tag{3.3.9}$$

$$\tilde{\mathbf{h}}_t = \tanh(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hr}(\mathbf{r}_t \odot \mathbf{h}_{t-1}) + \mathbf{b}_h) \tag{3.3.10}$$

$$\mathbf{h}_t = (1 - \mathbf{z}_t) \odot \mathbf{h}_{t-1} + \mathbf{z}_t \odot \tilde{\mathbf{h}}_t \tag{3.3.11}$$

The output at each time step is then calculated based on the current hidden state:

$$mathbfy_t = g(\mathbf{h}_t)$$

where g is a nonlinear function. Overall, the RNN architecture is a powerful tool for processing sequential data, with the ability to capture long dependencies and handle input sequences of arbitrary length. The LSTM and GRU variants of RNN have shown very promising results in various applications and have become the standard choice for many sequence processing tasks. In terms of mathematical functions, the basic RNN architecture can be formulated as follows:

$$\mathbf{h}_t = f(\mathbf{x}_t, \mathbf{h}_{t-1}) = \tanh(\mathbf{W}_{xh}\mathbf{x}_t + \mathbf{W}_{hh}\mathbf{h}t - 1 + \mathbf{b}_h)\ \mathbf{y}_t = g(\mathbf{h}_t) \tag{3.3.12}$$

where $\mathbf{W}_{xh}$, $\mathbf{W}_{hh}$, and $\mathbf{b}_h$ are the weight matrix and the bias vector for computing the hidden state, f is the transition function, and g is the output function.

3.4 Atrous Convolution

Atrous convolution, also known as dilated convolution, is a technique used to increase the receptive field of a convolutional neural network (CNN) without increasing the number of parameters. Atrous convolution is achieved by introducing gaps between the elements of the convolution kernel, which allows the kernel to cover a larger area of the input feature map while keeping the number of parameters constant. In traditional convolution, the convolution kernel is applied to the input feature map by sliding over the feature map and computing the dot product between the kernel and the overlapping region of the feature map. This process results in a reduction of the spatial resolution of the feature map. In atrous convolution, the kernel is applied to the input feature map by introducing gaps between the kernel elements, which increases the size of the receptive field without reducing the spatial resolution of the feature map.

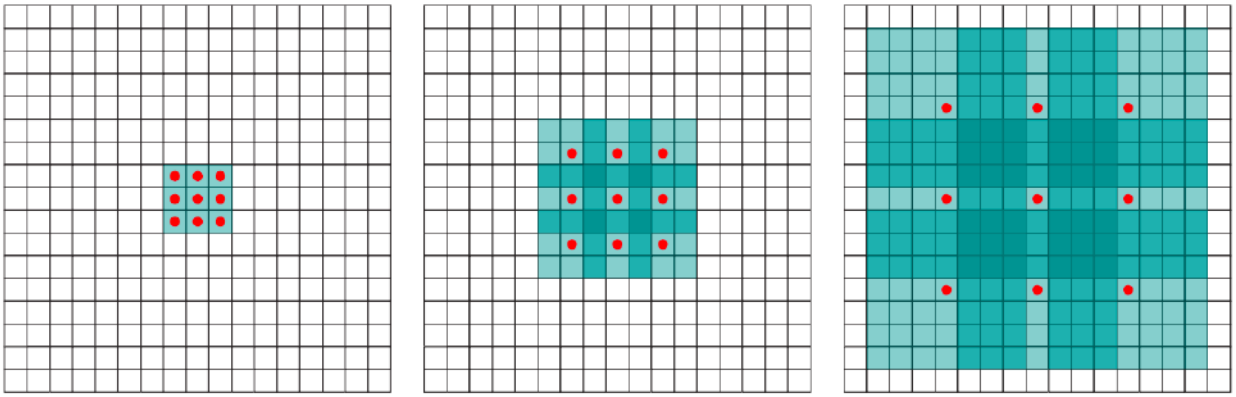

Figure 3.4.1: Examples of atrous convolution mechanisms with different expansion rates, in which the red squares represent the positions of the core parameters [YK15].

The size of the gaps, also known as the expansion rate, can be adjusted to control the size of the receptive field. An expansion rate of 1 corresponds to traditional convolution, while an expansion rate greater than 1 corresponds to atrous convolution. The larger the expansion rate, the larger the receptive field and the smaller the number of parameters. This allows capturing more context from the convolution kernel, which can be beneficial for applications such as image segmentation and object detection where context is important.

Atrous convolution can be implemented in different ways, depending on the specific task and the specific architecture of the CNN. One of the most common ways of implementing atrous convolution is to use the atrous convolutional layer, which is a variant of the traditional convolutional layer that has the ability to adjust the expansion rate. The atrous convolutional layer can be implemented using standard CNN libraries, such as TensorFlow [Mar+15] and PyTorch [Pas+19], and can be used in place of traditional convolution layers in a CNN architecture. Another way of implementing atrous convolution is to use the atrous spatial pyramid pooling (ASPP) module [Che+16]. The ASPP module is a technique that combines multiple atrous convolutional layers with different expansion rates to capture different context scales in the input feature map. The ASPP mechanism can be used in the encoder part of a CNN architecture to increase the content captured by the CNN and improve its performance in tasks such as image segmentation.

In addition, Atrous convolution can be combined with other techniques, such as upsampling and concatenation, to create an efficient network for semantic segmentation and example segmentation, this is done by using Atrous convolution in the encoder part of the network, followed by upsampling and concatenation layers in the decoder part of the network. This approach is known as Atrous spatial pyramid pooling (ASPP) and is widely used in state-of-the-art architectures such as DeepLabV3 [Che+17] and DeepLabV3+ [Che+18].

3.5 Attention Mechanisms

The attention mechanism is a technique used in neural networks to selectively focus on certain parts of the input data. It allows the model to weight the importance of different elements of the input data and produce

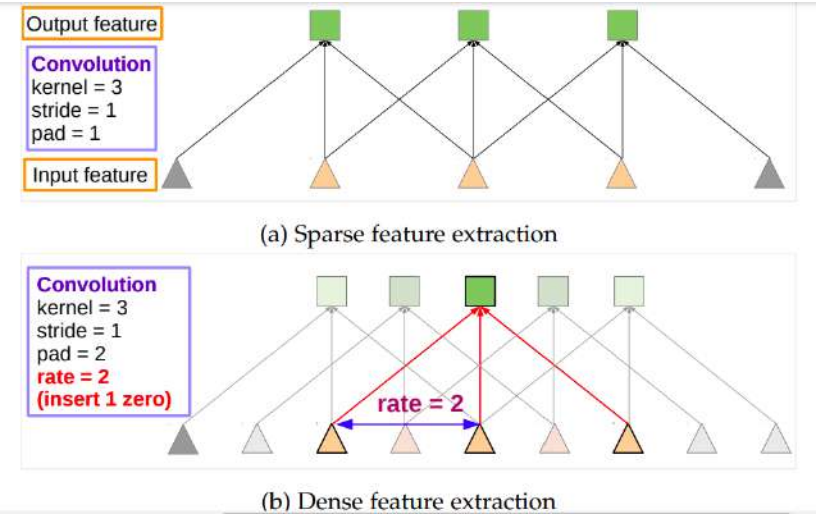

Figure 3.4.2: This figure illustrates the concept of the atrous convolution mechanism in a one-dimensional environment. In (a), a low-resolution input feature map is subjected to standard convolution to extract sparse features. In (b), a high-resolution input feature map is subjected to atrous convolution at a rate of r=2 to enable the extraction of dense features [Che+16].

a weighted sum of the input data, which is used as input for the next level. attention mechanisms have been widely used in a variety of tasks, such as natural language processing, image captioning, and speech recognition.

There are several types of attention mechanisms, each with its own characteristics and advantages. One of the most basic forms of attention is additive attention, also known as Bahdanau attention, which was introduced in the paper "Neural Machine Translation by Jointly Learning to Align and Translate" Bahdanau et al.in 2014. additive attention is used to compute the attention weights between elements of the input sequence using a combination of the values of the input elements and a set of learning parameters called attention weights. The attentionweights are computed using a feed-forward neural network and are used to generate a weighted sum of the input elements, which is then used as input for the next level. The Bahdanau attention mechanism is a popular approach to deep learning for sequence-by-sequence tasks such as machine translation. The formula for computing the Bahdanau attention weights is as follows:

$$a_t = softmax(v_a^T \cdot tanh(W_a \cdot h_t + U_a S_{t-1} + b_a)) \tag{3.5.1}$$

$$c_t = \sum_{i=1}^{T} a_{t,i} \cdot h_i \tag{3.5.2}$$

where:

- $\mathbf{h}_t$ is the t-first hidden state of the encoder
- $\mathbf{s}_{t-1}$ is the previous hidden state of the decoder
- $\mathbf{W}_a$, $\mathbf{U}_a$, $\mathbf{v}_a$ and $\mathbf{b}_a$ are learning parameters of the attention mechanism.
- $\boldsymbol{\alpha}_t$ is a vector of attention weights on the encoder hidden states, computed using a softmax function applied to a combination of the current decoder hidden state $\mathbf{s}_{t-1}$ and each encoder hidden state $\mathbf{h}_i$.
- $\mathbf{c}_t$ is the frame vector at time step t, which is a weighted sum of the hidden states of the encoder based on the attention weights $\boldsymbol{\alpha}_t$

Another form of attention isdot-product attention, also known asScaled Dot-Product Attention, which was introduced in the paper"Attention Is All You Need" byVaswani et al. in 2017. This mechanismattention calculates the weights by taking the product between the input elements and the weights and then scales the

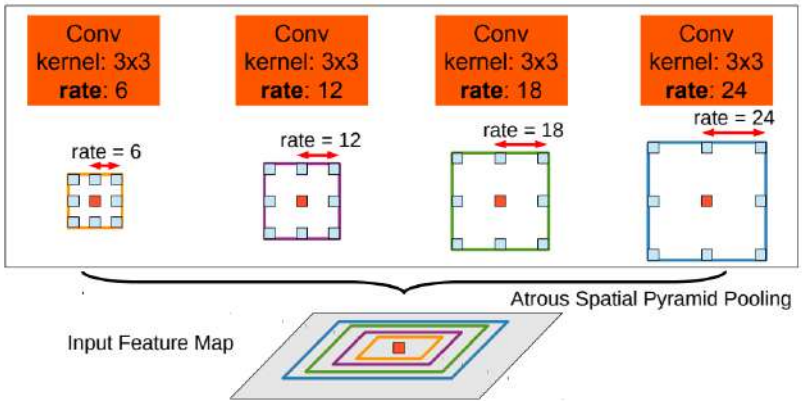

Figure 3.4.3: ASPP, short for Atrous Spatial Pyramid Pooling, uses several parallel filters at different rates to classify the central pixel (shown in orange), taking advantage of multi-scale features. The different colours demonstrate an effective Field-Of-Views [Che+16].

product by the square root of the dimension of the input elements. This attention mechanism is computationally more efficient than the additive attention and is used intransformer models [Vas+17]. Given a set of queries $Q \in \mathbb{R}^{n \times d_k}$, keys $K \in \mathbb{R}^{m \times d_k}$ and values $V \in \mathbb{R}^{m \times d_v}$, the attention mechanism computes a set of attention results $a(i, j)$ between each query $Q(i)$ and key $K(j)$ as follows:

$$a(i,j) = \frac{Q(i) \cdot K(j)}{\sqrt{d_k}} \tag{3.5.3}$$

where $\cdot$ denotes the operation of the quotient, d_k is the dimensionality of the keys, and $\sqrt{d_k}$ is a scaling factor that ensures that the dot quotient is not too large. The attention scores are then normalized using a softmax function, which results in a set of weights $w(i, j)$ that sum to 1:

$$w(i,j) = softmax(a(i,j)) \tag{3.5.4}$$

The weighted sum of the V values is then calculated using the caution weights as follows:

$$output(i) = \sum_j w(i,j) \cdot V(j) \tag{3.5.5}$$

where the sum is taken over all keys j. The scalar quadratic product attention mechanism can be implemented using matrix multiplication operations, where the queries, keys and values are represented as Q, K and V matrices, respectively, and the attention scores and weights are computed using the quadratic product and softmax functions applied to these matrices. Specifically, the attention scores are calculated as follows:

$$A = \frac{Q \cdot K^T}{\sqrt{d_k}} \tag{3.5.6}$$

where T denotes the shift operation and the weights are computed with the softmax function applied to the lines of A. Then the weighted sum of the values is calculated by matrix multiplication:

$$output = softmax(A) * V \tag{3.5.7}$$

where the softmax function is applied to the lines of A and the matrix multiplication is performed between the resulting weights and the matrix of values V.

Another form ofattention mechanism ismulti-head attention, which is used to compute multipleattention weights for input elements using multiple sets ofattention weights. Themulti-head attention mechanism allows the model to keep track of different parts of the input data at different locations, which can be beneficial for tasks such as natural language processing, where the order of items in the input data is important.multi-head attention is also used intransformer models, where it is used to computeself-attention between elements in

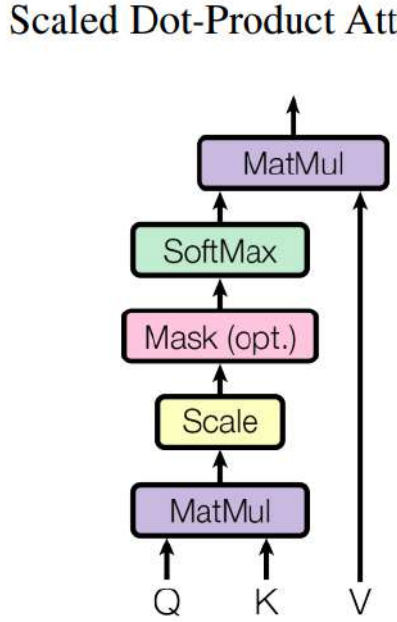

Figure 3.5.1: Scaled Dot-Product Attention [Vas+17].

the input sequence. Given the matrices Q, K and V, with the corresponding weight matrices W_q, W_k and W_v, we can define multi-head attention with H attention heads as follows:

$$\text{MultiHead}(Q, K, V) = \text{Concat}(\text{head}_1, \text{head}_2, ..., \text{head}_H)W_o \tag{3.5.8}$$

where each head can be defined as:

$$\text{head}_i = \text{Attention}(QWq, i, KW_{k,i}, VW_{v,i}) \tag{3.5.9}$$

and Attention is defined as:

$$\text{Attention}(Q, K, V) = \text{softmax}(\frac{QK^T}{\sqrt{d_k}})V \tag{3.5.10}$$

where d_k is the dimensionality of the key vectors and softmax is applied per row.

In the above equations, $/textConcat$ denotes the convolution along the dimension of the features, and W_o is a learning weight matrix applied to the convolution of the output. The matrices Q, K and V can be viewed as the queries, keys and values, respectively, of the attention mechanism, and W_q, W_k and W_v are weight matrices that project them into a common space. The multi-headed attention mechanism allows the model to simultaneously monitor different parts of the input, improving its ability to capture complex patterns in the data.

In addition to these types of attention, there are other variants, such as thesparse attention mechanism, which is designed to focus on a subset of items in the input data, and theaxial attention mechanism, which is designed to focus on specific regions in the input data. These variants and others were developed to improve the performance of the attention mechanism for solving more specific tasks.

Attention mechanisms have been shown to be effective in a wide range of tasks, such as natural language processing, image captioning, and speech recognition. They have been used to improve the performance of various neural network architectures, such as recurrent neural networks(RNN) and convolutional neural networks(CNN). Attention mechanisms are expected to continue to drive developments in machine learning and artificial intelligence.

Another important aspect of attentional mechanisms is interpretability. Attention mechanisms can be visualized as heat maps, which show the regions of the input data on which the model focuses. These heat

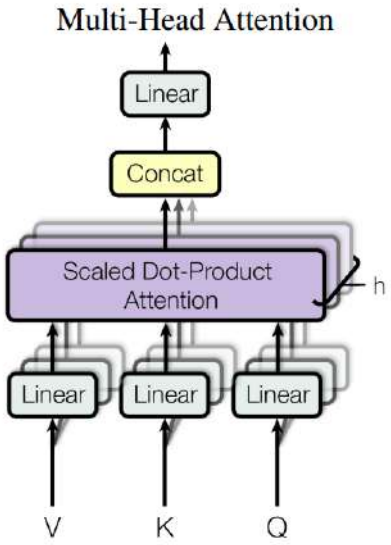

Figure 3.5.2: The Multi-Head Attention mechanism consists of several levels of attention working in parallel [Vas+17].

maps can be used to understand how the model makes its decisions and to identify any potential errors. This interpretability feature makes the attention mechanism a powerful tool for debugging and analyzing neural networks.

Concurrently, attention mechanisms are a technique used in neural networks to selectively focus on certain portions of the input data. They allow the model to weight the importance of different elements of the input data and produce a weighted sum of the input data, which is used as input for the next level. There are several types of attention mechanisms, such as additive attention, dot-attention and multi-head attention. They have been widely used in various tasks, such as natural language processing, image captioning and speech recognition, and are expected to continue to drive developments in machine learning and artificial intelligence.

3.6 Transformers

Transformers are a type of neural network architecture designed specifically for natural language processing tasks such as language translation, text summarisation and question answering. The architecture is based on the transformer model presented in the paper "Attention Is All You Need" [Vas+17] by Vaswani et al. in 2017. The transformer model consists of an encoder and a decoder, where the encoder is used to encode the input sequence and the decoder is used to generate the output sequence. Both the encoder and the decoder comprise a stack of identical layers, each comprising two sub-layers: a multi-head self-attention mechanism and a position-wise fully connected feed-forward network. The multi-head self-attention mechanism is the key element of the transformer model and is used to compute the attention weights between the elements of the input sequence. Attention weights are calculated using a combination of the positions of the input elements, their values and a set of learning parameters called attention weights. Attention weights are used to weight the importance of each element in the input sequence and are used to create a weighted sum of the input elements, which is then used as input for the next level. The fully coupled location-based feed-forward network is used to further process the output of the multi-head self-attention mechanism and consists of two linear transformations with a ReLU activation function [DSC21] in between. The feed-forward [BG94] network allows the model to increase its capacity and learn more complex representations of the input data. The transformer model also uses a technique called positional encoding [Che+21], which is used to embed the position of each element in the input sequence into the model. Positional encoding is added to the input embeddings and is used to provide the model with a sense of the relative position of the elements in the input sequence. This allows the model to understand the order of elements in the input sequence,

which is crucial for natural language processing tasks. One of the main advantages of the transformer model

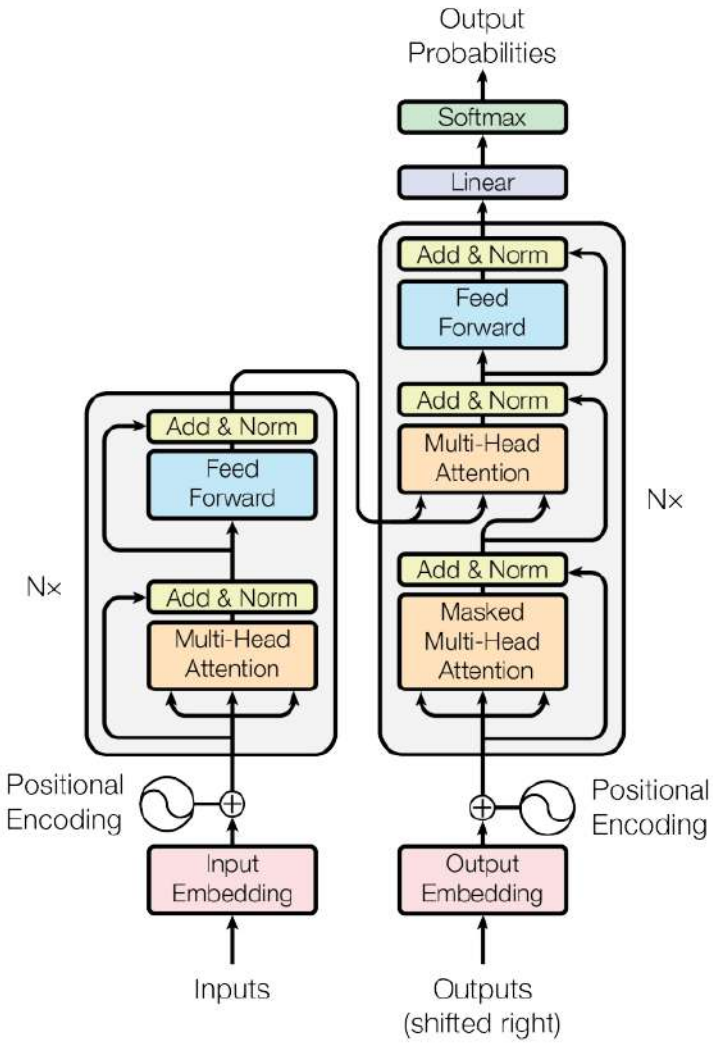

Figure 3.6.1: Transformer model with positional encoding [Vas+17].

is its ability to handle variable-length input sequences. This is achieved by the self-attention mechanism, which allows the model to keep track of different parts of the input sequence at different positions during the encoding process. In addition, the self-attention mechanism can be parallelized, which allows for faster computation and training. Another advantage of the transformer model is its ability to produce output sequences of variable length. This is achieved by the decoder, which uses the encoded input sequence and the attention mechanism to produce the output sequence. The transformer model is a powerful neural network architecture designed specifically for natural language processing tasks. The ability of the model to handle variable-length input sequences, its ability to parallelize the computation of the self-attention mechanism, and its use of positional coding are some of the reasons why it is so successful. The transformer model has achieved significant performance in a wide range of natural language processing tasks and is expected to continue to drive developments in the field of natural language processing.

3.7 Encoder-Decoder

The encoder-decoder is a deep learning architecture commonly used in computer vision for various image processing tasks, such as image segmentation, object detection, and image classification. The Encoder-Decoder architecture consists of two parts: an encoder network that extracts relevant features from the

input image and a decoder network that produces the output prediction. The encoder network is typically a deep convolutional neural network (CNN) that is responsible for extracting high-level features from the input image. The input image is passed through several convolutional network layers, where each layer applies a set of learning filters to the input to extract relevant features. The output of each layer is then passed through an activation function such as ReLU (Rectified Linear Unit) to introduce nonlinearity into the network. The output of the last layer of the encoder network is a compressed representation of the input image that preserves the most relevant information for subsequent operations. The decoder network, on the other hand, is responsible for generating the output prediction based on the features extracted from the encoder network. The decoder network usually consists of a series of decoding layers, which are also known as shifted symbolic layers or anode symbolic layers. These layers perform the reverse function of the convolutional layers, where they extend the feature map by increasing its spatial dimensions. The decoder network also uses bypass links, which allow the decoder network to access the lower layer features extracted from the encoder network. Skip connections are typically implemented by concatenating the output feature maps of the encoder network output to the input of the corresponding decoder layer. The decoder network also applies an activation function to the output of each layer, typically a sigmoid or softmax function, to produce the output prediction. For example, in image segmentation, the output prediction is a per-pixel probability map, where each pixel is assigned a probability of belonging to a particular class or object.

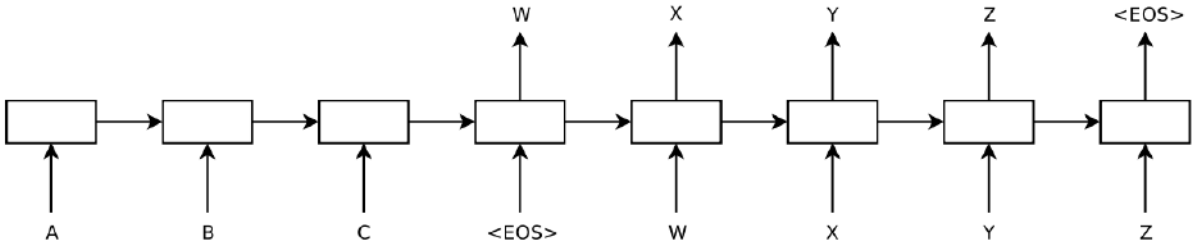

Figure 3.7.1: The architecture of the encoder-decoder model proposed in [SVL14], in which the network consists of two main parts: an encoder network and a decoder network.

The encoder network is shown on the left side of the figure 3.7.1 and consists of a recurrent neural network (RNN) with multiple layers. The input sequence is fed to the encoder one symbol at a time, with each symbol represented as a vector of real numbers. At each time step, the RNN updates its hidden state based on the current input symbol and its previous hidden state. The final hidden state of the RNN is used as a fixed-length representation of the input sequence, which is then passed to the decoding network.

The decoder network is shown on the right side of the figure 3.7.1 and also consists of a recurrent neural network with multiple layers. The decoder takes the fixed-length representation of the input sequence generated by the encoder as the initial hidden state and generates the output sequence one symbol at a time. At each time step, the RNN updates its hidden state based on the current input token and its previous hidden state and then generates a probability distribution over the possible output tokens using a softmax function. The output token with the highest probability is selected as the next token in the output sequence and the process is repeated until the end-of-sequence token is generated.

The encoder-decoder model is trained end-to-end using a variant of the backpropagation algorithm called backpropagation over time (BPTT) [BP21]. During training, the model learns to map input sequences to output sequences by minimizing a cross-entropy loss function between the predicted output sequences and the target sequences. Suppose we have a source sequence of length T and a target sequence of length L. The encoder takes as input the source sequence and generates a constant length frame vector $\mathbf{c}$, which represents the input sequence, as follows: For each time step t in the source sequence, the encoder takes the input vector $\mathbf{x}_t$ and passes it through a single-layer RNN with hidden size h. The hidden state of the RNN at time step t is denoted by $\mathbf{h}_t$:

$$mathbfh_t = f(\mathbf{x}t, \mathbf{h}t - 1)$$

The final frame vector

$$mathbfc$$

is computed as a function of the hidden states of the RNN at all time steps. A common way to compute $\mathbf{c}$ is to use the final hidden state $\mathbf{h}_T$ of the RNN encoder, although other methods can be used:

$$mathbfc = g(\mathbf{h}_1, \ldots, \mathbf{h}_T)$$

The decoder then uses the environment vector

$$mathbfc$$

and the previously generated target sequence (or a special sequence start symbol) to generate the next target symbol at each time step. In particular, the decoder operates as follows: For each time step l in the target sequence, the decoder takes as input the previous output y_{l-1} (or a special sequence start symbol) and the frame vector $\mathbf{c}$. The decoder passes these inputs through a single-layer RNN with a hidden size h. The hidden state of the RNN at time step l is denoted by $\mathbf{s}l$:

$$mathbfsl = f(yl - 1, \mathbf{c}, \mathbf{s}l - 1)$$

The decoder generates the output symbol y_l by passing the hidden state $mathbfs_l$ through a softmax layer:

$$y_l = \text{softmax}(\mathbf{W}\mathbf{s}_l + \mathbf{b})$$

The parameters of the RNN encoder and decoder, as well as the parameters of the softmax layer, are trained together to minimize the cross-entropy loss between the predicted target sequence and the actual target sequence.

$$Encoder : \mathbf{h}_t = f(\mathbf{x}t, \mathbf{h}t - 1)\mathbf{c} = g(\mathbf{h}_1, \ldots, \mathbf{h}_T) \tag{3.7.1}$$

$$Decoder : \mathbf{s}l = f(yl - 1, \mathbf{c}, \mathbf{s}_{l-1})y_l = \text{ softmax}(\mathbf{W}\mathbf{s}_l + \mathbf{b}) \tag{3.7.2}$$

One of the main advantages of the encoder-decoder architecture is its ability to capture both global and local context information from the input image. The encoder network captures the global context of the image by extracting high-level features that are relevant to subsequent tasks, while the decoder network captures the local context using skip links to access the lower-level features that contain detailed information about the image.

Another advantage of the encoder-decoder architecture is its ability to handle inputs of different sizes. Since the encoder network and decoder network are usually implemented as fully convolutional neural networks, they can process images of any size, making the architecture suitable for tasks such as image segmentation where the input images may have different dimensions.

There are several variants of the encoder-decoder architecture, such as U-Net, SegNet and Fully Convolutional Network (FCN). U-Net is a popular encoder-decoder architecture that is widely used in image segmentation tasks. The U-Net architecture consists of a shrinking path, which is the encoder network, and an expanding path, which is the decoder network. The contracting path consists of several convolutional processing layers, each of which is followed by a maximum aggregation layer, which reduces the spatial dimensions of the feature map. The expansive path consists of several deconvolutional layers, each of which is followed by a convolution with the corresponding feature map from the contractive path. The U-Net architecture also uses skip connections between the contracting and extending paths, allowing the decoding network to access the lower level features extracted by the coding network. The encoder-decoder architecture is a powerful deep learning architecture that has been widely used in computer vision for various image processing tasks. The architecture consists of an encoder network that extracts relevant features from the input image and a decoder network that produces the output prediction based on the extracted features. The encoder-decoder architecture is capable of capturing both global and local context information from the input image and can handle inputs of different sizes, making it suitable for a wide range of image processing tasks. The encoder network extracts relevant features from the input image, while the decoder network produces the output prediction based on the extracted features. The architecture is capable of capturing both global and local context information from the input image and can handle inputs of different sizes. The different variants of the encoder-decoder architecture, such as U-Net [RFB15], SegNet [BKC15], and FCN [LSD14], offer different trade-offs in terms of computational complexity, accuracy, and efficiency, making it important to choose the appropriate architecture for this work. Different variants of the architecture offer different trade-offs, and it is important to choose the appropriate architecture for the specific task.

3.8 Transfer Learning

Transfer learning in deep learning involves a powerful technique to enhance the performance of the model. By exploiting pre-trained models, which have been trained on large-scale datasets, transfer learning allows the extraction of general feature representations. These representations include high-level abstractions and patterns that can be applied to a wide range of tasks. The integration of transfer learning provides several key advantages, making it an essential tool in the deep learning paradigm. A major advantage of transfer learning is its ability to address data scarcity and high labeling costs. Obtaining labeled data for a target task can be a difficult and time-consuming process. Transfer learning mitigates this limitation by leveraging the knowledge gained from a source task and transferring it to the target task. The pre-learned model captures key features and patterns from the source task, allowing the model to generalize well to the target task even when limited labeled data is available. This advantage significantly reduces the burden of data collection and annotation, saving resources and time. In addition, the transfer learning technique accelerates the training process and improves convergence. The pre-trained models have already learned representations of fundamental features and patterns from huge amounts of data. Using these representations as initial parameters, transfer learning allows the model to start with a highly efficient initialization point. This initialization not only speeds up the training process but also enhances convergence, resulting in improved efficiency. This mechanism is particularly beneficial in situations where the pre-trained model and the target task share similar underlying patterns or structures. This alignment allows the pre-trained model to capture relevant information that can be transferred to the target task, leading to improved performance. The ability of the model to extract meaningful and distinctive features from the source task facilitates task-specific knowledge extraction during refinement or feature extraction. As a result, the model can be effectively adapted and specialized to the target task, leveraging its prior knowledge while focusing on task-specific nuances.

In addition, transfer learning provides a pathway for models to generalize well to unseen data and new tasks. The pre-trained model has already learned robust representations from a wide range of data. This broad exposure allows the model to capture key features that transcend specific domains or tasks. Consequently, when faced with new, unseen data, the model can leverage the representations it has learned to make accurate predictions and classifications. This generalization capability is a valuable advantage, as it reduces the risk of overfitting and enhances the robustness and adaptability of the model in different scenarios.

In summary, transfer learning in deep learning offers numerous advantages. It alleviates the challenges associated with lack of data and labeling costs by leveraging knowledge from an initial task. Reusing pre-trained models as initialization points speeds up the training process and improves convergence. In addition, the ability to capture task-specific knowledge while retaining general knowledge contributes to the adaptability and generalization capabilities of the model. By harnessing the power of transfer learning, deep learning models can achieve higher performance, robustness and efficiency, making it an indispensable technique in various applications.

Chapter 4

Benchmarks

This chapter analyses datasets used in the training of this thesis. We selected a combination of datasets in the field of image segmentation, which are differentiated in terms of their purpose, size dimensions and image shape, to present a comprehensive study around the models and their performance on each dataset.

4.1 Corsican Fire Database

The Corsican Fire database [Tou+17] was developed to meet the need for a comprehensive collection of wildland fire imagery. It includes a diverse set of images from various locations around the world, taken by different researchers and partners. The database includes both NIR and RGB images, all in PNG format. The database was first created in 2017, recognising the lack of a publicly available collection of large-scale wildland fire imagery and with the aim of providing an evolving open source repository for contributions. The database was selected for its extensive collection of well-annotated images, encompassing a wide range of wildfires and environments, including NIR and RGB data as well as manual segmentation masks. The images in the database have been taken primarily from UAVs, but also include some taken from the ground. This fire database is a valuable resource for researchers and practitioners working in the field of fire analysis, remote sensing and image processing.

Figure 4.1.1: Corsican Fire Database

In the following subsections, the database will be further analysed and characterised.

4.1.1 Analysis of the characteristics of the dataset

The Corsican Fire database is a collection of high-resolution images taken during various fires. The dataset contains a total of 1135 RGB images, of which 634 images are also accompanied by NIR (near infrared) data. These 634 images were acquired by a JAI AD-080GE camera, which is known for its high sensitivity and excellent image quality. These images are part of five different video sequences, each of which records a different fire event.

In addition, the Corsica fire database includes 419 different image sizes. Figure 4.1.2 illustrates the 10 most common image dimensions in the dataset. The prevalence of a particular image size can be attributed to the fact that all 634 images captured by the JAI AD-080GE camera have the same dimension 1024×768. This consistency in image size facilitates the processing and analysis of the dataset.

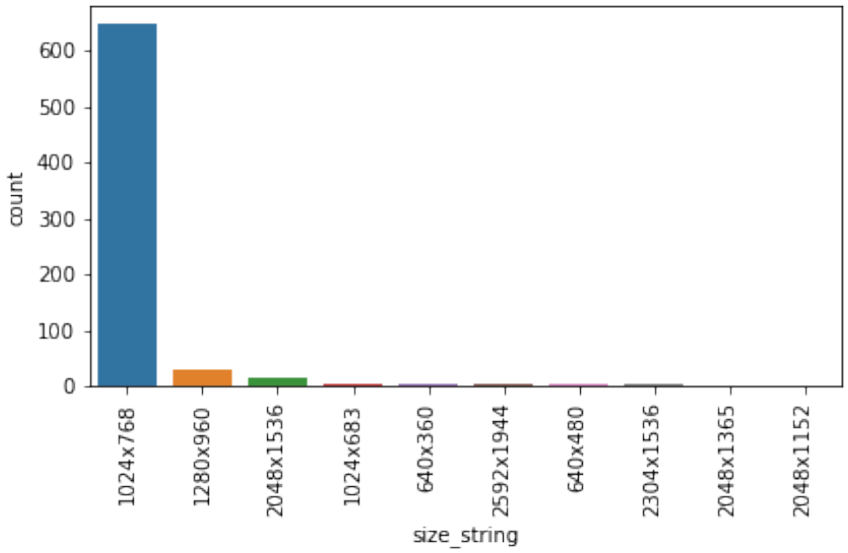

Figure 4.1.2: The top 10 image sizes that appear most often in the database.

4.1.2 Pre-processing of the Corsican Fire Database

It was decided to preprocess the entire dataset by cropping 256×256 and 128×128 dimension samples, respectively, without overlap from the 1135 images. This decision was made due to potential compatibility issues that arise in some architectures when the input dimension is not a power of two. Any dimension smaller was simply discarded. In addition, the ground truth masks and NIR images were cropped simultaneously, resulting in 87,672 samples. These samples were then indexed in a SQLite database with the corresponding ground truth and NIR truncation to allow for more efficient browsing. As this study focuses on segmentation rather than detection, images without fire pixels were removed from the dataset, even though they make up a significant proportion of the images, as shown in Figure 4.1.3. In addition, samples with an insufficient number of pixels were also excluded when fire was cropped close to the boundaries. To determine this threshold in pixel count, samples were examined manually, gradually increasing the value of this threshold until an area of fire that appeared significant was identified. All images with fewer than 20 fire pixels were excluded from the dataset, leaving 47,520 samples for further analysis.

In order to avoid overfitting and to allow for accurate model evaluation, a conventional method of separating the data into three subsets, namely a training set, a validation set and a test set, is used. These subsets are created with ratios of 0.7, 0.15 and 0.15, respectively.

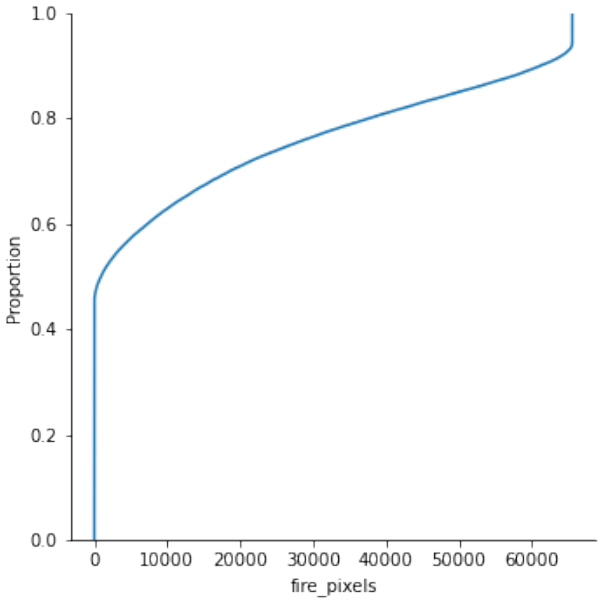

Figure 4.1.3: The empirical cumulative distribution of the samples is based on the number of fire pixels they contain.

4.2 COVID-QU-Ex Database

The COVID-QU-Ex [Tah+21] dataset, compiled by researchers at Qatar University, is a useful tool for people working in image segmentation, particularly in the context of respiratory diseases. With 33,920 chest X-ray images, including more than 11,000 COVID-19 cases, the dataset provides ample material for binary image segmentation. In addition, the lung segmentation masks provided with the dataset provide further support for the development of machine learning models for automatic detection and diagnosis of respiratory diseases. Overall, the COVID-QU-Ex dataset offers a comprehensive collection of high-quality chest X-ray images that are bound to prove useful for a wide range of research purposes, including binary image segmentation.

4.3 Kvasir-Instrument Databse

The Kvasir-Instrument dataset [Jha+20], a dataset, another addition to the range of medical imaging and analysis of gastrointestinal endoscopy. Gastrointestinal pathologies require complex procedures involving surgical instruments for tasks such as screening, biopsies and resections. However, the absence of comprehensive monitoring and analysis during and after these procedures results in the loss of vital information, including disease margins, developmental knowledge, and dimensions of resected areas. This knowledge gap impedes effective follow-up and complicates post-treatment reassessment, presenting significant challenges for both patients and medical professionals. Addressing these issues, the Kvasir-Instrument dataset includes 590 meticulous semiotic frames depicting a variety of gastrointestinal disease procedure tools such as forceps, balloons, biopsy forceps, and others. To ensure unparalleled accuracy, the dataset incorporates precisely edited ground truth masks and delineation frames, rigorously validated by two distinguished endoscopy experts. In particular, benchmarking using this dataset enables researchers to make a significant contribution to the advancement of automated endoscopic diagnostic and therapeutic segmentation tools for endoscopy of the gastrointestinal tract. As such, the Kvasir-Instrument dataset not only facilitates comprehensive analysis and accurate monitoring, but also fosters an environment conducive to cutting-edge research in this key medical discipline.

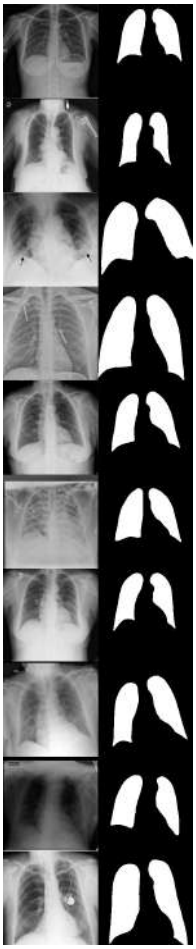

Figure 4.2.1: COVID-QU-Ex Database

4.4 Cell Dataset

The CELL [Sch+19] dataset is a thoroughly curated collection of fluorescence microscopy images that reveals the complex landscape of cellular entities. Comprising a rich collection of 1328 images, each carefully matched to its corresponding mask, this dataset is the quintessential representation of the fascinating universe of cellular microscopy. The genesis of the CELL benchmark finds its roots in the publication entitled "Multi-Domain Adversarial Learning". This dataset encapsulates the multifaceted complexities and dynamic features inherent in cellular structures. Within this dataset there is the potential to cultivate cutting-edge research and catalyze transformative discoveries across a range of disciplines.

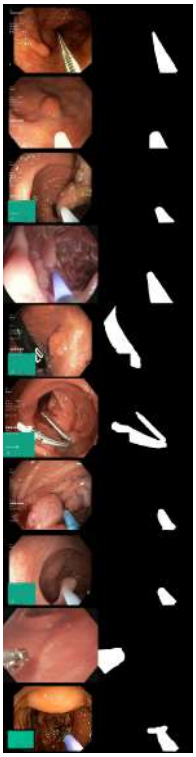

Figure 4.3.1: Kvasir-Instrument Database

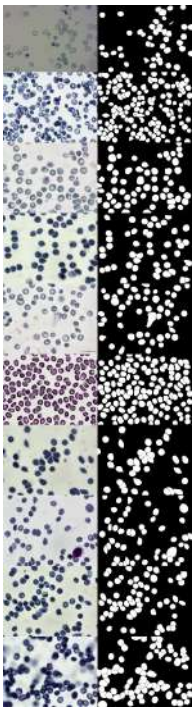

Figure 4.4.1: Cell Database

Chapter 5

Implementation

This chapter discusses the models, their design and configuration used in the experimentation of this thesis. A combination of models that have been recently and widely adopted in the field of image segmentation, as well as cutting-edge models were chosen to create a comprehensive evaluation. We also provide an in-depth discussion of the implementation details, such as the metrics used for the evaluation and the computational system used. The research was implemented using the Keras library, in conjunction with the TensorFlow framework.

5.1 DeepLabv3+

The use of both spatial pyramid pooling and encoder-decoder architectures in deep neural networks has been widely accepted for the task of semantic segmentation. These architectures have several advantages, such as the ability of the former to encode multi-scale context information and the ability of the latter to lead to more accurate object descriptions. The DeepLabv3+ [Che+18] model leverages the advantages of both architectures by incorporating an additional simple but effective decoder that enhances the accuracy of object outlines. The model, which is an extension of DeepLabv3 [Che+16], encodes rich semantic information and allows feature density control via atrous convolutions, adapting to available computational resources. Atrous convolutions, also known as dilated convolutions, are a powerful technique used in deep neural networks for the task of semantic segmentation. They allow to control the feature resolution and adjust the filter's field of view to capture multi-scale information. This is achieved by modifying the standard convolution function by introducing gaps, between the filter and the input feature map. The atrophy factor (r) controls the size of these gaps, and thus determines the sampling step of the input signal. When dealing with two-dimensional signals, atrous convolution is applied to the input feature map x at each location i in the output feature map y using a convolution filter w. The mathematical equation for this operation can be represented as follows:

$$y[i] = sumx[i + r \cdot k]w[k] \tag{5.1.1}$$

On the other hand, depthwise separable convolutions are a more efficient variant of classical convolutions, which are converted to depthwise convolution followed by pointwise convolutions [HTY18]. This significantly reduces the computational complexity while maintaining the quality of the output. Depth convolutions perform spatial convolutions independently for each input channel, while pointwise convolutions combine the output of depth convolutions. A visual representation of the DeepLabv3+ architecture can be found in the figure 5.1.1.

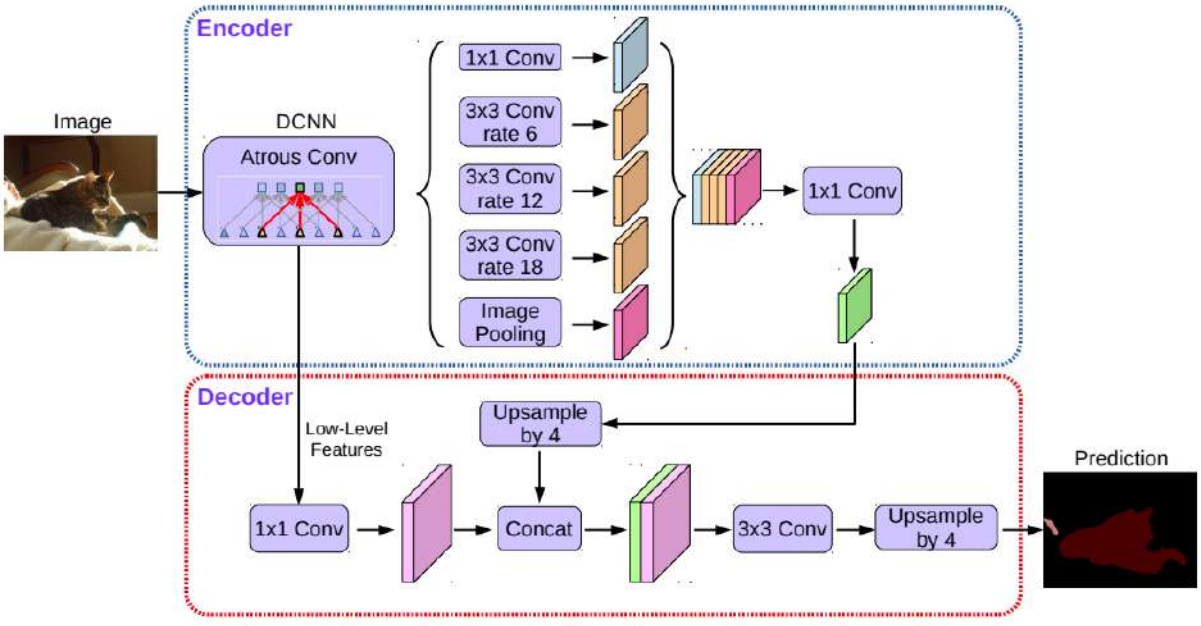

Figure 5.1.1: DeepLabv3+ is an extension of DeepLabv3 that uses an encoder-decoder structure. The encoder module is responsible for encoding context information at various scales using atrous convolution. On the other hand, the decoder mechanism is a simple but effective approach that improves segmentation results by improving object boundaries. [Che+18].

This model exploits separable Atrous convolutions, which are derived from background convolutions at rate $rate = 2$. They are shown to significantly reduce the computational complexity without compromising the quality of the output. The DeepLabv3+ model is used as an encoder in this approach for semantic segmentation. It uses atrous convolutions, which introduce gaps or "holes" between the filter and the input feature map, to extract features at a specific resolution computed by deep convolution neural networks. The

output step, which is the ratio of the spatial resolution of the input to the output, can be set to be 16 or even 8 for denser features. This can be done by removing the striding in the last or even the penultimate block and applying an inverse convolution. The last feature map before the logos is used as the output of the encoder, which includes 256 channels and rich semantic information. On the decoder side, the features from the encoder are first upgraded by a factor of 4 and merged with the corresponding low-level features from the network centerline that have the same spatial resolution. These low-level features are first passed through a convolution of 1×1 to reduce the number of channels, which would be quite large (e.g. 256 or 512) and may overwhelm the encoder features and make training difficult. After coalescence, several 3×3 coalescences are applied to improve the quality of the features and another upsampling with a factor of 4 is performed. It has been observed that the quality of the results is much better when using output stride = 8 instead of 16, although this sacrifices computational complexity. In this study, the DeepLabv3+ model used features from a pre-trained ResNet50 [He+15] network in ImageNet [Den+09] as a backbone.

5.2 HRNet

HRNet is a deep convolutional neural networking architecture designed for efficient fine-grained feature learning from high-resolution images. The basic idea behind HRNet is to maintain high spatial resolution throughout the network by using multiple parallel branches, each operating at different resolutions. This allows the network to efficiently exploit both high and low resolution information. Figure 5.2.1 illustrates the primary structure, but omits the stub (which consists of two stride-2 3×3 convolutions). The architecture consists of four stages, with the first stage involving high-resolution convolutions. The second, third and fourth stages include blocks that are repeated in two, three and four resolutions respectively. The HRNet

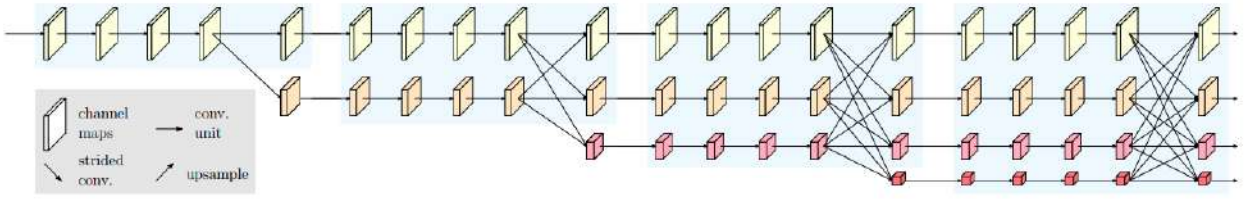

Figure 5.2.1: The HRNet architecture [Wan+19].

architecture consists of four main components: a stem module, multiple high-resolution (HR) blocks, multiple low-resolution (LR) blocks and a head module. The stem module is the first component of the HRNet architecture, it is responsible for reducing the resolution of the input image while maintaining high spatial resolution. The stem module consists of several convolutional resolution levels and maximum concentration levels that reduce the resolution by a factor of 4. This allows the network to extract more complex features and increase the number of channels in the feature maps.

HR blocks are the core of the HRNet network architecture and are responsible for maintaining high spatial resolution throughout the network. Each HR block consists of several convolution levels, beam normalization levels and ReLU activation levels. HR blocks are designed to efficiently utilize high-resolution information using large receptive fields and high-resolution feature maps. LR blocks are used to reduce the resolution of feature maps while maintaining the number of channels. Each LR block consists of several convolution levels, beam normalization levels and ReLU activation levels. LR blocks are designed to efficiently exploit low-resolution information using small receptive fields and low-resolution feature maps.

The header module is responsible for the final classification or prediction. It consists of several fully coupled layers and a softmax or sigmoid activation layer depending on the task. The header module receives the output of the HR and LR blocks and combines it to make the final prediction.

The HRNet architecture uses a multi-scale fusion mechanism that allows the network to efficiently combine information from different branches and make predictions. This mechanism is implemented by merging feature maps from different branches and then passing them through convolutional analysis layers to produce the final feature map. This final feature map contains rich high-resolution information, which is essential for tasks such as image classification, object detection and semantic segmentation.

HRNet is an architecture that uses multiple parallel branches, each of which operates at different resolutions, to efficiently exploit both high- and low-resolution information. The use of high-resolution and low-resolution blocks, along with a stem unit and a header unit, allows HRNet to maintain high spatial resolution across the network while reducing the resolution of feature maps. The multi-scale fusion mechanism allows HRNet to efficiently combine information from different branches, resulting in a final feature map that contains rich, high-resolution information. This makes HRNet a powerful architecture for visual recognition tasks.

5.3 HRNet+OCRNet+SegFix

HRNet + OCR [Yua+19] + SegFix [Yua+20] is an architecture designed for semantic segmentation tasks. It consists of three main components: HRNet, OCR and SegFix. The HRNet module is used to extract high-resolution feature maps. It is a multi-scale deep neural network that uses a pyramid of parallel convolution layers to extract features at different scales. The pyramid structure allows the network to capture both detailed and coarse aspects of the image, which is important for accurate segmentation. The use of parallel convolution layers allows the network to process the image at multiple scales simultaneously, improving the overall efficiency of the model. The HRNet component consists of an encoder-decoder structure where the encoder consists of multiple stages, where each stage contains many stacked residual blocks. Each stage increases the resolution of the feature maps and the decoder consists of several stages that reduce the resolution of the feature maps. The output of the HRNet segment is a set of high resolution feature maps that are fed to the next segment of the network. The OCR model is used to model the dependencies between objects. It is a transformer-based architecture that uses self-attention mechanisms to model the relationships between objects and their environment in the image. The self-attention mechanism allows the network to weight the importance of different features when making a prediction, which can improve the accuracy of segmentation results. OCR takes the output of the HRNet network as input and applies a series of transformation layers to model the relationships between objects. The transformer layers are composed of multi-head self-orientation networks, feed-forward networks and layer normalization. The output of the OCR segment is a set of object frame representations that are fed to the next stage of the network.

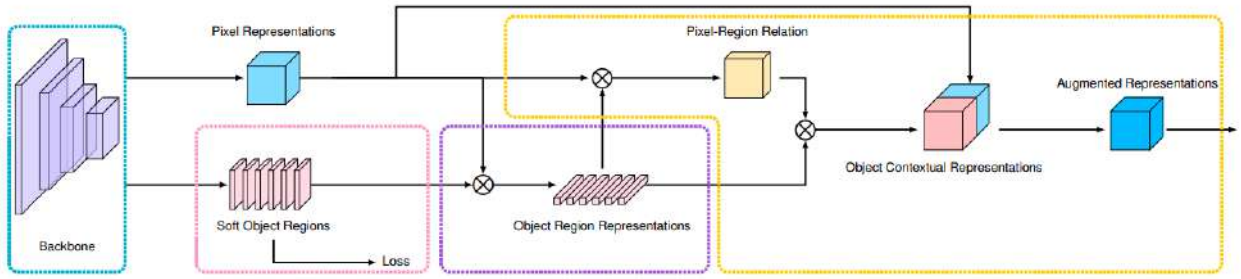

Figure 5.3.1: The pipeline architecture of OCR [Yua+19].

In the image 5.3.1 the pink dotted box creates the smooth areas of the objects within the boundary. (ii) The purple dashed box approximates the object region features and the orange dashed box computes the object frame features and enhanced features. The SegFix element is used to improve the final segmentation results. It is a fully convolutional network that accepts the output of HRNet and OCR as input and applies a series of convolutional and augmented sampling layers to produce the final segmentation map. SegFix allows the network to make detailed adjustments to the segmentation results, which can improve the overall accuracy of the model. SegFix also uses skip links from HRNet to incorporate high-resolution feature maps into the final forecast. The SegFix segment consists of several levels of convolutional analysis applied to the output of the OCR segment, followed by levels of upsampling to increase the resolution of the final forecast. The output of the SegFix component is a semantic segmentation map that assigns a label to each pixel in the input image. HRNet + OCR + SegFix is an architecture that combines the advantages of multiple neural network architectures to improve the accuracy of the final segmentation results. The HRNet segment is responsible for extracting high-resolution feature maps from the input image, the OCR segment is used to

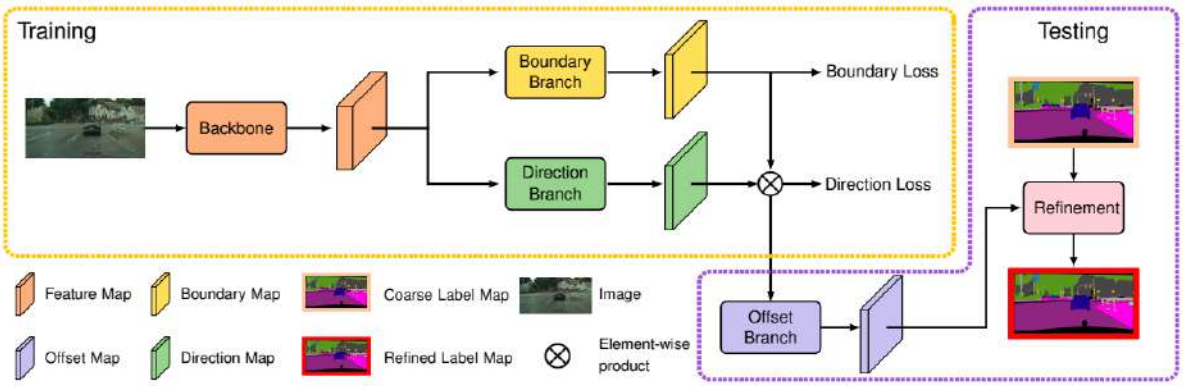

Figure 5.3.2: The pipeline architecture of SegFix [Yua+20].

model the relationships between objects in the image using transformer layers, and the SegFix segment takes the output of HRNet and OCR as input and improves the final segmentation results. The combination of these three components leads to an architecture that can accurately segment objects in an image while being computationally efficient. It is worth noting that this architecture is not the only one that combines different architectures and techniques to improve semantic segmentation, but it shows a new way of approaching the problem by using a transformer-based architecture, commonly used for natural language processing tasks.

5.4 DDRNet

The Deep Dual-resolution Networks (DDRNet) [Hon+21] architecture for accurate semantic segmentation of real-time street scenes consists of multiple interconnected modules. The DDRNet architecture is motivated by the observation that semantic road scene segmentation requires the network to learn both high- and low-resolution representations of the input image for accurate segmentation of objects of interest.

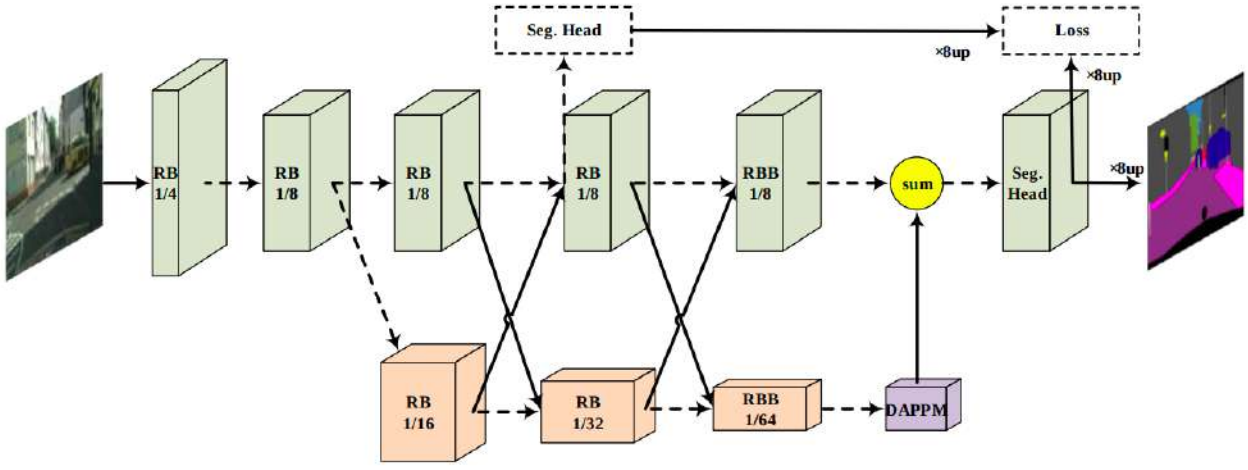

Figure 5.4.1: The term "RB" refers to consecutive residual basic blocks, the term "RBB" refers to a single residual congestion block, the term "DAPPM" refers to the Deep Aggregation Pyramid Pooling Module, and the term "Seg. Head" refers to the segmentation head. Black solid lines indicate information paths with data processing, including upgrading and downgrading, while black dashed lines indicate information paths without data processing. 'Sum' indicates point summation. Dotted boxes represent data not used in the inference stage. [Hon+21]

The first module of DDRNet is the feature extraction module, which is responsible for extracting features from the input image. The feature extraction module is implemented using a ResNet-101 architecture that

has been pre-trained in ImageNet. The ResNet-101 architecture consists of multiple residual blocks, each of which consists of two convolutional layers, beam normalization and a ReLU activation function. The purpose of the residue blocks is to learn the hierarchical features of the input image, which are then passed on to the next unit. Bilateral fusion is a technique used in the deep dense residue network (DDRNet) to fuse multi-scale features. In bilateral merging, low-level features from high-resolution feature maps are combined with high-level features from low-resolution feature maps using a weighted average.

The weights used in bilateral merging are based on two factors: spatial proximity and similarity of characteristics. Spatial proximity determines how close two pixels are to each other in the image, while feature similarity measures how similar the features are in those pixels. Weights are calculated as a combination of these two factors and are used to mix low- and high-level features. The bilateral fusion shown in the figure 5.4.2 is used in DDRNet to improve the accuracy of image segmentation and object detection tasks by preserving fine-scale details while maintaining a global perspective of the image. Therefore, the second module of DDRNet is the

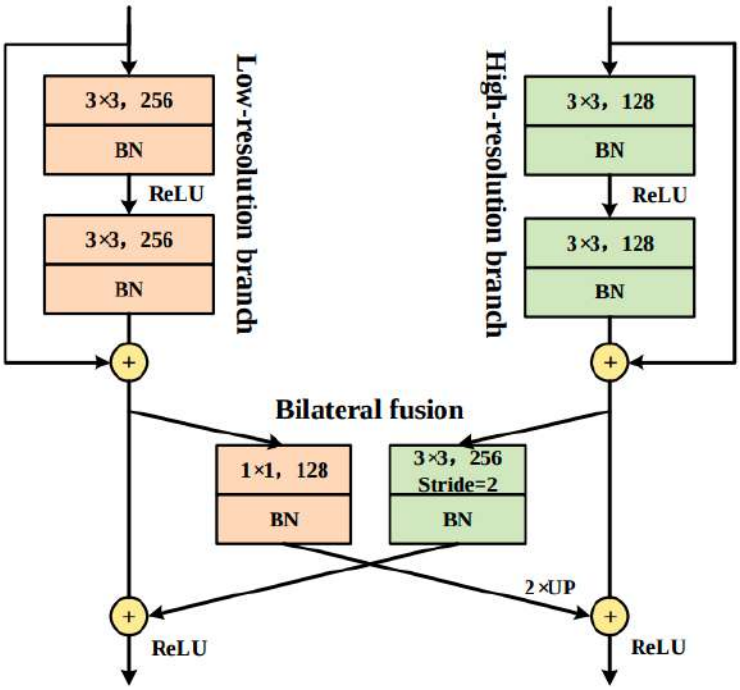

Figure 5.4.2: The details of the bilateral fusion of DDRNet. [Hon+21]

high-resolution feature fusion module, which is responsible for fusing the high-resolution features extracted from the feature extraction module with the low-resolution features. The high-resolution feature fusion module is implemented using a feature pyramid network (FPN) architecture. The FPN architecture consists of multiple top-down paths and lateral connections. The top-down paths are responsible for upsampling the low-resolution features to match the resolution of the high-resolution features, while the side links are responsible for merging the upsampled low-resolution features with the high-resolution features. The purpose of the high-resolution feature fusion module is to combine the high- and low-resolution features to improve the accuracy of semantic segmentation. The third module of DDRNet is the low-resolution feature fusion module, which is responsible for fusing the low-resolution features extracted by the feature extraction module with the high-resolution features. The low-resolution feature fusion module is implemented with a similar architecture as the high-resolution feature fusion module, but with the roles of the high- and low-resolution features reversed. The purpose of the low-resolution feature fusion module is to integrate the low-resolution features into the semantic partitioning to improve the real-time network performance. The fourth module of DDRNet is the multi-scale feature fusion module, which is responsible for fusing features at multiple scales. The multi-scale feature fusion module is implemented using a similar architecture as the high-resolution and low-resolution feature fusion modules, but with the addition of multiple scales. The purpose of the multi-scale feature fusion module is to incorporate features at multiple scales into the semantic partitioning to improve

the overall accuracy of the network. The fifth module of DDRNet is the semantic segmentation module, which

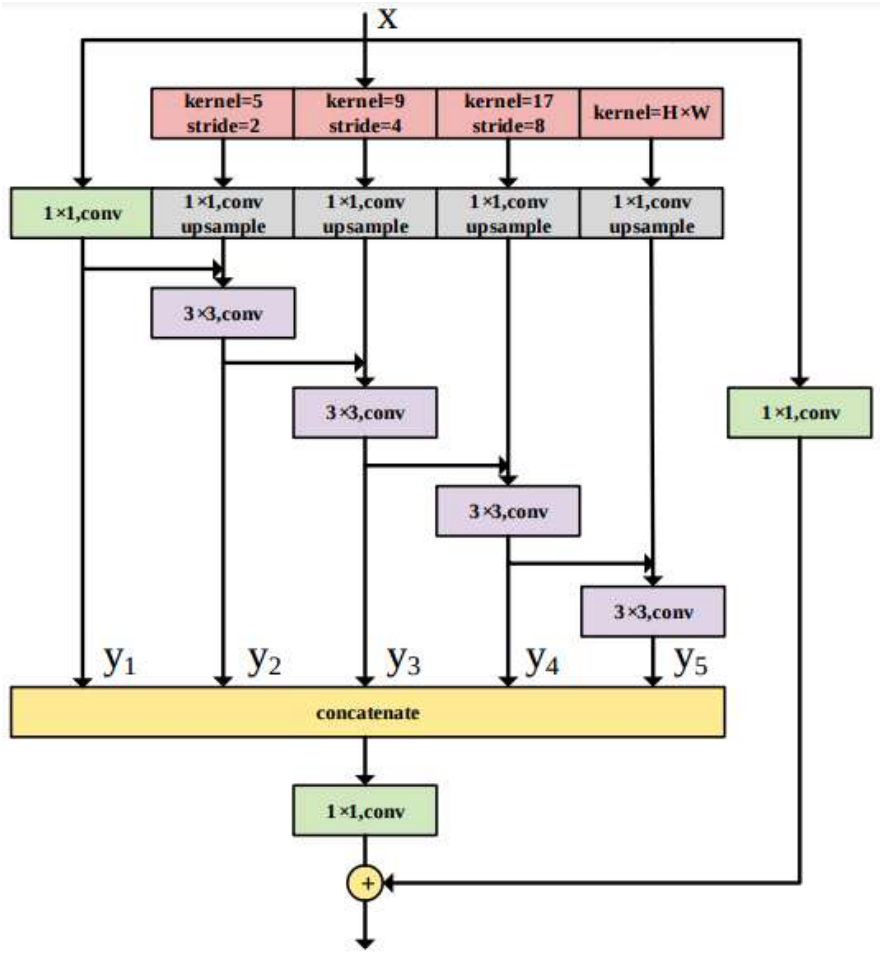

Figure 5.4.3: The detailed architecture of Deep Aggregation Pyramid Pooling. [Hon+21]

is responsible for performing semantic segmentation of the input image. The semantic segmentation module is implemented using a fully convolutional network (FCN) architecture. The FCN architecture consists of multiple convolutional network layers, bundle normalization and ReLU activation function. The purpose of the Semantic Segmentation module is to generate a probability map for each class in the input image. The DDRNet architecture for real-time and accurate semantic segmentation of street scenes consists of multiple interconnected modules. The feature extraction module is responsible for extracting features from the input image, the high-resolution feature extraction module. The feature extraction module is responsible for extracting features from the input image; the high-resolution feature fusion module is responsible for fusing the high- and low-resolution features; the low-resolution feature fusion module is responsible for fusing the low- and high-resolution features; the multi-scale feature fusion module is responsible for fusing the high- and low-resolution features. Still the low-resolution feature fusion module is responsible for fusing the low- and high-resolution features, the multi-scale feature fusion module is responsible for fusing the features at multiple scales, and the semantic segmentation module is responsible for performing the semantic segmentation of the input image. The architecture is designed to combine high- and low-resolution representations of the input image to improve accuracy while maintaining real-time performance. The use of residual blocks, FPN and FCN architectures within the DDRNet architecture allows the network to learn hierarchical, multiscale and semantic features of the input image, respectively, which are crucial to achieve accurate and efficient semantic segmentation of road scenes.

5.5 PIDNet

The architecture of Proportional-Integral-Derivative (PIDNet) [XXB22] networks for real-time semantic segmentation is based on an encoder-decoder structure. The encoder is responsible for extracting features from the input image and the decoder is responsible for generating the final segmentation. The encoder is implemented using a ResNet-101 [He+15] architecture that has been pre-trained on ImageNet and configured on the target dataset. The ResNet-101 architecture consists of multiple residual blocks, each consisting of two convolutional layers, beam normalization and a ReLU activation function. The purpose of the residue blocks is to learn the hierarchical features of the input image, which are then passed to the decoder. The decoder is responsible for creating the final partition from the features extracted by the encoder. The decoder is implemented using an Atrous Spatial Pyramid Pooling (ASPP) architecture. The ASPP architecture consists of multiple parallel branches, each of which consists of an atrous convolution layer and a global average pooling layer. The purpose of the ASPP architecture is to generate multi-scale features from the features extracted by the encoder. The multi-scale features are then upscaled to the original resolution of the input image using a shifted convolutional layer and fused with the features extracted by the encoder. The interwoven features are then passed to a final convolutional analysis layer to generate the final segmentation.

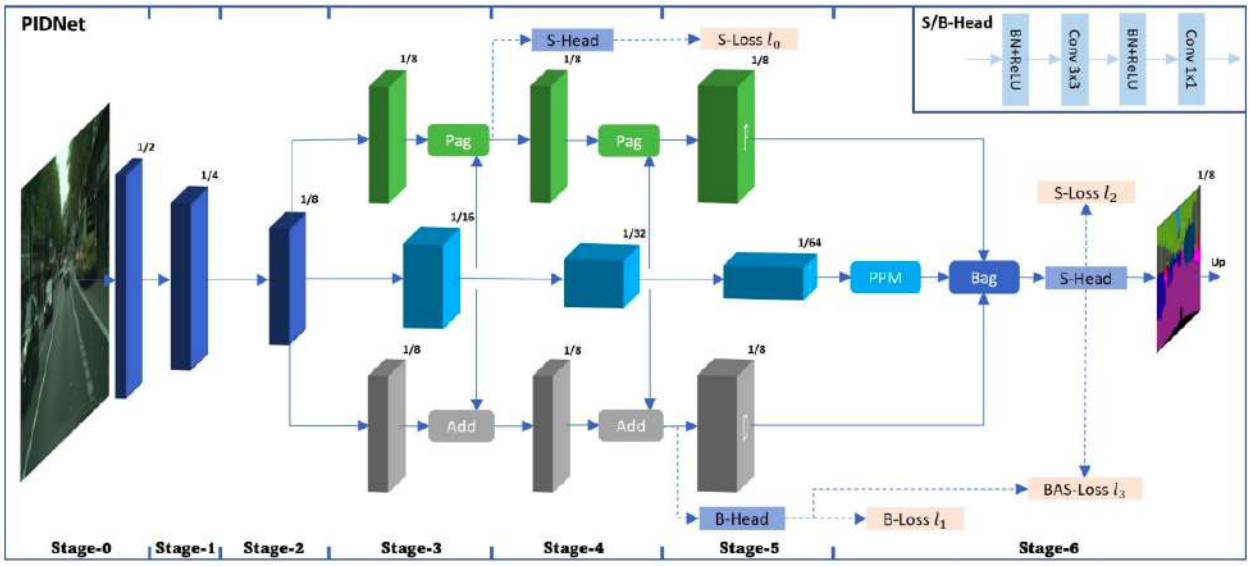

Figure 5.5.1: The PIDNet pipeline architecture. "S" and "B" denote the semantics and boundaries, while "Add" and "Up" refer to the element-wise summation and bilinear sampling upsampling functions respectively. In addition, "BAS-Loss" represents cross-entropy loss with boundary awareness. [XXB22]

The PIDNet architecture also includes a Proportional-Integral-Derivative (PID) loss function to improve accuracy and real-time performance. The PID loss function consists of three elements: a proportional loss element, an integral loss element, and a derivative loss element. The proportional loss component is responsible for minimizing the cross-entropy per pixel between the predicted segmentation and the ground truth segmentation. The Integral Loss component is responsible for minimizing the average cross-entropy between the predicted segmentation and the segmentation of the ground truth across the entire image. The Derivative Loss component is responsible for minimizing the gradient difference between the predicted segmentation and the ground truth segmentation. The purpose of the PID loss function is to balance the trade-off between accuracy and real-time performance by minimizing the Proportional Loss component for accurate segmentation, minimizing the Integral Loss component for real-time performance, and minimizing the Derivative Loss component to improve the smoothness of the predicted segmentation.

In addition, PIDNet introduces a new tuning mechanism to adjust the trade-off between accuracy and performance in real-time by adjusting the weight of the Integral Loss component. This mechanism allows to control whether real-time performance takes priority over segmentation accuracy. By adjusting the weight of the Integral Loss component, the network can be fine-tuned to achieve a balance between accuracy and real-time performance that is appropriate for a particular application or dataset.

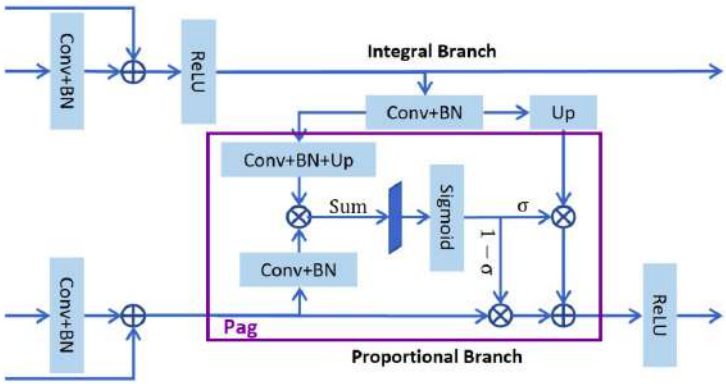

Figure 5.5.2: The Pag module is in lateral connection, in which "Sum" refers to summation by element along the channels, "σ" represents the output of the sigmoid function, and "Up" indicates bilinear upscaling. [XXB22]

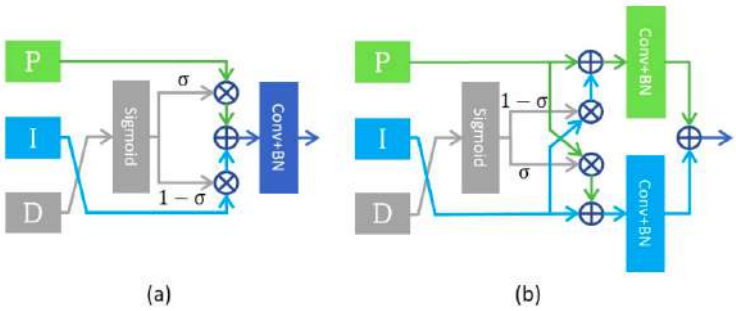

Figure 5.5.3: The figures show the Bag (a) and Light-Bag (b) sections, where "P", "I" and "D" denote the outputs of the detailed, frame and boundary branches respectively. [XXB22]

The use of the ResNet-101 encoder, ASPP decoder, PID loss function and resonance mechanism in the PIDNet architecture allows the network to learn hierarchical, multiscale and semantic features of the input image, balancing the trade-off between accuracy and real-time performance.

These segments work together to achieve accurate and efficient semantic segmentation, making PIDNet suitable for real-time applications such as autonomous driving, robotics and surveillance.

5.6 Swin Transformer

The architecture of the Swin Transformer model [Liu+21] for image recognition and segmentation is based on the Transformer Vision (ViT) [Dos+20] architecture, with the addition of shifted windows to capture both local and general context information. Shifted windows are used to divide the input image into smaller regions, called patches, which are then processed by the transformer layers. The layers of the transformer consist of multi-head self-attention layers as well as feed-forward layers. Multi-head self-attention layers are responsible for capturing the dependencies between surfaces in the input image. Each patch vector is converted into a query, key and value vector using linear projection. The self-attention mechanism is performed by computing the square product of the query vector with the key vector, and the square product is then passed through a softmax function to obtain the attention weights. The attention weights are then used to weight the value vectors and summed to obtain the output of the self-attention layer. This process is done for each head of the multi-head self-attention layer and the outputs from each head are combined and passed through a linear projection to obtain the final output of the multi-head self-attention layer. The feed-forward layers are responsible for performing the nonlinear transformations on the surface vectors. The feedforward layer consists of two linear projections with a ReLU activation function in between. The

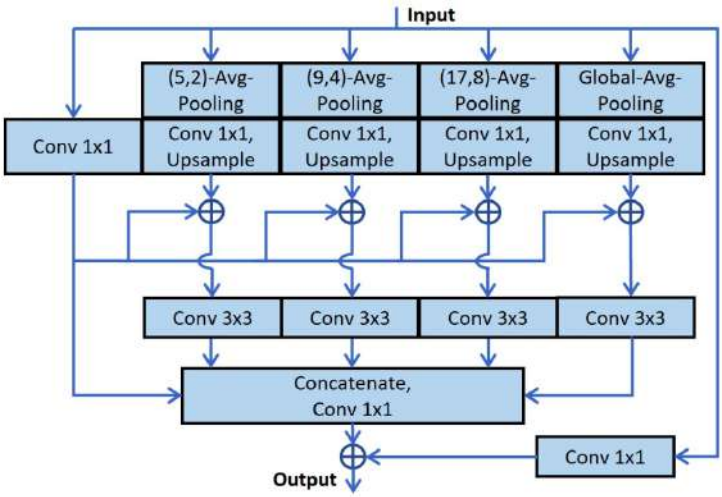

Figure 5.5.4: Illustration of the parallel structure of PAPPM. "(5,2)-Avg-Pooling" indicates the use of average pooling with a core size of 5x5 and strides of 2. "Bilinear upsampling" is used for upsample operations. [XXB22]

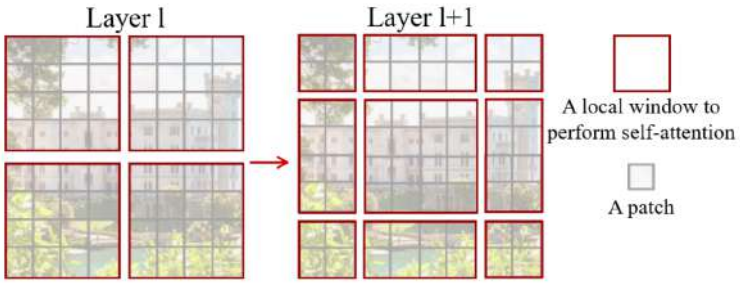

Figure 5.6.1: The shifted window method used to calculate self-attention in the Swin Transformer architecture. In layer "l" (on the left), a standard window partitioning method is used and self-attention is calculated within each window. However, in the next layer, "l+1" (on the right), the window partitioning is shifted, creating new windows. This has the effect of calculating the self-attention in the new windows that extend beyond the boundaries of the previous windows in layer "l", creating connections between them. [Liu+21]

purpose of the propulsion-promotion layer is to create a nonlinear representation of the input surface vectors. Transformer layers are designed to learn the general and local information of the image by capturing the dependencies between patches in the input image and performing nonlinear transformations on the patches. The Swin transformer architecture also includes a new training method called random shifting. The random shift method is used to avoid overfitting and improve generalization. The method involves randomly shifting the input image before extracting the patches. This causes the network to learn the invariance of the position of objects in the image, which improves the generalization of the network. The authors also proposed a new version of Swin Transformer, called Swin-T, which is designed to be used for image segmentation. Swin-T is an extension of the original Swin Transformer architecture and is designed to include a segmentation head. The segmentation head consists of a convolutional layer and a softmax layer. The convolutional layer is responsible for generating a feature map from the output of the transformer layers and the softmax layer is responsible for generating a probability map for each class in the input image.

The feature map is then upscaled to the original resolution of the input image and the probability map is used to generate the final segmentation. The proposed Swin-Transformer architecture consists of several key components, such as patch extraction, transformation layers, random shift and a segmentation head in the case of the Swin-T variant. These elements work together to efficiently capture both local and general context information in the input image, which is crucial for accurate image recognition and segmentation.

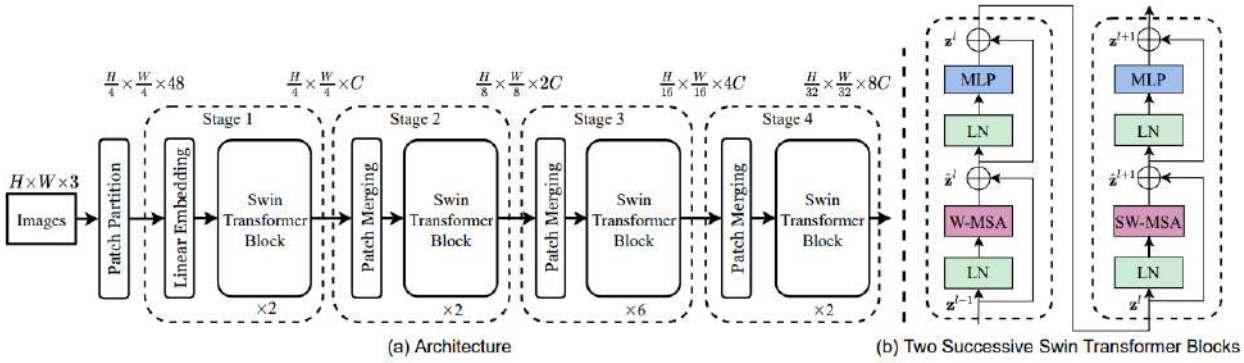

Figure 5.6.2: (a) The architecture of a transformer Swin (Swin-T), (b) two successive blocks of Transformer Swin. W-MSA and SW-MSA are multi-head self-attention modules [Liu+21].

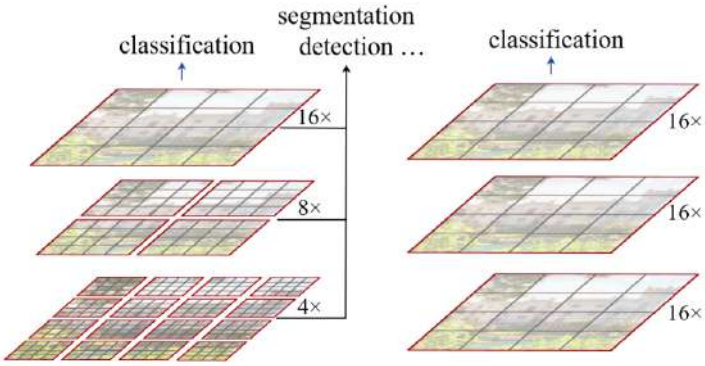

Figure 5.6.3: The first diagram illustrates the architecture of the Swin transformer, which combines image patches (in grey) to create hierarchical feature maps in deeper layers. It has linear computational complexity with respect to the size of the input image, since it only computes the self-attention within each local window (shown in red). On the other hand, ViT [Dos+20] produces feature maps of only one low resolution and has quadratic computational complexity with respect to the size of the input image due to the overall computation of self-attention. [Liu+21]

The shifted windows used in the patch extraction process allow the network to learn both local and general context information. Transformer layers are designed to learn the general and local image information by capturing the dependencies between patches in the input image and performing nonlinear transformations on the patches. The multi-head self-orientation layers are responsible for capturing the dependencies between patches in the input image, and the forwarding layers are responsible for performing nonlinear transformations on the patches. The random displacement method is used to avoid overfitting and improve generalization by forcing the network to learn the invariance of the position of objects in the image. The segmentation head in the Swin-T variant includes a convolutional analysis layer and a softmax layer, which produce a feature map and a probability map respectively, which are used to generate the final segmentation.

5.7 BASNet

The architecture of the BASNet [Qin+19] model for detecting distinct objects is based on training on a U-shaped structure. The U-shaped structure of the network consists of an encoder and a decoder. The encoder is responsible for extracting features from the input image, while the decoder is responsible for generating the feature map.

The network encoder consists of multiple convolutional layers, which are responsible for extracting features from the input image. The convolutional layers are designed to learn the hierarchical representations of the input image. The convolutional layers are implemented using the ResNet-101 architecture, which is a widely

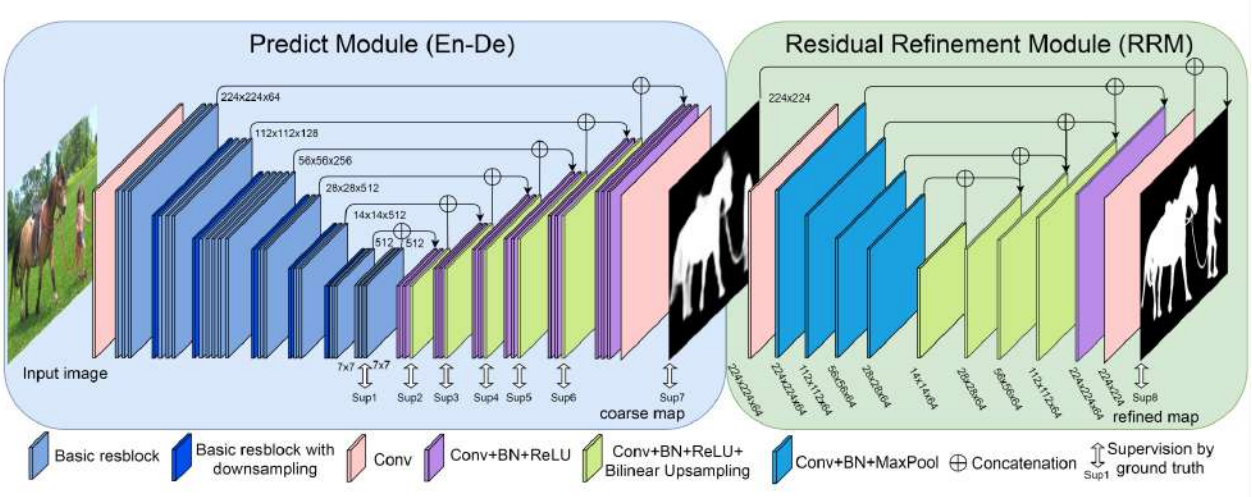

Figure 5.7.1: Architecture of BASNet [Qin+19].

used deep convolutional neural network for image classification. The ResNet-101 architecture consists of a set of residual blocks, each of which consists of two 3×3 convolutional layers with step 2 and a shortcut connection. The shortcut connection is used to reduce the number of parameters in the network and improve the gradient flow through the network. The feature maps generated by the convolutional network layers are then passed through a boundary enhancement module, which is responsible for enhancing the boundary information of the feature maps. The boundary enhancement module consists of several layers of convolutional convolution with dilated convolution and atrous convolution. Dilated convolution and atrous convolution are used to increase the receptive field of the convolution layers and capture the fine details of the input image.

The network decoder is responsible for generating the resolution map from the feature maps generated by the encoder. The decoder consists of multiple layers of convolutional resolution, which are responsible for upgrading the feature maps to the original resolution of the input image. The resampled feature maps are then passed through a boundary enhancement module, which is responsible for enhancing the boundary information of the resampled feature maps.

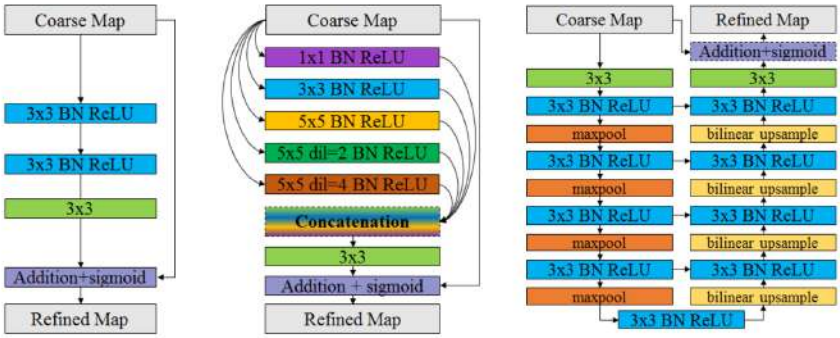

Figure 5.7.2: Illustration of the different Residual Refine units (RRM). In the third, a multi-scale RRM MS refining unit. In the first, the first one is a BasNet RRM encoder-decoder refinement unit. [Qin+19].

The boundary enhancement module also consists of several levels of dilated convolution and atrous convolution, similar to the boundary enhancement module in the encoder. This allows the network to refine the boundary information and generate a more accurate saliency map. In addition to the network architecture, the authors also proposed a new loss function called boundary-aware loss function to improve the detection of object boundaries. The loss function is designed to penalize the network for misclassifying pixels near object boundaries. The loss function consists of two parts: binary cross-entropy loss and boundary loss. The

binary cross-entropy loss is used to measure the difference between the predicted salinity map and the ground truth salinity map. Boundary loss is used to measure the difference between the predicted salinity map and the ground-truth salinity map near the subject boundary. Boundary loss is calculated by convolving the ground-truth salinity map with a Sobel operator to calculate the slope of the ground-truth salinity map. The Sobel operator is used to calculate the slope of the saliency ground-truth map and is used to identify pixels near the boundaries of the object. To achieve high quality regional segmentation and clear boundaries, the authors propose a hybrid loss:

$$l^{(k)} = l_{bce}^{(k)} + l_{ssim}^{(k)} + l_{iou}^{(k)} \tag{5.7.1}$$

where, $l_{bce}^{(k)}$, $l_{ssim}^{(k)}$ and $l_{iou}^{(k)}$ represent the BCE, SSIM and IoU losses, respectively.

The authors also proposed a new version of the network, BASNet-v2, which includes some improvements to the original architecture to improve the accuracy and performance of the network. The improvements in BASNet-v2 include a deeper encoder, a boundary-aware attention module, and a boundary-aware decoder. The deeper encoder consists of additional convolutional layers with ResNet-101 architecture, which allow the network to extract more detailed features from the input image. The boundary-aware attention module is used to adjust the importance of different regions of the feature maps, which helps the network to better identify the boundaries of objects. The boundary-aware decoder is designed to improve the boundary information of the feature maps that have been upgraded in a more efficient way. By implementing these changes, the accuracy and performance of the network are improved, allowing for more accurate and efficient detection of distinct objects. It is worth noting that, the proposed BASNet architecture is a deep edge-to-edge neural network, which means that the network is trained to learn both feature extraction and salinity map generation simultaneously. This allows the network to learn a more complete and robust representation of the input image. The U-shaped architecture is designed to take advantage of both shallow and deep networks. The encoder consists of deep layers, which are responsible for extracting features from the input image, while the decoder consists of shallow layers, which are responsible for generating the salinity map. This allows the network to extract both high and low level features from the input image, which is crucial for accurate detection of objects of prominent importance. In addition, the use of dilated convolution and atrous convolution in the boundary enhancement and boundary refinement modules allows the network to increase the receptive field of the convolution levels and capture the fine details of the input image. This helps the network more accurately identify object boundaries. The boundary-aware loss function is used to penalize the network for misclassifying pixels near object boundaries, which helps the network learn to recognize object boundaries more accurately. In conclusion, the proposed BASNet architecture is a robust and efficient solution for detecting distinct objects. Using a U-shaped architecture, ResNet-101 encoder, boundary enhancement and boundary refinement units, boundary-aware loss function, and the proposed tuning mechanism in the architecture enable the network to effectively capture and refine the boundary information of the input image, which are vital for accurate detection of distinct objects. The proposed BASNet-v2 version also includes some improvements to the original architecture to improve the accuracy and performance of the network.

5.8 BISNet

The BiSeNet [Yu+18] architecture is composed of two main paths: the spatial path and the frame path. The spatial path is responsible for extracting detailed information from the entry image, while the frame path is responsible for extracting framework information from the entry image. The two routes are then combined to create the final partition map. The spatial path consists of various levels of coherent analysis, which are responsible for extracting features from the input image. Conclusive layers are implemented using ResNet-101 architecture, which is a widely used deep coherent neural network for image classification. ResNet-101 architecture consists of various residual blocks, each consisting of two 3×3 total layers with step 2 and a shortcut connection. The shortcut connection is used to reduce the number of parameters in the network and improve the slope flow through the network. The characteristics maps produced by the collegiate network layers then pass through a spatial pyramid aggregation unit (SPP), which is responsible for reducing spatial analysis of the attributes maps and increasing the receptive field of the aggregate network layers. The framework path is responsible for extracting framework information from the input image. The frame path consists of various spatial pyramid aggregation modules atrous (ASPP), which are responsible

for capturing the frame information of the entry image on different scales. The ASPP modules consist of various condensation levels with dilated condensation and abrasive assembly. The dilated condensation and adroid condensation are used to increase the receptive field of the condensation levels and capture the frame information of the input image on different scales. The two roads then combine to create the final partition map. The characteristics maps produced by the spatial path and frame path are joined together and passed through a layer of coherent processing, which is responsible for the creation of the final partition map. The final partition map then passes through a softmax layer, which is responsible for converting the network output into a probability map. In addition to network architecture, authors also proposed a

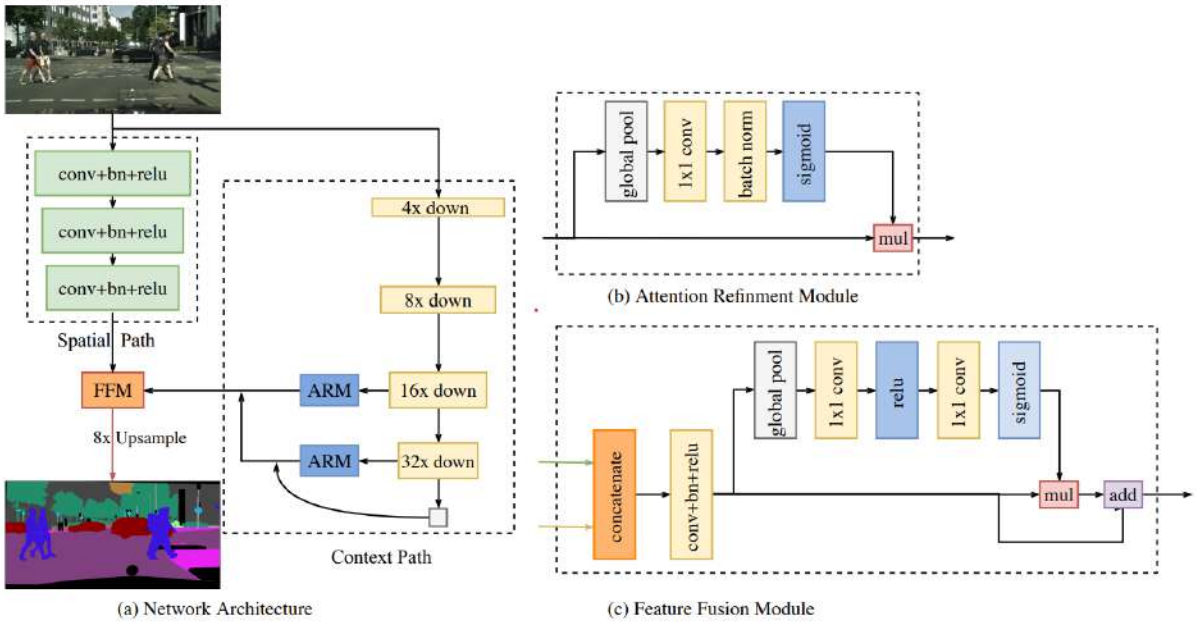

Figure 5.8.1: The Bilateral Segregation Network is presented with an overview covering (a) the network architecture, where the block size reflects the spatial size and thickness of the number of channels. In addition, the components of both the (b) Attention Refinement Module (ARM) and the (c) Feature Merge Unit (Feature Fusion Module, FFM) are described. It is important to note that the process represented by the "red line" is used exclusively during the tests. [Yu+18].

new loss function called pyramid concentration unit to improve network accuracy. The pyramid assembly unit is used to integrate global framework information into the network. The pyramid concentration unit consists of various pyramid concentration levels, each responsible for capturing global framework information at different scales. The pyramid concentration layers are implemented using the mean concentration used to reduce spatial analysis of the characteristics maps and increase the receptive field of the concentric layers. The proposed loss function:

$$Loss = \frac{1}{N}\sum_{i=1}^{N} L_i = \frac{1}{N}\sum_{i=1}^{N} -\sum_{j=1}^{M}[y_j^{(i)} \cdot log(\frac{e^{p_j}}{\sum_{j'=1}^{M} e^{p_{j'}}}) + (1 - y_j^{(i)}) \cdot log(1 - \frac{e^{p_j}}{\sum_{j'=1}^{M} e^{p_{j'}}})] \tag{5.8.1}$$

In the above equation, N is the total number of training samples, M is the number of output pixels in each sample, $y_j^{(i)}$ is the basic truth value of the $j-est$ pixels in the $i-st$ sample, $p_j^{(i)}$ is the predicted probability of the $j-th$ pixels in the $i-th$ sample, w_{ji}) is the weight of the $j-th$ pixels in the $i-th$ sample and Θ represents the total of all parameters of the model. is a normalization parameter that helps avoid over-adjustment.

To further improve the accuracy and effectiveness of the network, authors also proposed a new method of education called multiple-scale training. Multiple-scale training is used to train the network at different scales to improve network generalisation capacity. Multiple-scale training is carried out by accidental re-size of the input image during training. The BiSeNet architecture is a powerful and effective solution for

semantic segmentation in real time. The use of an architecture of two paths, the ResNet-101 encoder, the ASPP decoder, the SPP concentration unit and pyramid and the multi-scale training method in architecture allows the network to capture and integrate effectively detailed and contextual information from the input image. Architecture is designed to achieve a balance between accuracy and efficiency, which makes it suitable for applications in the real world. The proposed pyramid concentration unit in the loss function also improves network accuracy by incorporating global framework information. The authors of the publication also conducted extensive experiments in various reference data sets, which show that the proposed BiSeNet architecture outweighs modern methods in both accuracy and efficiency. It is worth mentioning that BiSeNet is a deep neural network from edge to end, meaning that the network is trained to learn simultaneously both the extraction of features and semantic segmentation. This allows the network to learn a more complete and powerful representation of the input image. Two paths architecture is designed to exploit the advantages of both shallow and deep networks. The spatial path consists of deep layers, which are responsible for extracting detailed information from the entrance image, while the frame path consists of shallow layers, which are responsible for extracting framework information from the entry image. This allows the network to extract both high and low-level characteristics from the input image, which are vital for accurate semantic segmentation.

5.9 UNET3+

The UNet 3+ [Hua+20] architecture is a modified version of popular UNet architecture, designed for medical image segmentation work. The UNet architecture consists of two main elements: a coder and a decoder. The encoder is responsible for extracting features from the input image, while the decoder is responsible for creating the final partition map. The UNet 3+ architecture improves UNet's performance by incorporating a full-scale connection mechanism, which connects the encoder and decoder features maps to multiple scales. In addition, the UNet 3+ architecture also incorporates a new attention mechanism called Multi-Attention Blocks (MABs). The UNet 3+ architecture encoder consists of multiple levels of coherent processing, which are responsible for extracting features from the input image. Conclusive layers are implemented using ResNet-101 architecture, which is a widely used deep coherent neural network for image classification. ResNet-101 architecture consists of various residual blocks, each consisting of two 3×3 total layers with step 2 and a shortcut connection. Shortcut connection is used to reduce the number of parameters in the network and improve the slope flow through the network. The characteristics maps produced by the coherent network layers then pass through a spatial pyramid aggregation unit (SPP), which is responsible for reducing spatial analysis of the characteristics maps and increasing the receptive field of the concentric network layers. The decoder of the UNet 3+ architecture is responsible for creating the final segmentation map. The decoder consists of multiple levels of coherent analysis with upsampling, which are responsible for increasing spatial analysis of characteristics maps. The decoder also includes a new mechanism called full-scale connection (full-scale connection), which connects the encoder and decoder features maps on multiple scales. The full-scale connection mechanism allows the decoder to access the encoder characteristics on different scales, which improves network performance.

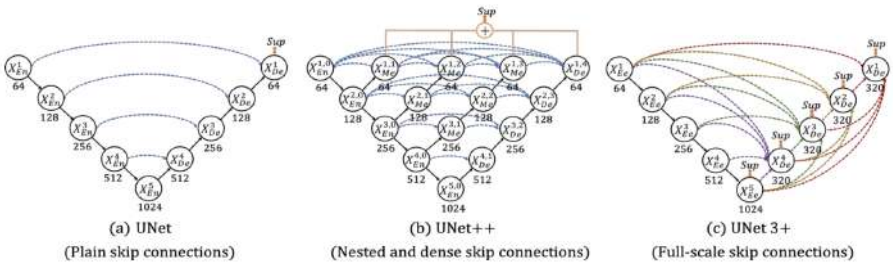

Figure 5.9.1: Comparison of UNet (a), UNet++(b) and UNet 3+ (c) [Hua+20]

The UNet 3+ architecture also incorporates another attention mechanism called Multi-Attention Blocks (MABs). MABs are responsible for learning more powerful and distinctive features than the entry image. MABs consist of various levels of conjunctive interconnection at different expansion rates, which are used to capture the input image frame information on different scales. In addition, MABs also include a caution

mechanism, which is used to weigh the different characteristics maps produced by the condensation levels. The attention mechanism allows the network to focus on the most important features for the work of semantic partitioning. MABs are integrated into the encoder and decoder of UNet 3+ architecture, which allows the network to learn more powerful and distinctive features than the input image. In addition to network architecture, the authors also proposed a new loss function called the dice loss, which is used to measure the difference between the predicted saliency map and the saliency map of the actual value. The loss of Dice is a popular loss function for semantic partition operations, which is used to measure the similarity between two data sets. The loss of Dice is calculated as the harmonic average of the intersection on the compound (IoU) between the predicted saliency map and the saliency ground-truth map. The loss of Dice is sensitive to the problem of imbalance of classes, which is a common problem in semantic partition work. To further improve the accuracy and effectiveness of the network, the authors also proposed a new method of education called multi-scale training. Multiple-scale training is used to train the network at different scales to improve network generalisation capacity. Multiple-scale training is carried out by accidental resize of the input image during training. The UNet 3+ architecture is a powerful and effective solution for medical image segmentation work. The authors of the work proposed a new design of network architecture called Full-Scale Connected UNet, which links the map features of the encoder and decoder to multiple scales, which improves network performance. In addition, writers also suggested a different attention mechanism called Multi-Attention Blocks (MABs), which allows the network to learn more powerful and distinctive features. The UNet 3+ architecture also incorporates a new loss function called dice loss and a new training method called multi-scale training, which further improves network accuracy and efficiency. The authors of the study also conducted extensive experiments in various sets of medical image data, which indicate that the proposed UNet 3+ architecture outweighs modern methods both in terms of accuracy and efficiency.

5.10 Attention U-Net

Attention U-Net (AU-Net) [Okt+18] is a deep learning model for semantic image partitioning. It is an extension of popular U-Net architecture, which combines a coder network with a decoder network to create a pixel segmentation mask of an input image. AU-Net incorporates a mechanism of attention to U-Net architecture to enhance its ability to capture long-range dependencies between pixels and improve the accuracy of partition.

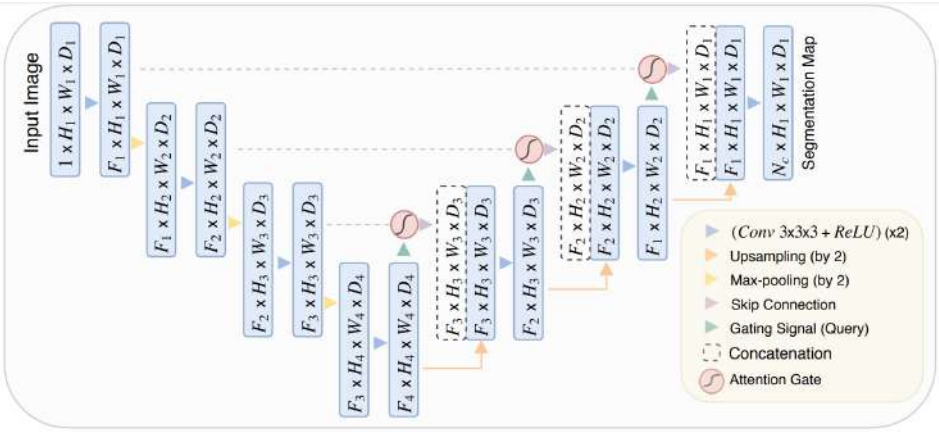

Figure 5.10.1: The proposed U-Net segmentation model is depicted as a block diagram. The input image is processed through the network encoding section, where it is progressively filtered and degraded by one factor 2 on each scale. For example, H4 represents a degraded version of H1 by a factor 8. The Nc parameter indicates the number of classes. The features disseminated through the bypass connections are filtered by attention gates (Attension Gates, AGs). [Okt+18].

U-Net architecture consists of a shrinking path, which is the encoder network, and an expanding path, which is the decoder network. The constrictive path consists of various levels of concentric analysis, each followed by a maximum concentration level, which reduces the spatial dimensions of the characteristic map. The expansion path consists of various deconsymbolic layers, each followed by a combination with the corresponding

characteristic map from the constrictive path. U-Net architecture also uses omission links between the contraction and expansion pathways, allowing the decoding network to access the lower level features exported from the coding network.

AU-Net architecture expands U-Net architecture by incorporating a mechanism of attention to the network's procurement course. The attention mechanism selectively weighs the importance of each feature map in different spatial positions, allowing the network to focus on the most important features for segmentation work. The attention mechanism consists of two elements: a channel attention unit and a spatial attention unit. The channel attention unit (channel attention module) calculates a set of channel attention maps for each feature map on the contribution path. Each caution map is obtained by passing the characteristic map through two fully connected layers, followed by a sigmoid activation function. The sigmoid function escalates the output of fully connected layers to the range [0, 1], representing the importance of each channel on the feature map. Attention maps per channel are then multiplied with the original feature maps to produce the weighted characteristic maps. The spatial attention module calculates a set of spatial attention maps for each weighted map of characteristics. Each spatial attention map shall be obtained by passing the weighted characteristic map through two condensation levels, followed by a softmax activation function. The softmax function escalates the output of the concentric activation layers to the range [0, 1], representing the importance of each spatial position on the characteristics map. Spatial attention maps are then multiplied by weighted characteristic maps to produce the characteristics guided by attention.

Attention feature maps (attention-guided feature maps) then pass through the other levels of condensation in the contraction path to produce the compressed representation of the input image. Compressed representation retains the most relevant information about subsequent tasks. The extension path of AU-Net architecture is similar to U-Net architecture, with the addition of attention portals. The attention gates (attention gates) are inserted between the omission connections and the corresponding decompression layers. Attention portals selectively weigh feature maps from the shrinking path based on the attention maps calculated by the attention mechanism on the shrinking path. Attention portals ensure that the decoding network focuses on the most important features for segmentation work and suppresses unrelated features.

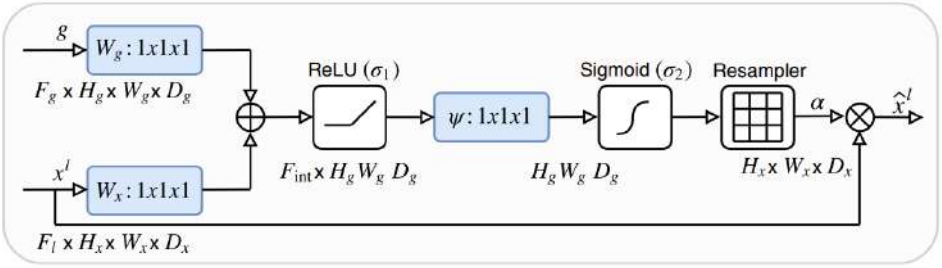

Figure 5.10.2: Schematic of the additive attention gate (AG). [Okt+18]

.

The exit of the last layer on the expansion path is a pixel-wise segregation mask, where each pixel is assigned a possibility of belonging to a particular class or object. The segmentation mask is obtained by passing the characteristic map through a layer of coherent analysis followed by a sigmoid activation function. AU-Net architecture has several advantages over U-Net architecture. The attention mechanism improves the network's ability to capture long-range dependencies between pixels and enhances the accuracy of partition. The mechanism of caution also reduces the number of parameters in the network, improving the effectiveness of the training process. The attention mechanism can also be applied to other types of deep learning architectures, such as ResNet and DefenseNet architectures, to improve their performance. The decoder element of the Attention U-Net model is similar to the standard U-Net architecture. It includes a number of decontributing layers, which are also known as layers of upward confluence, and are responsible for restoring the spatial dimensions of the characteristics maps. Up-convolutional layerers perform the reverse function of convolutional layerers by increasing spatial analysis of feature maps. However, the U-Net Attention model includes an additional element called the attention mechanism. The attention mechanism is a unit that selectively focuses on the most relevant areas of the input image and on the characteristics maps produced by the encoder. This mechanism allows the decoder to adaptally monitor the most informative areas of the

input image, improving the quality of the output segmentation mask. The attention mechanism consists of two main elements: the query network and the key price network. The query network produces a set of questions used to monitor the relevant areas of key-price charts. The core value network produces the core value pairs used to represent the characteristics maps.

The mechanism of attention works in two phases. In the first phase, the query network produces a set of questions used to monitor the most enlightening areas of key-price features maps. The questions are generated by applying a set of coherent layers to the decoder characteristics maps. The output of the coherent layers then passes through a softmax function to create the attention map. The caution map highlights the most relevant areas of feature maps that are most informative for the current decoding step. In the second phase, the caution map is used to calculate a weighted sum of key-value pairs. Key-price pairs are created by applying a set of condensing levels to the encoder's feature maps. The output of the collegiate analysis layers is then divided into two parts: key and value. The key is used to calculate the similarity between the questions and key-price pairs, while the price is used to create the weighted sum of key-price pairs. The caution mechanism is then used to merge the decoder characteristics maps with the weighted sum of key-value pairs. This merge function allows the decoder to focus on the most informative areas of characteristic maps, improving the quality of the segmentation mask.

One of the main advantages of the Attention U-Net model is its ability to capture detailed details and complex structures in the input image. The attention mechanism allows the decoder to adaptally monitor the most informative areas of the features maps, improving the quality of the output segmentation mask. Another advantage of the U-Net Attention model is its ability to handle inputs of different sizes. Since the model is implemented as a fully coherent neural network, it can process images of any size, making it suitable for tasks such as partitioning images where input images may have different dimensions. The attention mechanism is a critical element of the model that selectively focuses on the most relevant areas of the input image and on the characteristics maps produced by the encoder, improving the quality of the output segmentation mask. The ability of the Attention U-Net model to record detailed details and handle inputs of different sizes makes it a promising architecture for future research in a wide range of image segmentation work in both medical and non-medical fields.

5.11 EMANet

The Expectation-Maximization Attention Network (EMANet) [Li+19] is a deep learning model designed for semantic segmentation work. EMANet incorporates the Adaptation-Maximization (EM) algorithm and the self-awareness mechanism to improve the accuracy of semantic segmentation. EMANet consists of two main elements: the characteristics export network and the EM self-care network. The characteristics export network is responsible for extracting high-level features from the input image, while the EM-watch network is responsible for improving the segmentation results by integrating the EM algorithm and the self-awareness mechanism.

The features export network is usually a coherent neural network (CNN) that has been pre-trained in a large set of data such as ImageNet. The input image passes through the characteristics export network, which exports a feature map. The feature map contains information on the spatial distribution of the characteristics in the image. The EM-aware network is a new element that incorporates the EM algorithm and the self-awareness mechanism. The EM algorithm is a powerful tool for modeling data with hidden variables. It consists of two steps: step E and step M.In step E, the algorithm assesses the ex post distribution of hidden variable data of the observed data. In step M, the algorithm shall update the model parameters based on the estimated ex-post distribution.

The EM-Caution network consists of two branches: the care sector and the segmentation industry. The care branch calculates the attention weights based on the characteristic map using the self-aware mechanism. The segmentation industry uses attention weights to improve segmentation results by incorporating the EM algorithm. In the caution industry, the feature map first passes through a layer of 1x1 concentric analysis to reduce the number of channels. The resulting feature map is then used to calculate attention weights. Attention weights are calculated by first applying a softmax function to the feature map to generate a probability map. The probability map is then used to calculate attention weights by applying another layer of convergences. Attention weights are then used to calculate the weighted sum of the characteristic map,

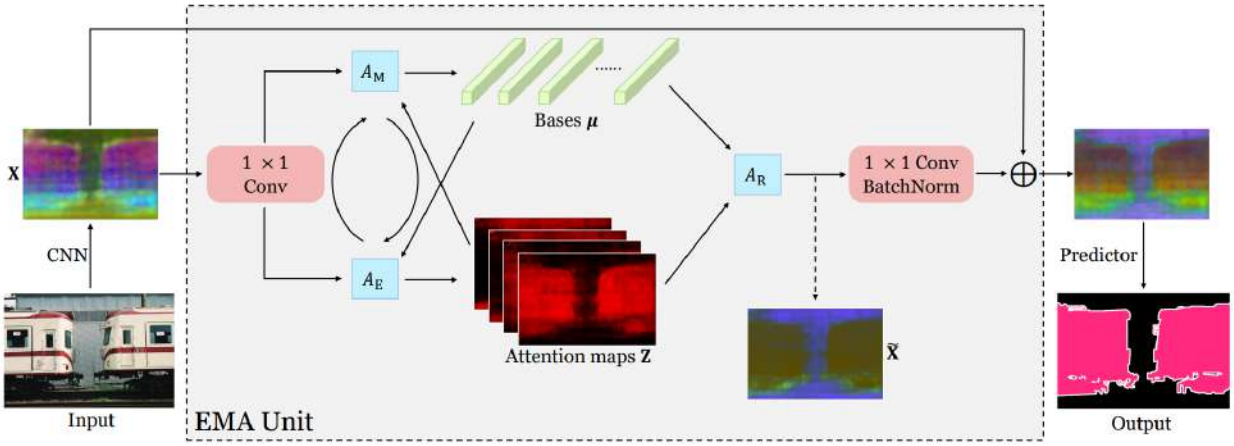

Figure 5.11.1: The proposed EMA mechanism has an overall structure consisting of the EMA mechanism as its key element. The EMA mechanism includes alternate executions of AUs and AMs. To improve its performance, two conjunctions 1x1 are included at the beginning and end of EMA and the output is combined with the initial entry to create a residue-like block. [Li+19]

which is the caution map. In the segmentation industry, the caution map first passes through a 1×1 collegiate analysis layer to reduce the number of channels. The resulting feature map is then combined with the feature map from the characteristics export network. The interwoven feature map is then used to calculate the segmentation results using the EM algorithm.

The EM algorithm applies to each pixel of the caution map. In step E, the algorithm estimates the ex-post distribution of the hidden variable, which is the pixel segmentation label. The ex post distribution shall be assessed on the basis of the attention weights and the characteristic map. In step M, the algorithm shall update the model parameters based on the estimated ex-post distribution. The algorithm repeats between step E and step M until convergence. The Expectation-Maximization Attention Network (EMANet) is a new deep learning architecture that utilizes a re-option-Maximization (EM) algorithm to learn and improve attention maps for semantic segmentation. The EM algorithm is a widely used optimization technique that values the parameters of a statistical model by alternating two steps: the step E calculates the expected value of caches variable data of observed data and current estimates of parameters, while the step M updates parameters estimates maximising the probability of observed data data data data data of expected cache values of variables.

In EMANet, the EM algorithm is applied to the caution maps, which are learned in a repeat manner by alternate assessment of the expected characteristics maps and the caution maps. Expected feature maps represent the expected activation of each channel of data map of current attention maps, while the attention maps represent the probability distribution of the significance of each channel feature map for segmentation work. Attention maps shall be updated by maximising the probability of the observed data maps of the expected features maps, while the expected features maps shall be updated by calculating the weighted sum of the charts with the attention maps.

EMANet is composed of three main elements: a trunk network, an attention improvement unit (attention reformation module) and a segmentation head. The torso network is usually a deep neural convection network (CNN) responsible for extracting high-level features from the input image. The care improvement unit consists of two subunits: a unit of expectation and a unit of maximization. The partition head is a decoder network that produces output forecast based on improved features.

The expectation unit shall assess the expected characteristics maps by calculating the weighted sum of the characteristics maps with the attention maps. Attention maps are learned by applying a softmax function to the output of a convection layer, which ensures that the attention values for each channel add to one. The expected feature maps are then calculated by multiplying each feature map with the corresponding attention map and sumping the results on all channels. The maximisation unit shall update the survey

maps by maximising the probability of the observed data maps of the expected characteristics maps. The maximisation module consists of two levels of condensation followed by a sigmoid function, which ensures that the attention values for each channel are limited between zero and one. The first layer of coherent analysis calculates the ratio of logarithmic probability of each channel of data maps of the expected characteristics maps, while the second layer of coherent analysis gathers the ratios of logarithmic probability on all channels to receive the attention maps.

The partition head is a decoding network that produces output prediction based on refined features. The segmentation head is usually composed of a series of layers of aggregate analysis and sampling upgrades that gradually increase spatial analysis of characteristics, followed by a final layer of concentric analysis that produces the segmentation map. The partition head can be adapted to different segmentation tasks by altering the number of output channels and the size of the final layer of aggregate imaging.

EMANet has several advantages over other models based on attention for semantic segmentation. Firstly, EMANet uses an EM repeat algorithm to learn and improve attention maps, which ensures that the attention maps capture both local and global input image framework information. Secondly, EMANet applies the survey maps directly to the characteristics maps, which avoids the loss of spatial analysis caused by the collection or reduction of sampling functions. Third, EMANet is able to handle input of arbitrary sizes using dilated spirals on the torso network.

5.12 Evaluation Metrics

5.12.1 Loss Functions

Cross Entropy

Cross-entropy is a widely used loss function that has been widely applied in a variety of deep learning applications, including image segmentation and semantic segmentation. Calculates the difference between the predicted probability distribution and the actual probability distribution. The mathematical function of cross-entropy is defined as follows:

$$H(p,q) = -\sum(p(x) * log(q(x))) \tag{5.12.1}$$

where p is the actual probability distribution, q is the predicted probability distribution while the variable x represents each possible event.

In image/semantic segmentation tasks, the cross-entropy loss function is used to optimize network parameters by comparing the predicted segmentation output with the ground truth segmentation. The output of the segmentation model is a probability map where each pixel represents the probability of belonging to a particular class. By comparing the predicted probability map with the ground truth mask, the cross-entropy loss function measures the difference between the two and provides a scaled value that the optimization algorithm can use to update the model parameters. The goal of the optimization is to minimize the cross-entropy loss function, which leads the model to produce more accurate segmentations.

Binary Cross Entropy

Binary Cross Entropy (BCE) loss is a loss function commonly used in image segmentation tasks, where the goal is to predict the probability of each pixel belonging to a certain class (e.g., object or background). The BCE loss is calculated by comparing the predicted probability of a pixel being a certain class with the actual class label of the pixel. The mathematical expression for BCE loss is:

$$BCE\,loss = -(y * log(p) + (1-y) * log(1-p)) \tag{5.12.2}$$

where y is the actual label (0 or 1) and p is the predicted probability that the label is 1. The BCE loss is a measure of how well the predicted probabilities match the actual labels. A lower BCE loss indicates better model performance.

In image segmentation tasks, the goal is to predict the class label for each pixel in an image. The predicted probability for each pixel is usually represented by a two-dimensional array of the same size as the image, where each element of the array corresponds to a pixel and is the predicted probability that that pixel belongs to a certain class. The BCE loss is then calculated for each pixel by comparing the predicted probability with the actual class label. Binary Cross Entropy loss is a loss function that measures the difference between the predicted probability of a label and the actual label. It is commonly used in image segmentation tasks where the goal is to predict the class of each pixel in an image.

5.12.2 Accuracy Metrics

Pixel Accuracy

Pixel accuracy (PA) is a common accuracy metric used in image segmentation tasks where the goal is to predict the class label of each pixel in an image. PA is a measure of how well the predicted class labels match the actual class labels for all pixels in the image. The mathematical expression for pixel accuracy is:

$$PA = (number\,of\,correctly\,classified\,pixels)/(total\,number\,of\,pixels) \tag{5.12.3}$$

This metric is simple to calculate, it simply counts the number of pixels correctly classified by the model and divides it by the total number of pixels in the image. A higher PA indicates better model performance. Pixel accuracy is often used as a key evaluation metric for image segmentation models, as it provides a quick and simple way to measure the overall performance of the model. However, it can be affected by class imbalance, where one class is more prevalent than the other. In such cases, the model can achieve high PA by simply predicting the majority class for all pixels, which is not optimal performance. Pixel accuracy (PA) is a simple and widely used accuracy metric for image segmentation tasks, where the goal is to predict the class label of each pixel in an image. It is calculated by counting the number of correctly classified pixels and dividing it by the total number of pixels. A higher PA indicates better model performance. However, it is affected by class imbalance and may not always be the best metric to evaluate model performance.

Intersection Over Union

Intersection over Union (IoU) is a common evaluation metric used in image segmentation tasks where the goal is to predict the class label of each pixel in an image. IoU is a measure of the similarity between predicted and actual class labels for a given class. It is calculated by taking the ratio of the intersection of the predicted and actual class labels to the union of the predicted and actual class labels. The mathematical expression for IoU is:

$$IoU = (True\,Positive)/(True\,Positive + False\,Positive + False\,Negative) \tag{5.12.4}$$

where $True\,Positive$ (TP) is the number of pixels correctly classified as the target class, $False\,Positive$ (FP) is the number of pixels incorrectly classified as the target class, and $False\,Negative$ (FN) is the number of pixels correctly classified as the non-desired target class.

IoU is commonly used in image segmentation applications as it provides a more detailed assessment of model performance than Pixel Accuracy. It considers not only the number of correctly classified pixels, but also the number of incorrectly classified or missed pixels. A higher IoU indicates better model performance. Intersection versus union (IoU) measures the similarity between predicted and actual class labels for a given class by taking the ratio of the intersection of predicted and actual class labels to the union of predicted and actual class labels. This is a more nuanced evaluation metric than pixel accuracy, as it considers not only the number of correctly classified pixels, but also the number of misclassified or missed pixels. Therefore, a higher IoU indicates better model performance.

Precision

Precision is a common evaluation metric used in image segmentation tasks where the goal is to predict the class label of each pixel in an image. It is a measure of how well the model is able to correctly identify pixels

belonging to the target class. It is calculated by dividing the number of true positives (TP) by the sum of true positives (TP) and false positives (FP). The mathematical expression for the accuracy is:

$$Precision = TP/(TP + FP) \tag{5.12.5}$$

where the variables TP and FP are defined above. Accuracy is a useful metric for evaluating a model's performance when the number of false positives is particularly significant, such as in medical imaging or object detection, where false positives can have serious consequences. A high accuracy indicates that the model has a low rate of false positives, while a low accuracy indicates a high rate of false positives. Accuracy measures the percentage of correctly classified pixels among the pixels classified in the target class. It is calculated by dividing the number of true positives by the sum of true positives and false positives. This is a useful metric when the number of false positives is particularly important, such as in medical imaging or object detection, where false positives can have serious consequences. A high accuracy indicates that the model has a low false positive rate, while a low accuracy indicates a high false positive rate.

Recall

Recall is a metric used to evaluate the performance of a classifier, especially in the context of binary classification and image segmentation. It is defined as the percentage of true positive cases (ie, the number of correctly identified positive cases) over the total number of true positive cases. Mathematically, it can be represented as:

$$Recall = TP/(TP + FN) \tag{5.12.6}$$

where the variables TP and FP are defined above.

In image segmentation, Recall is often used to measure the performance of an algorithm in detecting objects or regions of interest in an image. A high value of the Recall metric indicates that the algorithm has successfully identified a large percentage of true positives in the image. It is important to note that the Recall metric is sensitive to class imbalance, i.e. when the dataset has an unequal number of samples for each class. It is usually used in conjunction with accuracy, to get a full understanding of model performance.

F1-Score

F1-Score is a metric used to evaluate the performance of a classifier, especially in the context of binary classification and image segmentation. It is the harmonic mean of precision and recall and is defined as:

$$F1 - Score = 2 * (Precision * Recall)/(Precision + Recall) \tag{5.12.7}$$

In image segmentation, the F1-Score is often used as a measure of the overall performance of an algorithm in detecting objects or regions of interest in an image. It balances the trade-off between precision and recall, and a high F1-Score indicates that the algorithm has achieved a good balance between correctly identifying correct positive cases and minimizing the number of false positive cases. It is important to note that the F1-Score is sensitive to class imbalance, i.e. when the dataset has an unequal number of samples for each class. It is usually used in conjunction with precision and recall to fully understand model performance. In image segmentation applications, F1-Score is commonly used to evaluate model performance, especially in medical imaging, where the true-positive rate is of great importance.

Dice Coefficient

The Dice coefficient, also known as the Dice similarity coefficient (DSC), is a metric used to evaluate the performance of image segmentation algorithms. It is defined as the ratio of the intersection of the predicted partition and the ground truth partition to the union of the two partitions. Mathematically, it can be represented as:

$$DSC = 2 * (True\,Positives)/(2 * True\,Positives + False\,Positives + False\,Negatives) \tag{5.12.8}$$

In image segmentation, DSC is often used as a measure of an algorithm's performance in detecting objects or regions of interest in an image, especially in medical imaging. It ranges from 0 to 1, where a value of 1 indicates perfect segmentation and a value of 0 indicates completely incorrect segmentation. It is important to note that DSC is not sensitive to class imbalance, unlike accuracy or F1-Score. It is usually used in conjunction with other metrics, such as Precision and Recall metrics, to provide a complete understanding of model performance. In image segmentation, DSC is a commonly used evaluation metric, especially in medical imaging, where the object of interest may be limited and precision may not be the most appropriate metric to evaluate model performance.

Mean Square Error

Mean Squared Error (MSE) is a widely used metric for evaluating the performance of image segmentation algorithms, especially for evaluating the similarity of two images. Calculates the root mean square difference between the predicted and the actual image. Mathematically, it can be represented as:

$$MSE = (1/N) * sum((predicted - ground truth)^2) \tag{5.12.9}$$

where N is the number of pixels in the image.

In image segmentation, MSE is often used as a measure of an algorithm's performance in detecting objects or regions of interest in an image. It is commonly used when images are continuous, such as grayscale images, and ranges from 0 to infinity, where a value of 0 indicates perfect segmentation and a higher value indicates worse segmentation. It is important to note that MSE is sensitive to extreme pixels, i.e. pixels with high intensity values. In some cases, it may be useful to use other metrics, such as mean absolute error (MAE), which are less sensitive to outliers. It is used as an evaluation metric to compare an algorithm's prediction to the ground truth, especially in medical imaging, where the object of interest may be small and accuracy may not be the best metric to evaluate algorithm performance.

Chapter 6

Experiments & Results

This chapter presents the results of the experiments carried out with the models analyzed in chapter 5 on the data sets presented in chapter 4. The results are presented both in quantitative form, in the following tables, and in qualitative form with examples of mask images produced by the trained models, which are compared to the ground truth.

6.1 Computer System and Computational Nodes

The experiments of this work were performed entirely on ARIS, which is the Greek supercomputer developed and operated by GRNET in Athens. ARIS combines 5 different architectures divided into corresponding "node islands".

More specifically, the infrastructure consists of: An island with 426 computing nodes (thin nodes). Each node has two processors and each processor includes 10 processing cores thus offering a total of 8,520 cores (CPU cores). These nodes are suitable for highly parallel applications that can break their data into many small pieces before processing them. An island of thick nodes consisting of 44 nodes. Each node offers 4 processors, 40 cores and 512 GB of main memory per node. These nodes are suitable for applications that need very large central memory and not so much for high scaling. An island of GPU acceleration nodes (GPU nodes) consisting of 44 nodes. Each node contains 2 processors with 10 cores per processor, 64 GB of memory and 2 NVidia K40 GPUs. These nodes are suitable for applications that perform computational operations that can leverage graphics cards as coprocessors to accelerate calculations. An island of Xeon Phi accelerator nodes (phi nodes) consisting of 18 nodes, each containing 2 processors with 10 cores, 64 GB of memory and 2 Intel Xeon Phi 7120P coprocessors. It is suitable for parallel applications that leverage Intel Xeon Phi coprocessor technology. An island of machine learning nodes (ml node) consisting of 1 server, which contains 2 Intel E5-2698v4 processors, 512 GB of main memory and 8 NVIDIA V100 GPU cards. In summary, Aris consists of: - the data processing system used to process the data:

- 426 thin nodes: These are standard compute nodes that do not have accelerators.
- 44 GPU nodes: These nodes are equipped with "2 × NVIDIA Tesla k40m" accelerators.
- 18 phi nodes: These nodes are equipped with "2 × INTEL Xeon Phi 7120p" accelerators.
- 44 fat nodes: These nodes offer a higher number of cores and more memory per core than thin nodes.
- 1 ML node: This node is powered by "8 × NVIDIA Volta V100" accelerators.

In the present work, ML nodes were used to conduct experiments, which were executed using SLURM tasks. Computational resources were subject to some limitations, such as a maximum run time of 96 hours and a maximum memory usage of 100 GB. These limitations had an impact on the choice of parameters used in the previous paragraph.

6.2 Presentation of Experiments & Results

In the present study, we evaluated the performance of various binary segmentation models using four different metrics: IoU, Recall, Precision, and Dice Loss. We performed experiments on two different image resolution sets for the Corsican Fire Database: 128×128 and 256×256. For each analysis, we trained and tested the models and reported the results in separate tables.

Table 6.1: Trained and evaluated on the Coriscan Fire database, Image size: 128×128, Epochs: 200

	Mean IoU %	Recall %	Precision %	Dice Loss %
DeepLabv3+	77.11616	91.87359	86.43572	8.92813
HRNet	71.43123	92.82825	82.51125	11.36854
HRNet + OCR	78.85412	87.44712	**92.14019**	11.03240
PIDNet	61.10719	85.45229	74.52885	19.75203
DDRNet	78.50521	87.99254	88.97642	8.92639
Swin Transformer	**79.35761**	**94.75985**	87.90729	**7.68386**
UNet3+	77.68424	88.13359	88.72304	11.28386
BASNet	77.11616	91.87359	86.43572	8.92813
BiSeNet	70.71571	76.85744	78.68710	21.07191
EmaNet	78.61003	88.13534	89.30744	10.48374
AuNet	78.59888	88.07890	90.92335	9.55796

Table 6.2: Trained and evaluated on the Coriscan Fire database, Image size: 256×256, Epochs: 200

	Mean IoU %	Recall %	Precision %	Dice Loss%
DeepLabv3+	82.17820	92.07052	87.34108	10.03188
HRNet	78.54949	87.72961	86.78484	13.26110
HRNet + OCR	82.35714	94.12200	86.52257	10.35253
PIDNet	70.58202	91.14873	77.46957	17.46671
DDRNet	82.04155	94.08146	86.30517	8.85696
Swin Transformer	83.73605	**94.22274**	86.73371	**8.44437**
UNet3+	80.52014	90.37410	85.86813	11.88129
BASNet	83.43717	90.50857	86.58721	9.07097
BiSeNet	72.90698	89.77735	79.47915	18.21647
EmaNet	83.32208	90.71482	87.20958	10.23520
AuNet	**84.07098**	89.81584	**90.46853**	9.42433

Tables 6.1 and 6.2 show the results of the models used in this thesis. Overall, the analysis shows performance differences among the 11 deep learning models for fire image segmentation. The Swin Transformer model stands out with the highest Mean IoU and Recall and lowest Dice Loss, indicating its effectiveness in accurately capturing fire events in the 128×128 image set. Additionally, Swin Transformer exhibits the highest Recall and lowest Dice Loss in the 256×256 image set. Achieving this performance can be attributed to several technical factors. First, the Swin Transformer's Self-Attention Mechanism allows it to efficiently capture long-range dependencies within images, enabling a better understanding of the complex patterns and shapes associated with fire events. At the same time, the hierarchical structure of the Swin Transformer allows the model to extract multi-scale features, capturing both local and global information, which is crucial for accurate segmentation. In addition, Swin Transformer uses trainable Positional Encodings, facilitating the model's ability to efficiently encode spatial information, which is important for accurate fire localization in images.

The AuNet model achieved significant results in fire case segmentation, outperforming several other models on the 256×256 image set, as shown by the provided evaluation metrics, achieving the best performance in Mean IoU (84.07098%) and Precision (90.46853%). This performance can be attributed to its unique architecture and design choices. The AuNet architecture combines the U-Net architecture with attention mechanisms, which play a key role in capturing and highlighting relevant features for accurate segmentation. The U-Net

architecture, with its encoder-decoder structure, is known for its ability to capture both low- and high-level features. By incorporating attention mechanisms, AuNet enhances the model's ability to focus on important regions and features, particularly important for fire segmentation tasks. Attention mechanisms in AuNet allow the model to selectively attend to informative regions while suppressing irrelevant or noisy regions. This is particularly beneficial in fire segmentation, where the distinction between the fire and surrounding areas is crucial. By directing attention to the discriminative visual features of fire, such as color, intensity, and texture, AuNet can efficiently learn discriminative representations for accurate segmentation. In addition, AuNet's impressive performance can also be attributed to its ability to capture context information and spatial dependencies through skip connections. Skip connections allow the model to integrate information from multiple analyses, facilitating the fusion of both local and global context. This is particularly beneficial in fire segmentation work, where the size and shape of fire cases can vary significantly. The high Mean IoU, Recall, Precision and low Dice Loss values achieved by AuNet further demonstrate its effectiveness in accurate fire case segmentation. The model's ability to exploit attentional mechanisms, capture context information, and combine low- and high-level features contributes to its excellent performance.

The HRNet + OCR model also performs well with a high Precision value, indicating a low false positive rate. The PIDNet and BiSeNet models exhibit relatively lower Mean IoU values and higher Dice Loss, indicating room for improvement in their performance. The BiSeNet model exhibited relatively lower performance with an average IoU of 70.72% and a recall of 76.86%. This is due to certain limitations set by its architecture itself. BiSeNet uses a relatively more superficial architecture, which may limit its ability to efficiently capture context information. Consequently, the model struggles to accurately detect and segment fire events in complex scenes. BiSeNet's smaller receptive field may hinder its ability to capture long-range dependencies and understand the overall context of fire events, leading to incomplete and inaccurate segmentations.

Models that demonstrated high performance metrics exhibited notable architectural features. They excelled at capturing content-related information, enabling a comprehensive understanding of complex patterns and patterns associated with fire events. This achievement was facilitated by the integration of self-attention mechanisms and hierarchical feature extraction. In particular, models such as Swin Transformer leveraged self-aware mechanisms to capture long-range dependencies, while HRNet and UNet3+ used hierarchical architectures for efficient multi-scale feature extraction. Accurate location and accurate location coding were essential factors in successfully segmenting fire incidents. The best-performing models used techniques such as learnable positional encoding, which enhanced their ability to efficiently encode spatial information. Models such as Swin Transformer and DDRNet used learnable location encodings, helping to accurately locate fire events within images.

On the other hand, the lower performing models showed limitations in some areas. Models such as BiSeNet and DDRNet have struggled to effectively capture context information, possibly due to their shallower architecture or the absence of mechanisms to capture long-range dependencies. As a result, these models demonstrated an incomplete understanding of the overall context of fire events. Another limitation observed in the lower performing models was the presence of smaller receptive fields, as observed in BiSeNet. This limited their ability to capture long-range dependencies and integrate global information, resulting in incomplete segmentations and inaccurate identification of fire events.

6.3 Presentation of Results Using Transfer Learning

In the present study, we conducted research involving the application of transfer learning [Zhu+20] techniques to tackle the task of lung segmentation in medical images. Our experiments focused on leveraging the knowledge gained from training deep learning models on the Corsican Fire database and then applying transfer learning to evaluate their performance on the COVID-QU-Ex, Kvasir-Instrument, and Cell datasets. The objective was to evaluate the effectiveness of using the pre-trained weights from the source task to improve lung segmentation accuracy on the target medical dataset. First, we trained the deep learning models using the Corsican Fire Database, which provided a wide range of images, with their corresponding masks. The models were trained to learn complex feature representations related to fire segmentation. Through this process, we obtained well-trained models with optimized weights and feature extraction capabilities. We then used the transfer learning technique to adapt these pre-trained models to the lung segmentation task using the COVID-QU-Ex, Kvasir-Instrument and Cell datasets. These datasets consist of large collections

of medical images, each annotated with precise lung segmentation masks. By initializing the models with the pre-trained weights obtained from the Corsican Fire database, we aimed to leverage previously learned representations to improve the accuracy and efficiency of lung segmentation. The results of our experiments revealed significant improvements in lung segmentation performance when transfer learning was used. Compared to models trained solely on the COVID-QU-Ex, Kvasir-Instrument, and Cell datasets without the aid of transfer learning, models with transfer learning capability demonstrated improved precision, recall, and overall accuracy. The use of pre-trained weights from the original work facilitated the models' ability to capture relevant anatomical structures and distinctive features, leading to more accurate and consistent lung segmentation results. In addition, the application of transfer learning reduced the training time required to converge the models. By initializing the models with pre-trained weights, we provided a robust starting point that allowed the models to quickly adapt to the lung segmentation task. This accelerated convergence not only saved computational resources, but also enabled the deployment of the models in real-time scenarios where fast and accurate segmentation is crucial.

6.3.1 Presenting Results on the COVID-QU-Ex Dataset Using Transfer Learning

The results recorded on the COVID-QU-Ex dataset using transfer learning are presented in tables 6.3 and 6.4 for images of size 128×128 and 256×256 respectively.

Table 6.3: Trained on the Coriscan Fire database and evaluated on the COVID-QU-Ex dataset using transfer learning, Image size: 128×128

	Mean IoU %	Recall %	Precision %	Dice Loss %
DeepLabv3+	82.69175	87.45567	90.07400	9.09253
HRNet	71.70770	66.58837	93.22822	18.82998
HRNet + OCR	75.89908	75.14210	81.37119	17.33686
PIDNet	**92.75437**	**94.02607**	95.06878	**4.12282**
DDRNet	76.56956	82.78332	85.71495	12.69335
Swin Transformer	82.59734	86.29640	85.67260	9.05948
UNet3+	90.18486	91.58323	**95.96047**	5.07815
BASNet	82.34931	86.07644	85.09008	9.16974
BiSeNet	78.20549	78.87384	80.25656	15.14450
EmaNet	78.11949	81.22073	81.00905	12.60521
AuNet	81.11299	84.93019	90.18278	9.96949

Table 6.4: Trained on the Coriscan Fire database and evaluated on the COVID-QU-Ex dataset using transfer learning, Image size: 256 × 256

	Mean IoU %	Recall %	Precision %	Dice Loss %
DeepLabv3+	84.05765	87.26623	86.28042	11.01869
HRNet	87.18854	84.81492	91.98151	8.96609
HRNet + OCR	80.72064	78.68837	87.85881	17.00936
PIDNet	93.96398	94.97285	94.81903	4.75383
DDRNet	85.20530	86.35141	88.67397	10.27042
Swin Transformer	76.33766	73.26006	84.24978	17.71929
UNet3+	**94.95149**	**94.46418**	**96.32667**	**3.97119**
BASNet	75.84068	69.11792	73.42729	17.92031
BiSeNet	78.26812	76.32833	85.33376	19.49315
EmaNet	80.83709	77.45330	77.89674	14.17940
AuNet	74.55735	63.42070	71.68256	19.33504

In both cases, the models were evaluated based on the Mean IoU (Intersection over Union), Recall, Precision and Dice Loss metrics. It appears that the PIDNet and UNet3+ models achieved the highest scores in Mean

IoU and Dice Loss, indicating better segmentation accuracy and similarity between the predicted masks and the ground truth masks. PIDNet also exhibited high Recall and Precision values, demonstrating its ability to capture a large proportion of true positive pixels while minimizing false positives. HRNet, HRNet + OCR, DDRNet, Swin Transformer, BASNet, BiSeNet, EmaNet, and AuNet also achieved competitive results, but slightly underperformed PIDNet and UNet3+ in these metrics.

The different performance of PIDNet in the lung and fire segmentation tasks can be attributed to several factors. First, the datasets themselves have different characteristics. The lung segmentation dataset can have clear and distinct boundaries between lung regions, making it easier for the model to learn and accurately segment them. However, the fire segmentation dataset likely contains more complex and varied fire patterns, posing challenges for the model to generalize and accurately capture fire areas. Additionally, the choice of model architecture can affect performance. While PIDNet may have been effective for lung segmentation, it may not be suitable for fire segmentation due to the differences in visual characteristics and complexity between the two tasks. Task-specific characteristics, such as the presence of specific architectural elements, the number of layers, or the use of skip connections, could affect the model's ability to capture relevant features and generalize to unseen data. Furthermore, fire segmentation is generally a more difficult task compared to lung segmentation due to the complex and dynamic nature of fires. PIDNet may not be specifically designed or equipped to handle such complexity, resulting in reduced fire segmentation performance. The effectiveness of transfer learning also depends on the similarity between the source (lung segmentation) and target (fire segmentation) domains. If the two tasks have significant differences in optical characteristics or imaging conditions, the transferable knowledge from lung segmentation may not be directly applicable to fire segmentation. At the same time, the optimization procedure and hyperparameter settings for PIDNet may be tuned specifically for lung segmentation, and these settings may not be optimal for fire segmentation. Fine-tuning or tuning the hyperparameters of PIDNet specifically for fire segmentation could potentially improve its performance in this area.

6.3.2 Presentation of Results on the Kvasir-Instrument Dataset Using Transfer Learning

Results recorded on the Kvasir-Instrument dataset using transfer learning are presented in tables 6.5 and 6.6 for images of size 128×128 and 256×256 respectively.

Table 6.5: Trained on the Coriscan Fire database and evaluated on the Kvasir-Instrument dataset using transfer learning, Image size: 128×128

	Mean IoU %	Recall %	Precision %	Dice Loss %
DeepLabv3+	77.14478	75.07174	61.14997	35.41356
HRNet	46.22021	16.88175	18.96797	90.09740
HRNet + OCR	63.88215	59.00847	49.83600	50.29171
DDRNet	69.46435	73.68438	52.30587	37.36653
Swin Transformer	**83.57536**	84.37541	**68.64790**	28.31224
UNet3+	77.76119	72.04443	49.41012	52.47760
BASNet	72.69732	71.05387	57.27568	38.44070
BiSeNet	70.71586	56.15295	67.24958	40.50739
EmaNet	81.56526	**85.22605**	67.75965	35.01416
AuNet	81.28684	80.89353	70.65854	**25.05858**

For the 128×128 6.5 image size, DeepLabv3+, leveraging atrous convolutions and pyramid pooling units, outperforms Mean IoU, highlighting its ability to distinguish object classes. HRNet, characterized for incorporating high-resolution features, faces challenges in capturing the overall context, resulting in lower Mean IoU and Recall. HRNet + OCR, with an integrated OCR module, improves context understanding, enhancing Mean IoU and Recall. DDRNet's dense dilated convolutions effectively capture multi-scale frame cues, resulting in balanced performance. Swin Transformer uses self-aware mechanisms, capturing long-range dependencies and yielding the highest Mean IoU. UNet3+, with symmetric shrink and expand paths, achieves competitive Mean IoU and Recall, but shows a tendency for false positives, affecting Precision. BASNet,

Table 6.6: Trained on the Coriscan Fire database and evaluated on the Kvasir-Instrument dataset using transfer learning, Image size: 256×256

	Mean IoU %	Recall %	Precision %	Dice Loss %
DeepLabv3+	73.85830	61.16712	65.44764	34.94264
HRNet	78.72248	70.34056	68.36770	31.95459
HRNet + OCR	76.40077	76.79989	65.90791	28.05802
PIDNet	71.47782	78.32768	65.92414	38.39293
DDRNet	77.40945	72.61441	73.72814	24.10564
Swin Transformer	76.88540	75.12174	65.51796	29.75468
UNet3+	82.72024	72.45127	64.89172	35.89299
BASNet	77.88340	75.43476	65.12625	29.37803
BiSeNet	78.24850	73.00691	74.26201	27.09357
EmaNet	83.24617	79.48328	64.31091	34.13178
AuNet	**84.37481**	**87.21559**	**77.46184**	**18.91523**

leveraging boundary-aware modules, balances Mean IoU, Precision and Recall. BiSeNet's spatial and frame paths offer multi-scale feature extraction, but struggle with fine details and true positives. EmaNet, with context embedding attention, maintains a balance between global and local information, resulting in competitive performance.

DeepLabv3+ maintains its ability for multiscale feature integration and content understanding, showing stable performance at both 6.5 6.6 image sizes. HRNet, which is based on high-resolution feature learning, effectively preserves spatial information, resulting in superior performance compared to its performance with the smallest image size. The integration of the OCR module into HRNet + OCR further enhances the understanding of the general context, translating into notable Recall and Precision, although the impact on Mean IoU remains limited. PIDNet's pyramid dilated convolutions continue to accurately capture multi-level features, resulting in stable and balanced performance in Mean IoU, Recall and Precision. Similarly, the stable performance of DDRNet indicates its reliability in capturing complex image and frame details, reflecting its dense architecture with dilated residues. However, the outstanding performer remains the Swin Transformer, a model based on self-attentive mechanisms. It exhibits remarkable adaptability to larger image size, achieving the lowest Dice Loss, which indicates its ability to capture the entire frame and achieve accurate segmentations. This performance confirms the power of self-attention mechanisms in handling complex image structures. While UNet3+ and BASNet maintain their relative positions on the performance spectrum, it is worth noting that some architectural peculiarities may contribute to their different performance dynamics. UNet3+ continues to exhibit multi-scale feature capture, but faces challenges in reducing false positives, which highlights the impact of architectural choices on accuracy-related metrics. EmaNet's context integration attention mechanism continues to demonstrate adaptability to both local and global contexts, delivering competitive performance across multiple metrics. Ultimately, AuNet emerges as the top model, consistently outperforming all metrics. Its attention mechanisms, acting as complex selection filters, exhibit remarkable precision and accuracy in object segmentation, leading to extremely low dice loss and high accuracy. We therefore observe that for an image size of 128×128, Swin Transformer emerges as the dominant one, exhibiting excellent proficiency in capturing the overall frame and minimizing disparity, making it a strong choice for tasks that require high Mean IoU and Dice loss reduction. In contrast, AuNet's precision-driven performance on larger 256x256 images makes it an optimal choice when aiming for higher Recall, Precision, and most importantly, lower Dice Loss.

6.3.3 Presentation of Results on the Cell Data Set Using Transfer Learning

The results recorded on the Cell dataset using transfer learning are presented in tables 6.7 and 6.8 for images of size 128×128 and 256×256 respectively.

Evaluation of the models at image sizes 128×128 6.7 and 256×256 6.8 reveals the complex relationship between architecture and performance metrics. For the image size 128×128, several models show compelling results. The Swin Transformer stands out by achieving the highest Mean IoU (85.44680%) and Precision (95.01020%), indicating its remarkable ability to accurately segment cells while minimizing false positives. Its

Table 6.7: Trained on the Coriscan Fire dataset and evaluated on the Cell dataset using transfer learning, Image size: 128×128

	Mean IoU %	Recall %	Precision %	Dice Loss %
DeepLabv3+	81.05836	90.82868	93.65665	5.00242
HRNet	82.09502	86.18971	93.67148	5.49589
HRNet + OCR	74.20295	86.94881	88.94111	11.26265
PIDNet	66.26700	71.92245	75.24118	19.90965
DDRNet	83.28248	92.77814	93.51345	4.02310
Swin Transformer	**85.44680**	93.85210	**95.01020**	**3.01809**
UNet3+	83.01646	**93.86075**	93.53656	3.77546
BASNet	80.11727	89.59109	88.03489	7.31221
BiSeNet	80.95970	86.30021	87.93396	6.65844
EmaNet	78.37056	90.63074	89.59932	7.44206
AuNet	83.22239	93.70270	93.93919	5.35603

Table 6.8: Trained on the Coriscan Fire dataset and evaluated on the Cell dataset using transfer learning, Image size: 256×256

	Mean IoU %	Recall %	Precision %	Dice Loss %
DeepLabv3+	79.88971	85.75538	87.49906	11.70098
HRNet	74.39137	85.14735	77.51167	17.35575
HRNet + OCR	88.32061	90.19877	92.22846	5.75654
PIDNet	82.10590	89.62398	91.68552	8.58695
DDRNet	80.96483	82.17690	85.89050	13.53812
Swin Transformer	86.89709	91.48734	90.60963	5.44250
UNet3+	84.34225	94.21741	88.58120	8.52977
BASNet	80.78324	82.76205	84.00797	11.12321
BiSeNet	84.41522	91.09681	89.64404	9.49268
EmaNet	85.57298	88.87509	87.37724	8.03816
AuNet	**89.00210**	**94.92373**	**94.68843**	**4.99055**

architectural emphasis on self-care allows for global recording of content, necessary for distinguishing complex cell boundaries. In addition, UNet3+ presents a significant Recall (93.86075%), highlighting its ability in true positive identification. This success can be attributed to UNet3+'s encoder-decoder architecture, which facilitates the extraction of multi-scale features crucial for cell segmentation. Additionally, AuNet excels in recall (93.70270%) and Dice Loss reduction (5.35603%), highlighting its power in detecting true positives and minimizing the disparity between the predicted masks and the baseline of truth. When moving to the larger image size of 256x256, AuNet emerges as the top choice, outperforming the other models in all metrics, including the highest Mean IoU (89.00210%), Recall (94.92373%), Precision (94 .68843%) and Dice Loss (4.99055%). Its architectural design combines care mechanisms and U-Net structures, facilitating integrated feature extraction and robust performance in all aspects of cell segmentation. Additionally, HRNet + OCR demonstrates its capability with significant improvements in Average IoU (88.32061%) and Accuracy (92.22846%), highlighting the effective use of multi-scale parallel fusion to preserve both high detail analysis as well as the semantic context, which is crucial for the precise delineation of cell boundaries. These results highlight the importance of architectural nuances in achieving optimal binary cell division. While models such as Swin Transformer exploit general context capture through self-awareness, UNet3+ exploits encoder-decoder architecture to extract multi-scale features, and AuNet combines warehousing mechanisms with U-Net structures for integrated performances. As the analysis demonstrates, a fine-grained understanding of these architectural components enables informed model selection based on specific segmentation requirements and image sizes.

6.4 Presentation of Qualitative Results

The subsection presents the qualitative results of the models on the four datasets for two , in order to gain a more complete picture of the performance of the models.

6.4.1 Qualitative Results of DeepLabv3+

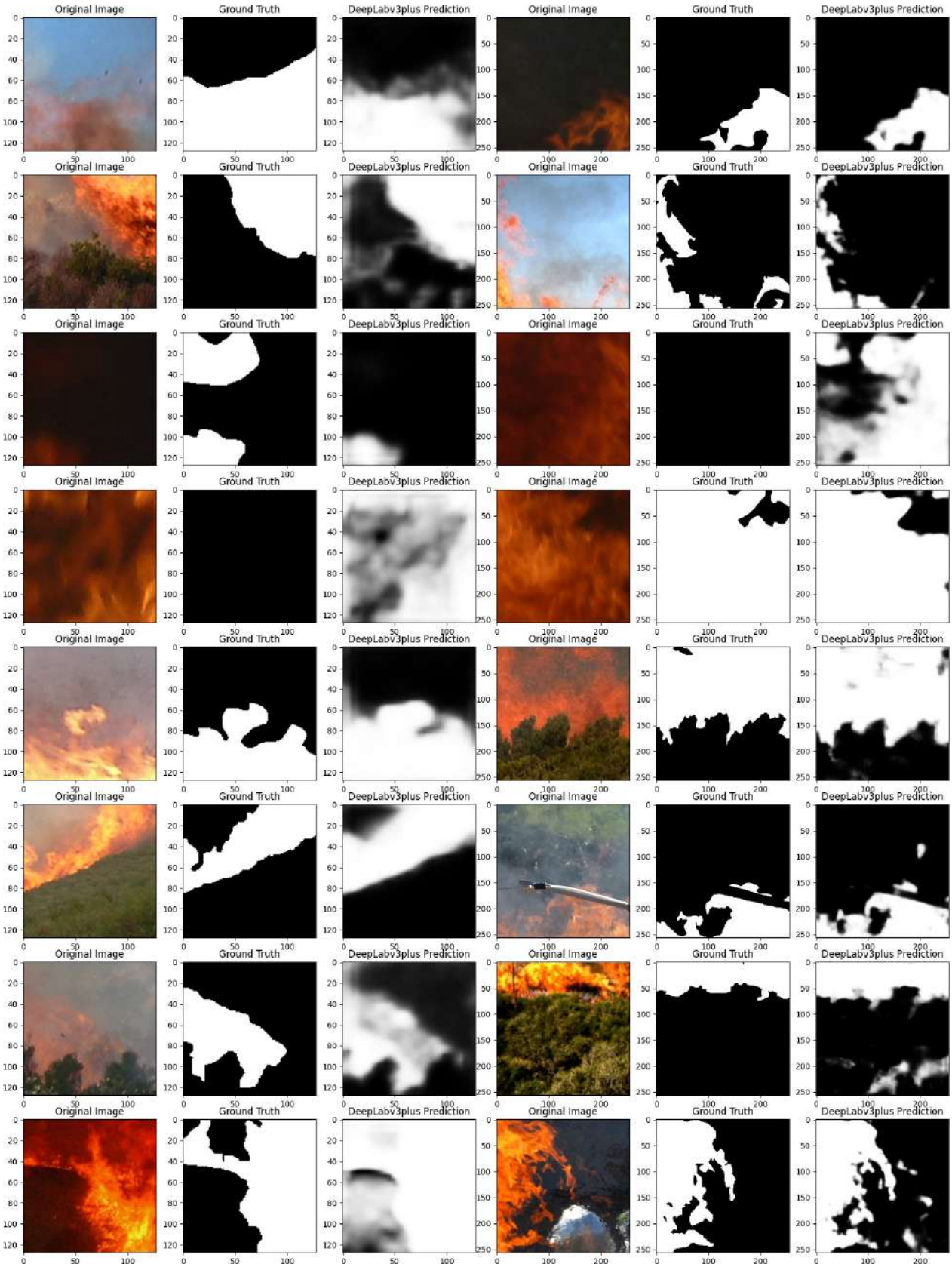

Figure 6.4.1: The original, mask and prediction for 128×128 and 256×256 image sizes respectively on the Corsican Fire dataset

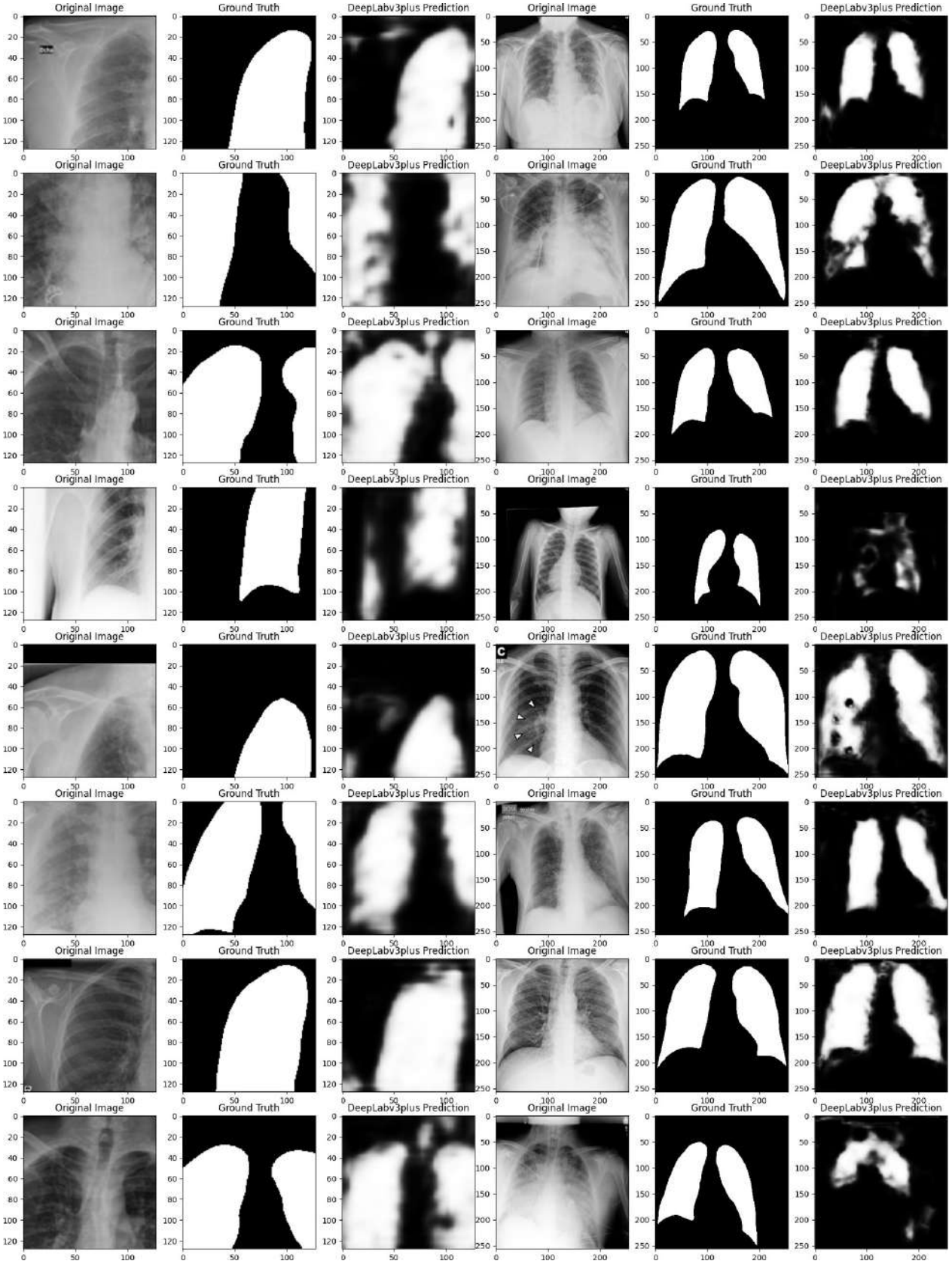

Figure 6.4.2: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively on the COVID-QU-Ex dataset

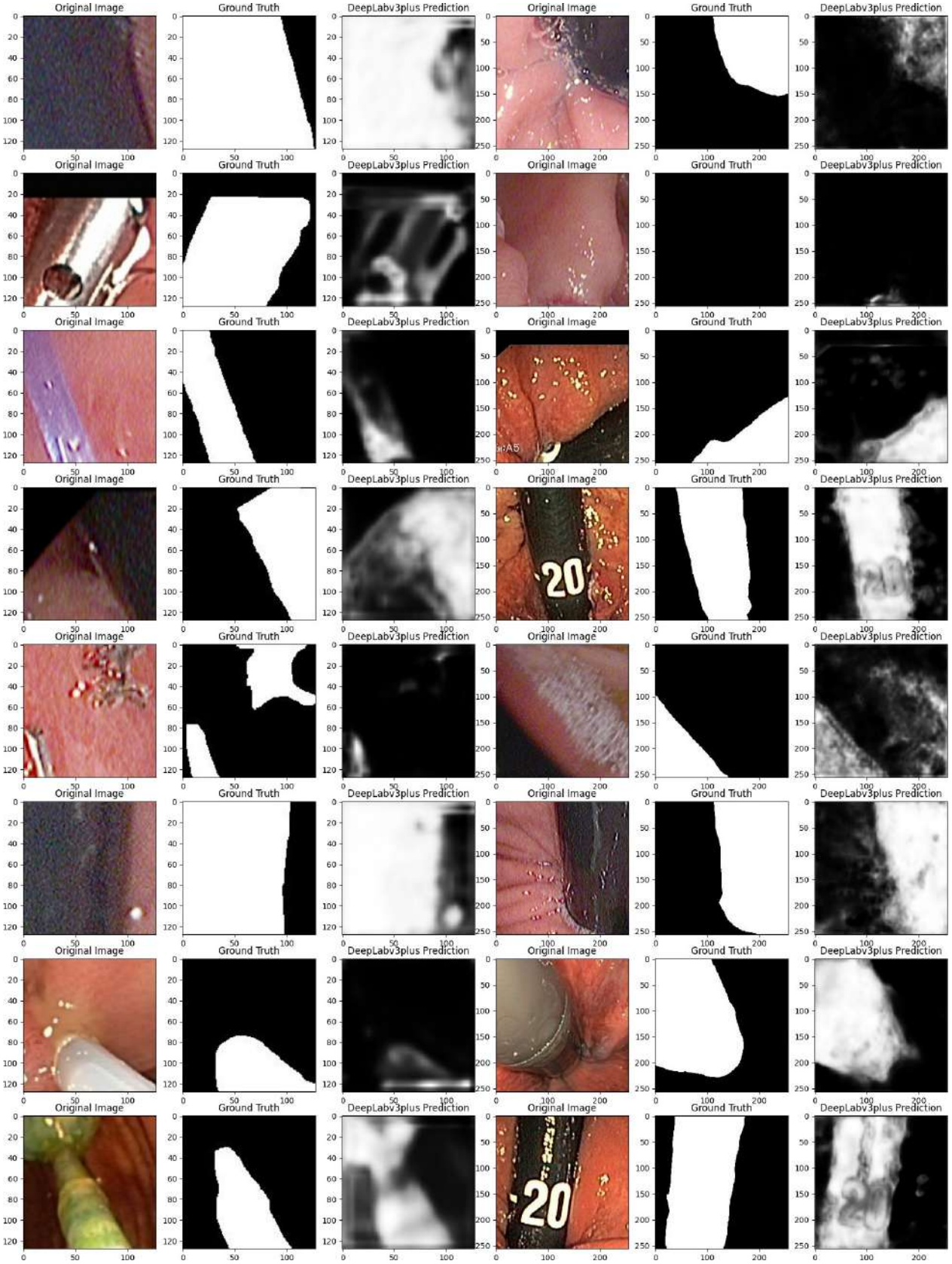

Figure 6.4.3: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Kvasir-Instrument dataset

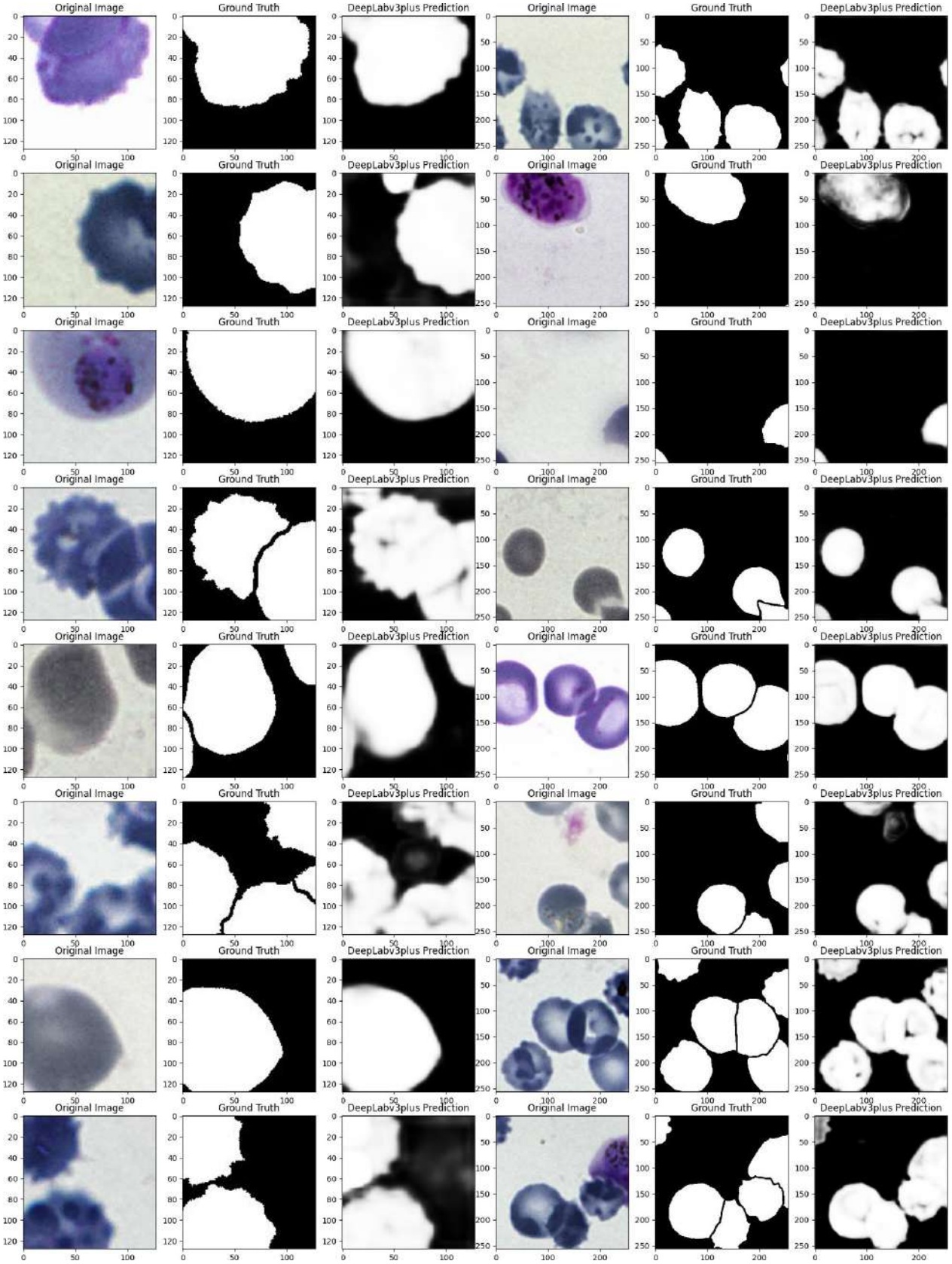

Figure 6.4.4: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Cell dataset

6.4.2 Qualitative Results of HRNet

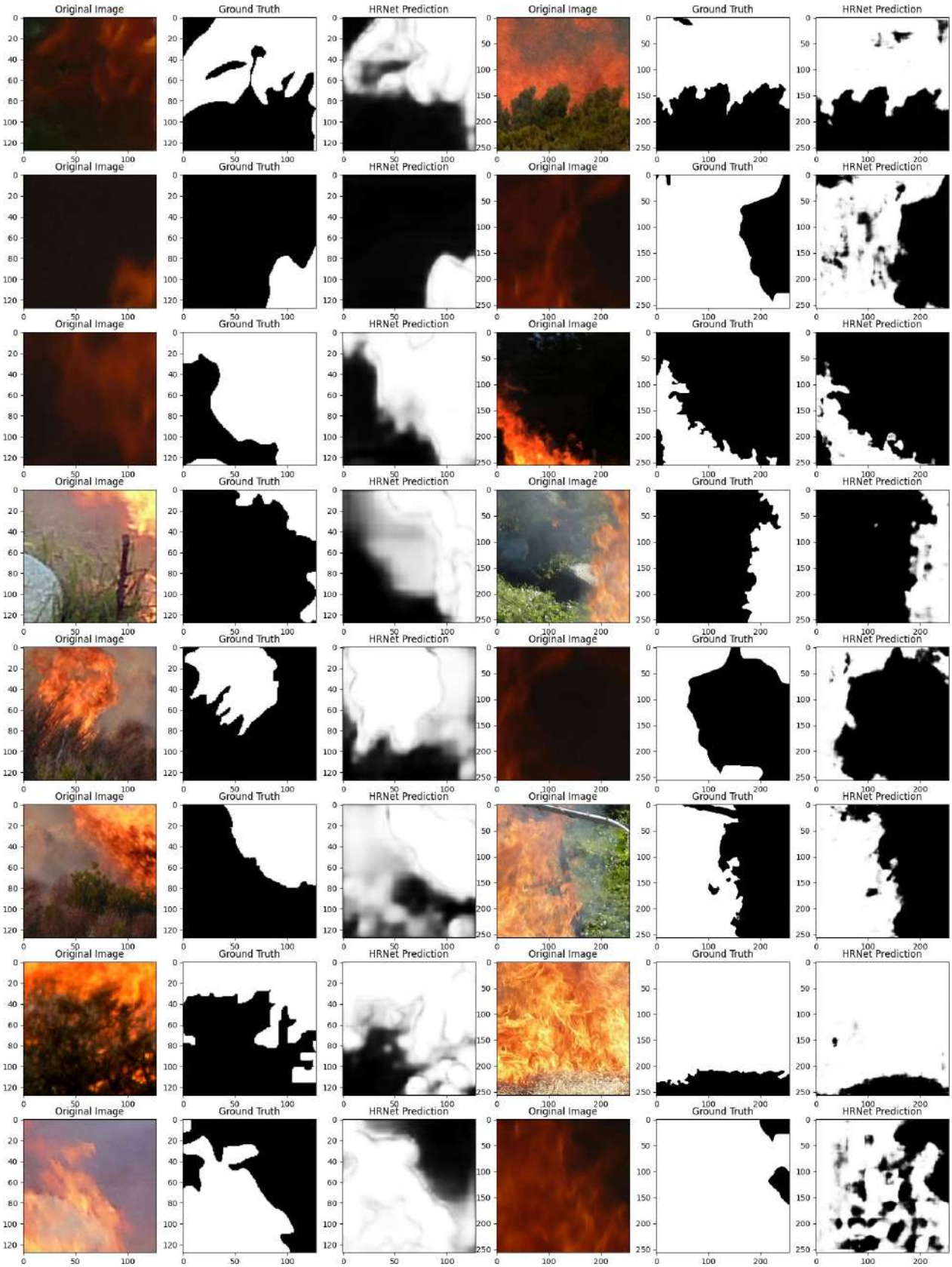

Figure 6.4.5: The original, mask and prediction for 128×128 and 256×256 image sizes respectively on the Corsican Fire dataset

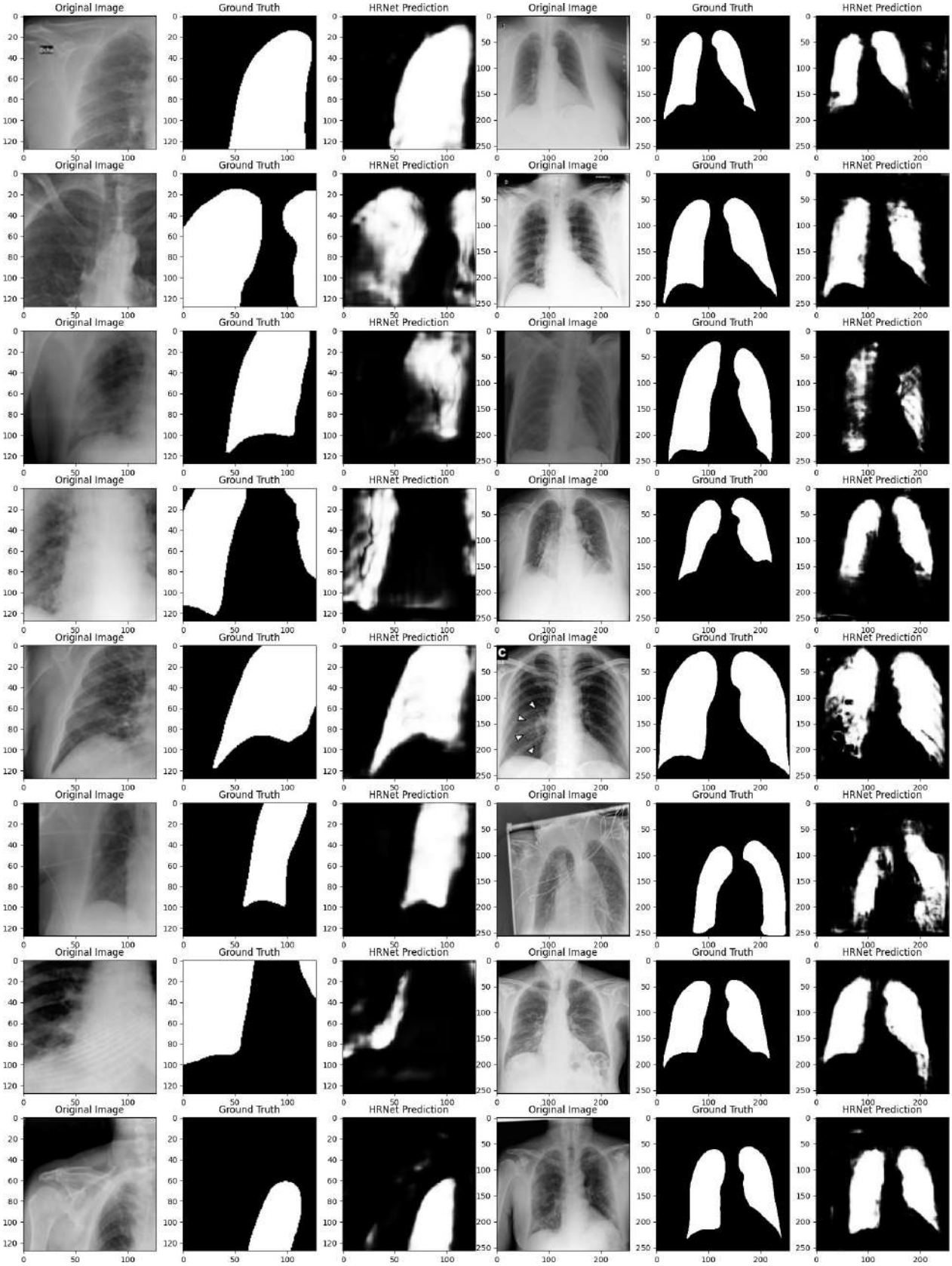

Figure 6.4.6: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively on the COVID-QU-Ex dataset

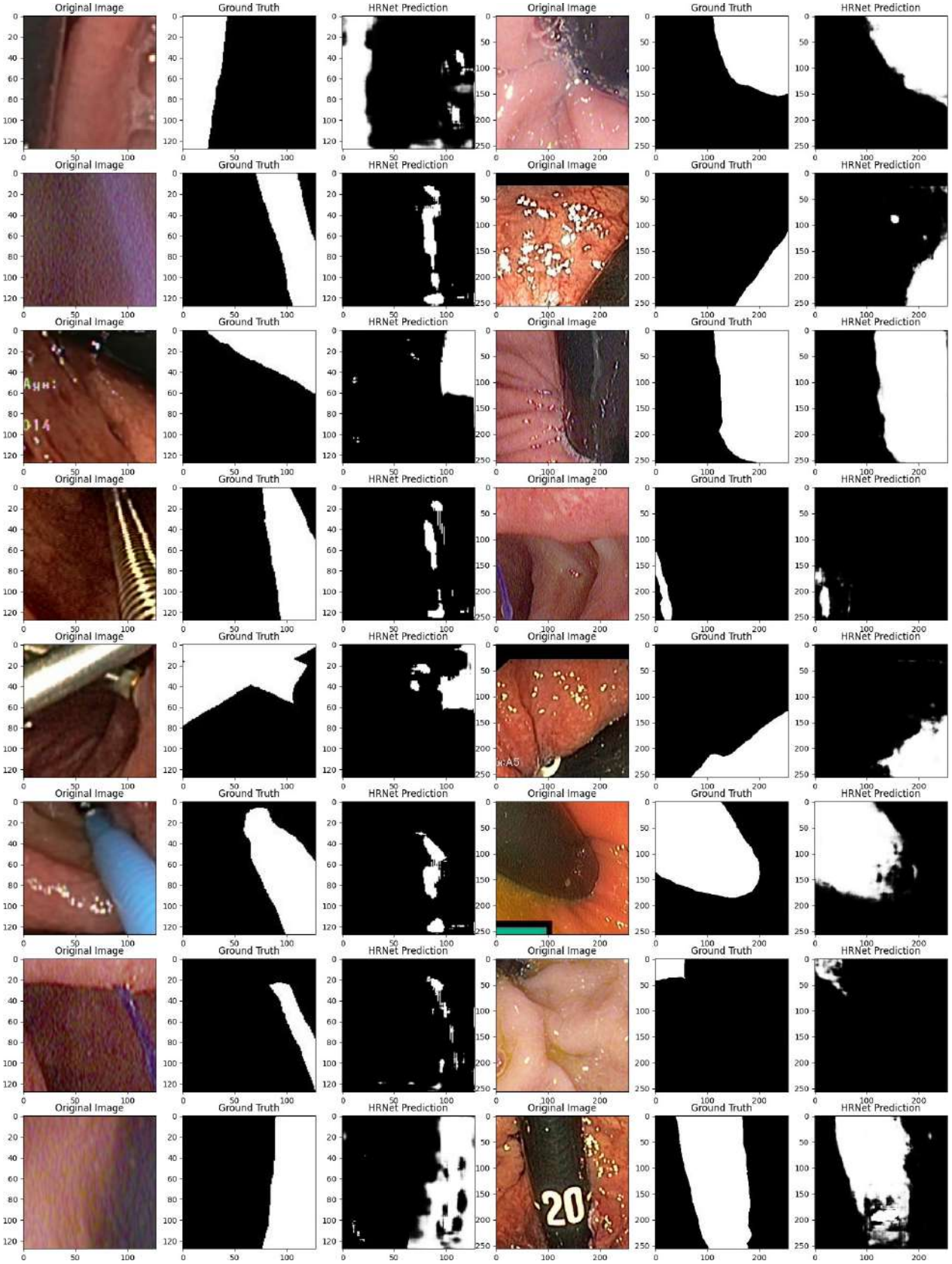

Figure 6.4.7: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Kvasir-Instrument dataset

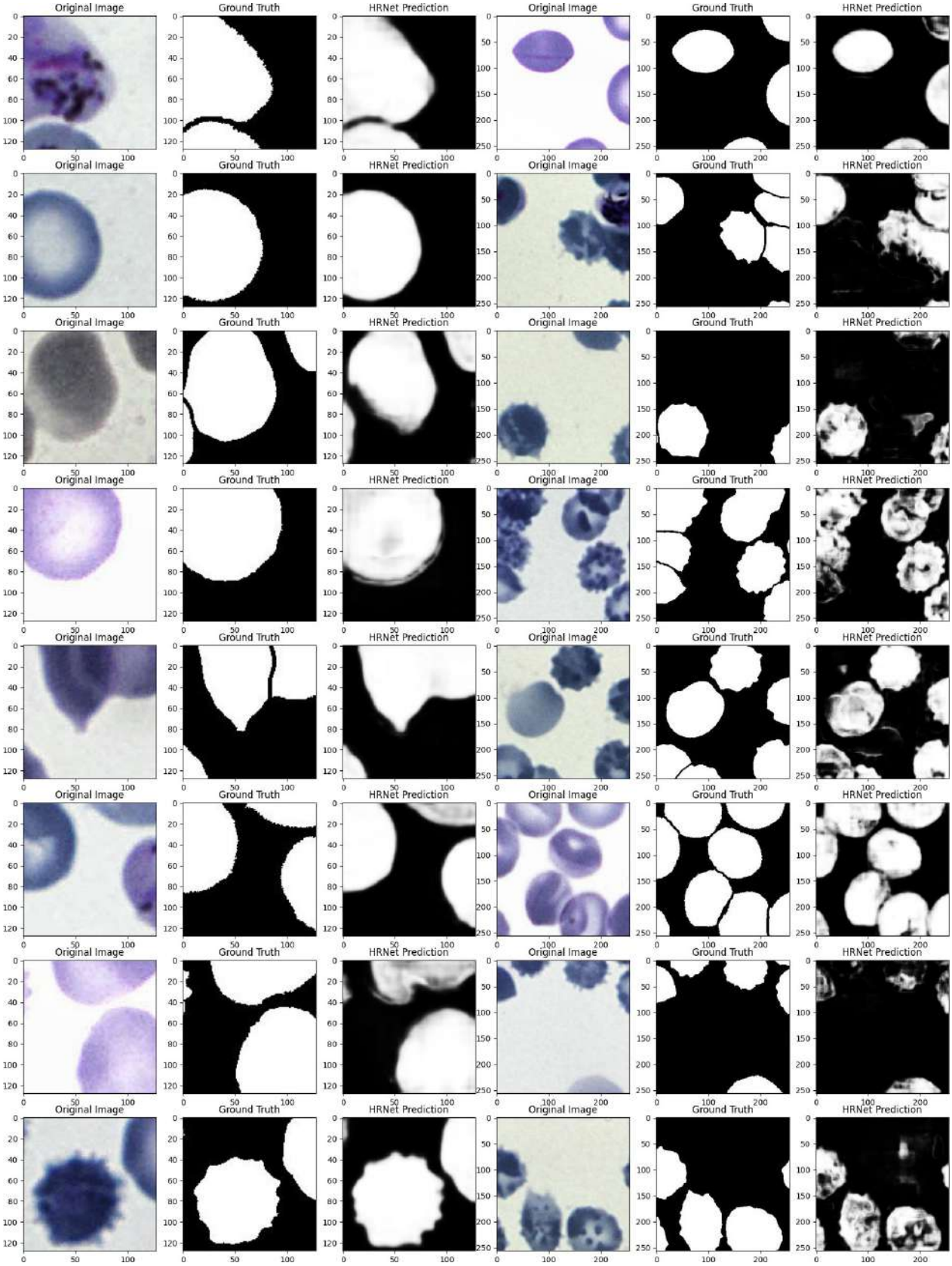

Figure 6.4.8: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Cell dataset

6.4.3 Quality Results of HRNet + OCR

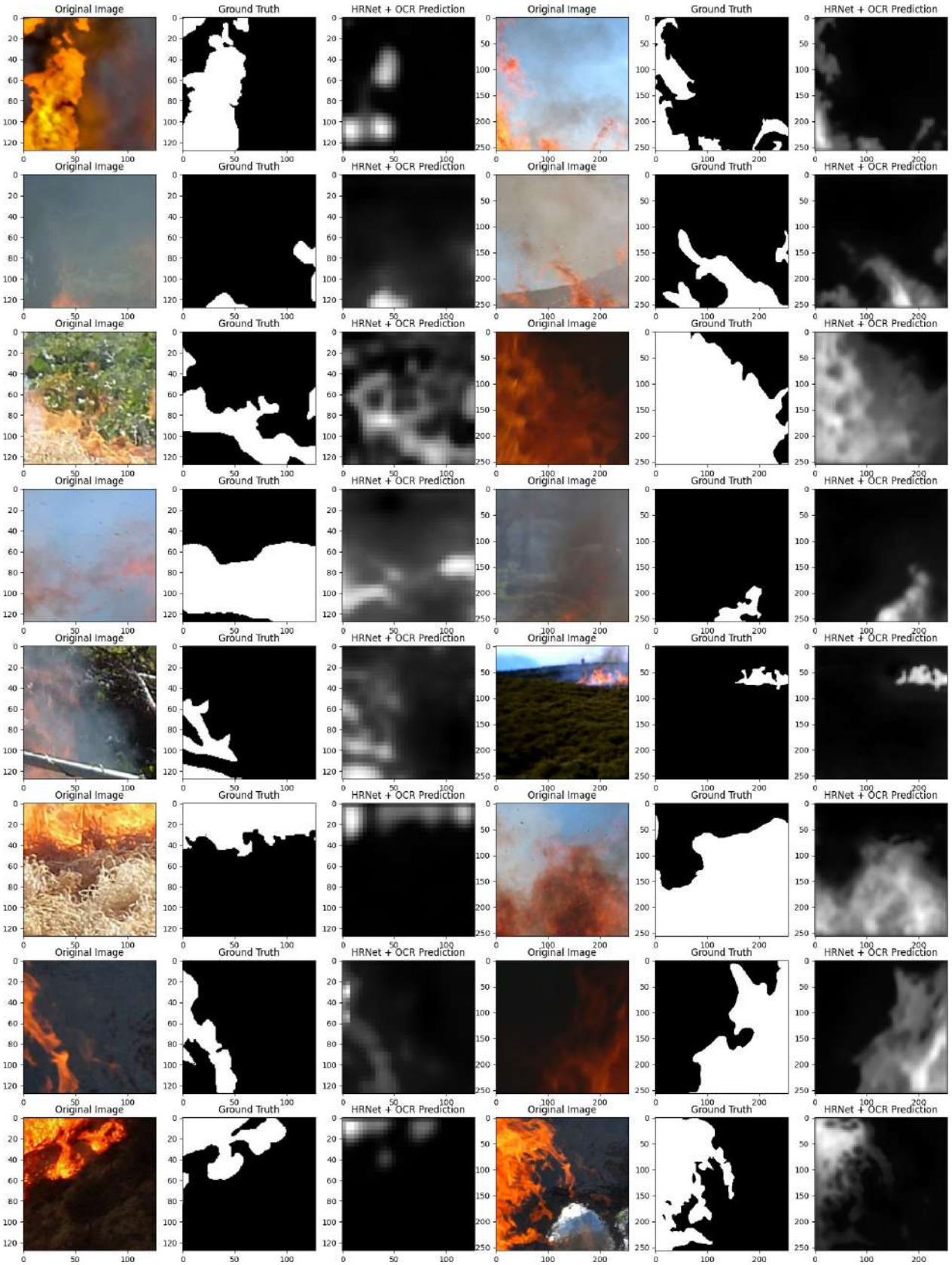

Figure 6.4.9: The original, mask and prediction for 128×128 and 256×256 image sizes respectively on the Corsican Fire dataset

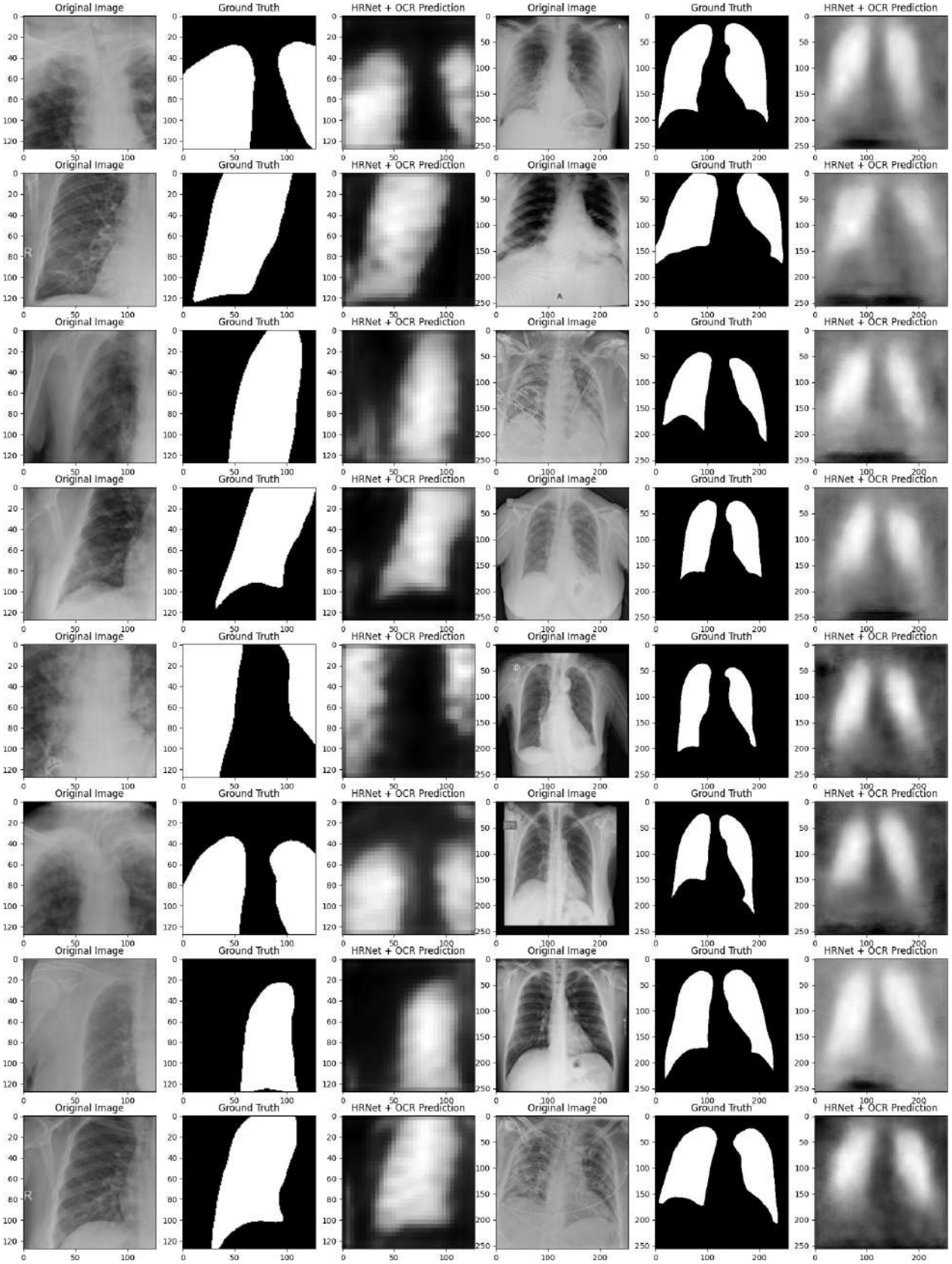

Figure 6.4.10: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively on the COVID-QU-Ex dataset

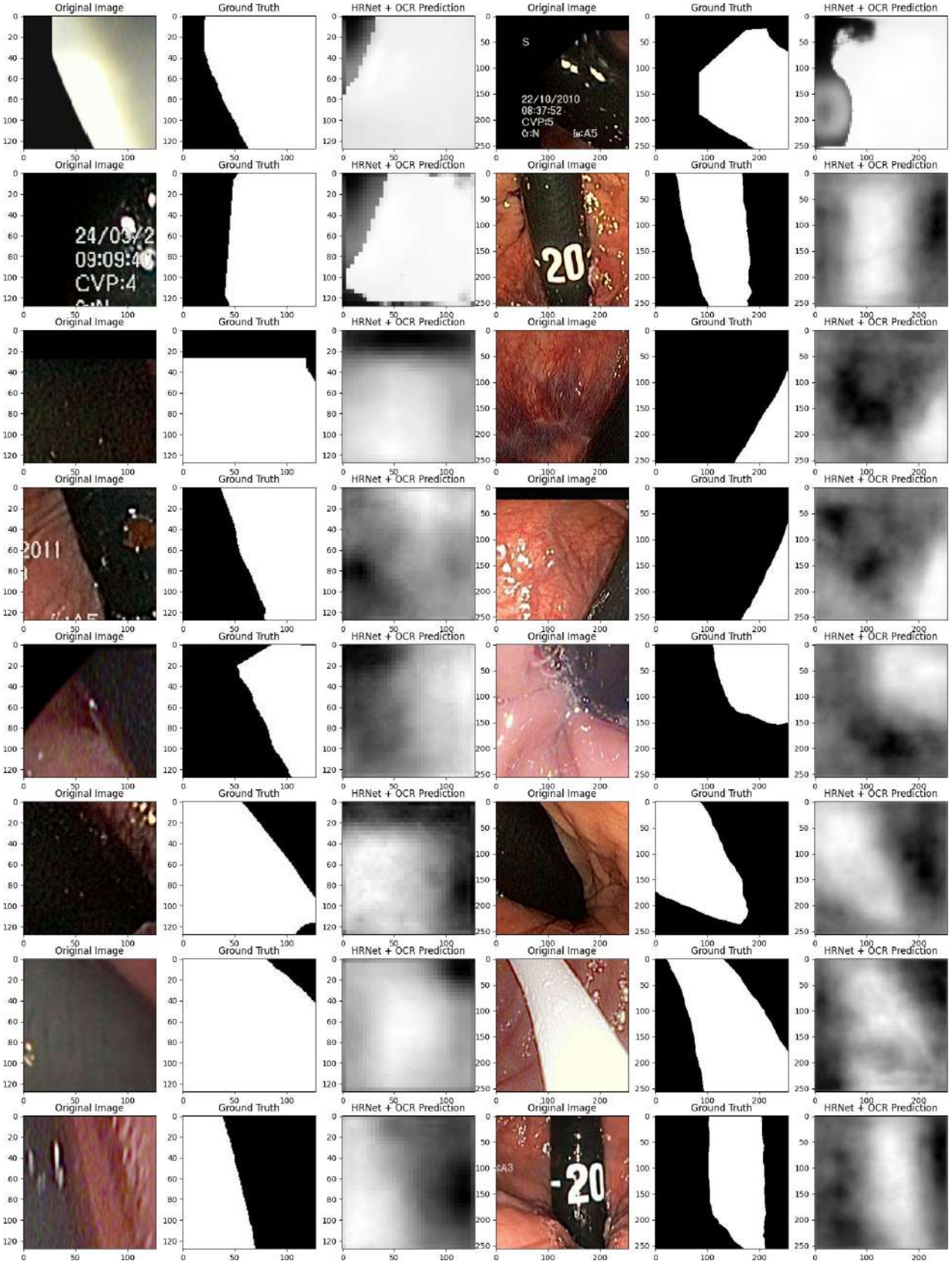

Figure 6.4.11: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Kvasir-Instrument dataset

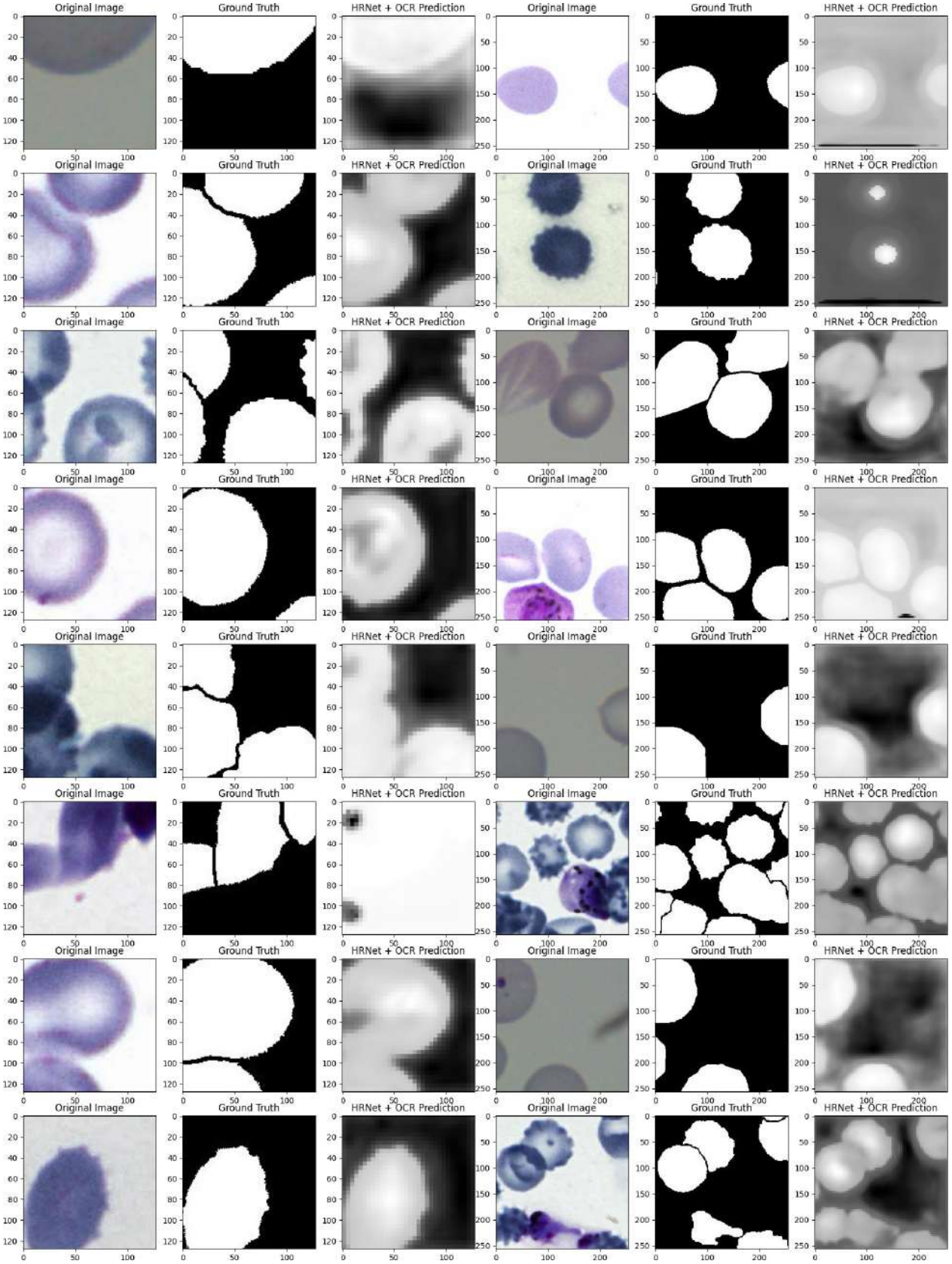

Figure 6.4.12: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Cell dataset

6.4.4 Qualitative Results of PIDNet

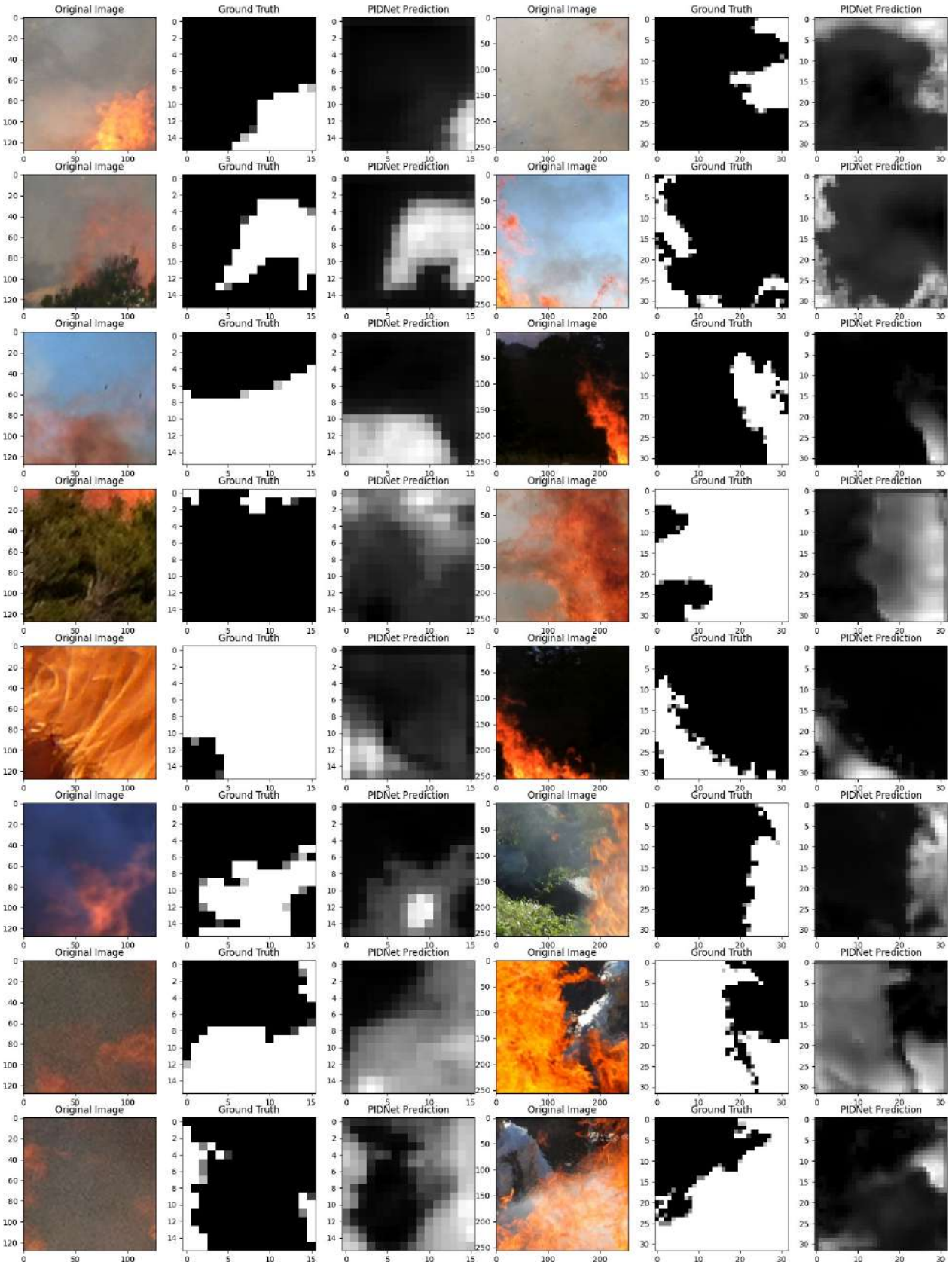

Figure 6.4.13: The original, mask and prediction for 128×128 and 256×256 image sizes respectively on the Corsican Fire dataset

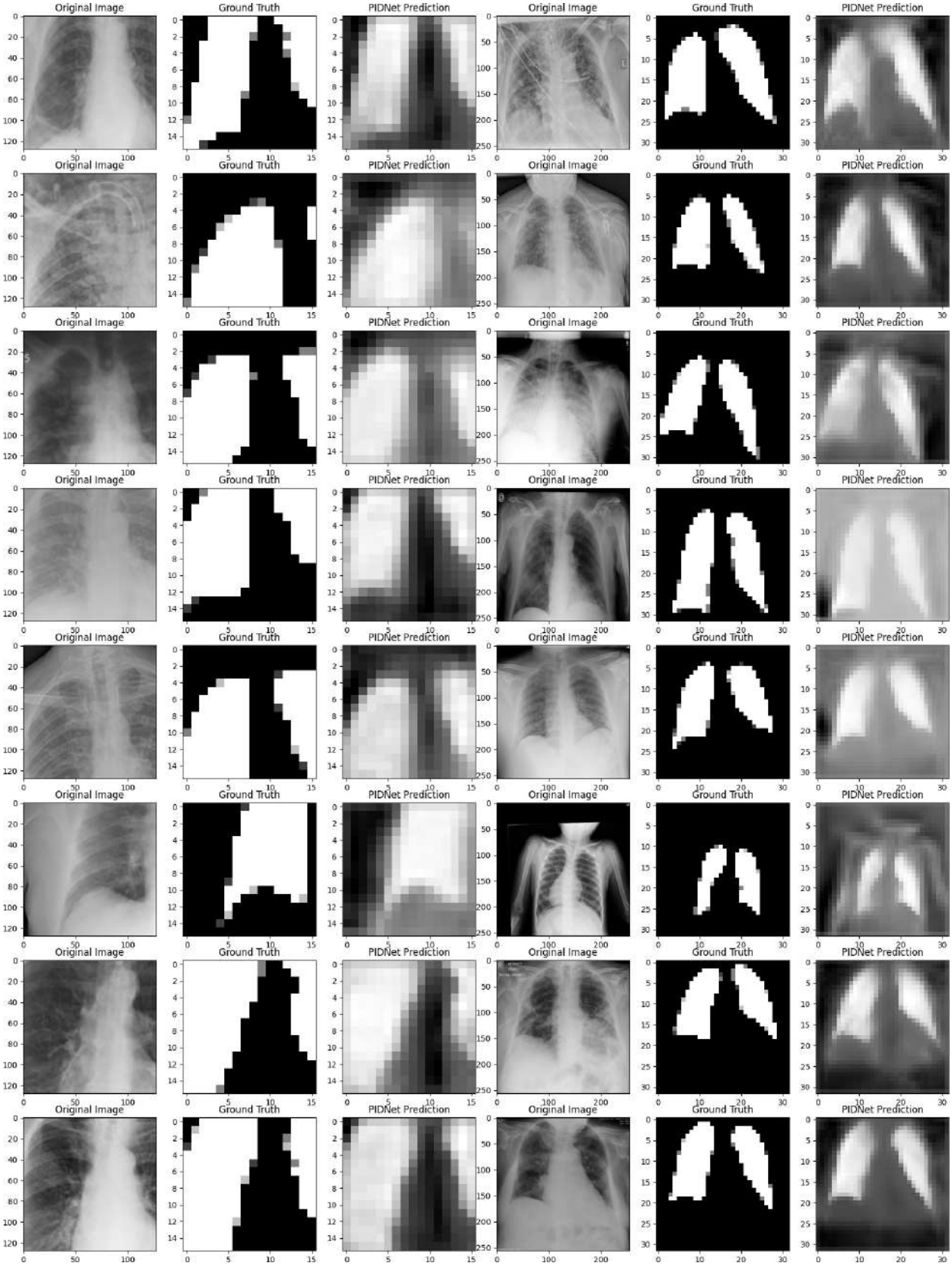

Figure 6.4.14: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively on the COVID-QU-Ex dataset

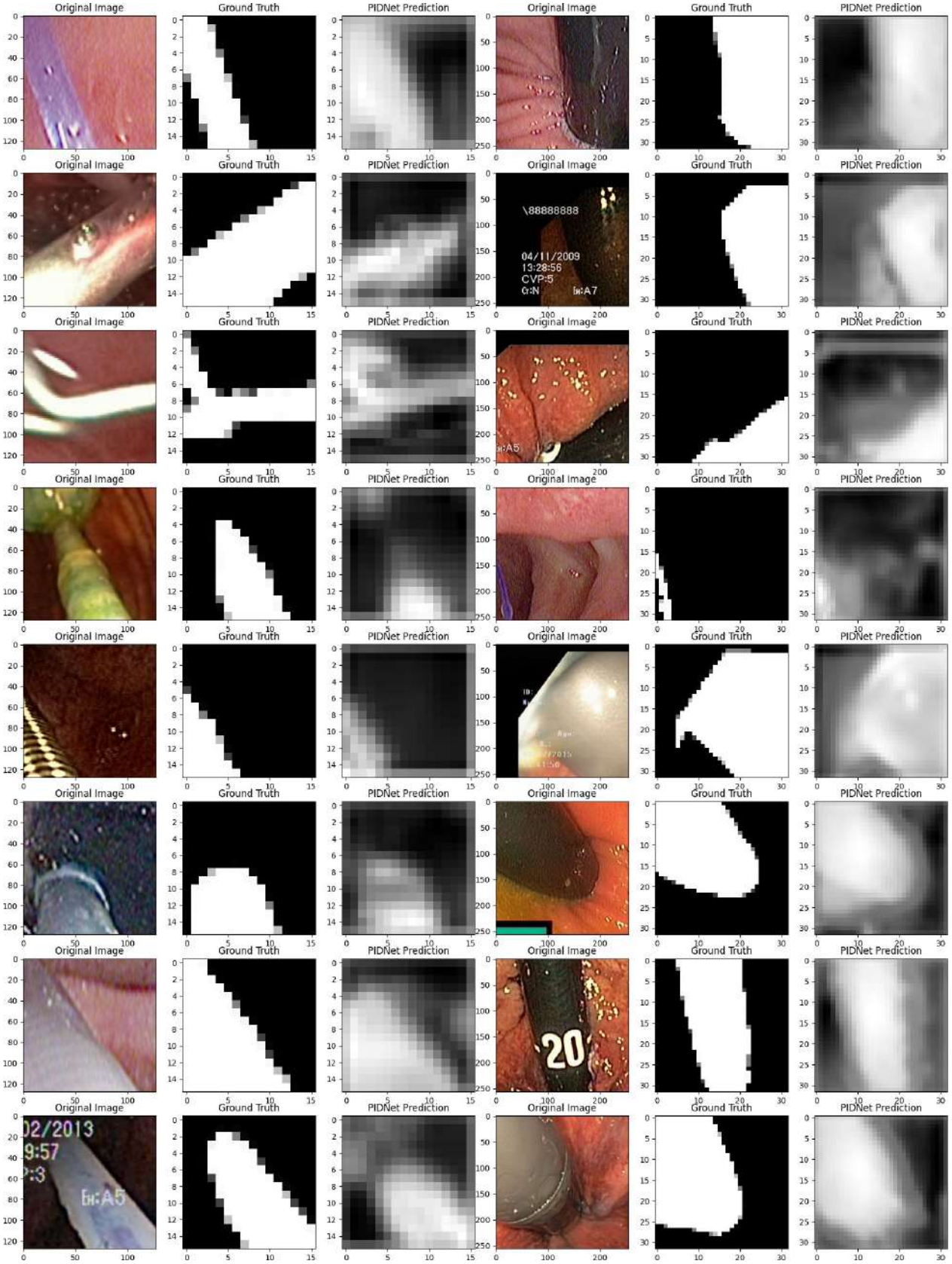

Figure 6.4.15: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Kvasir-Instrument dataset

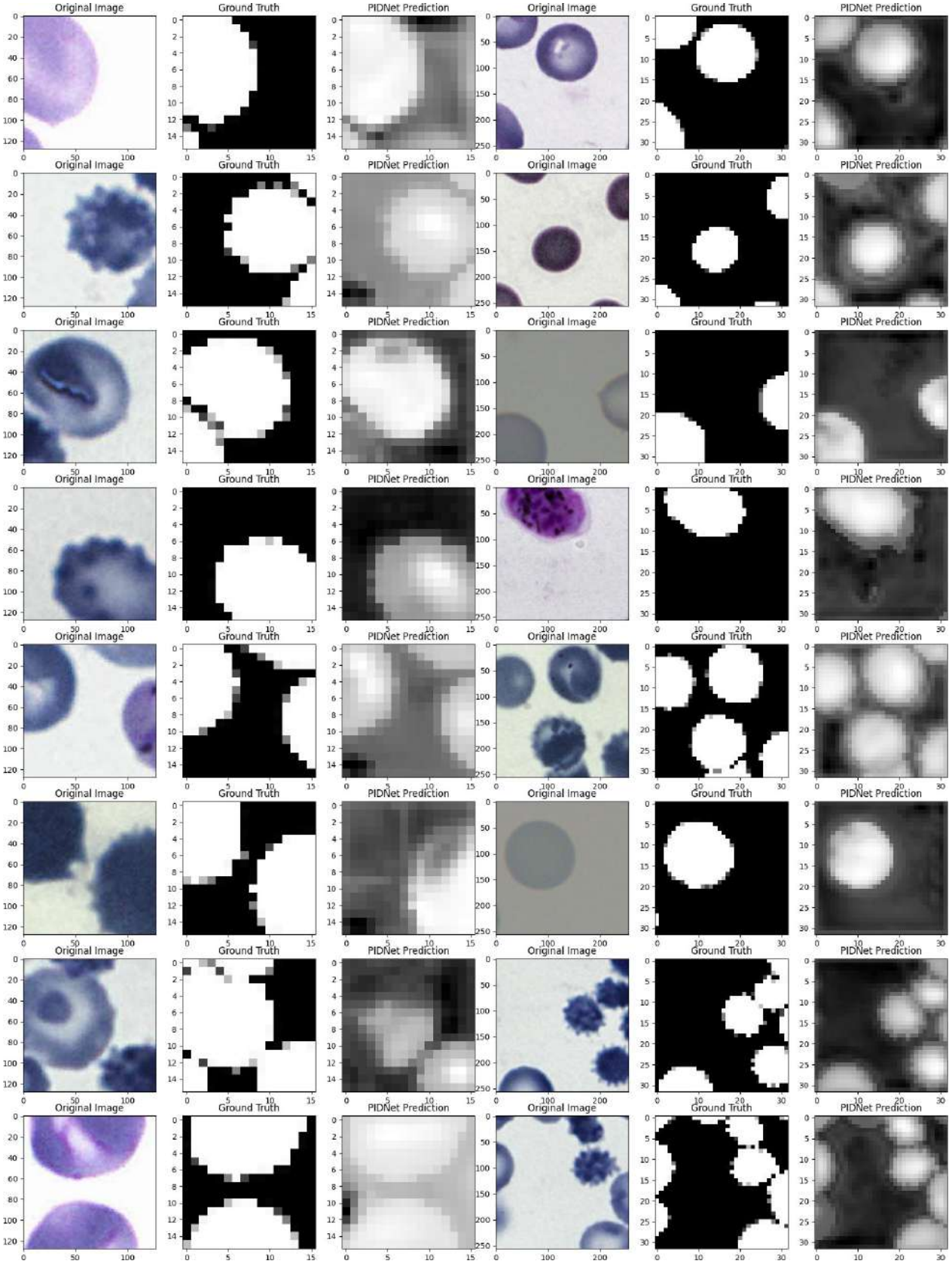

Figure 6.4.16: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Cell dataset

6.4.5 Qualitative Results of DDRNet

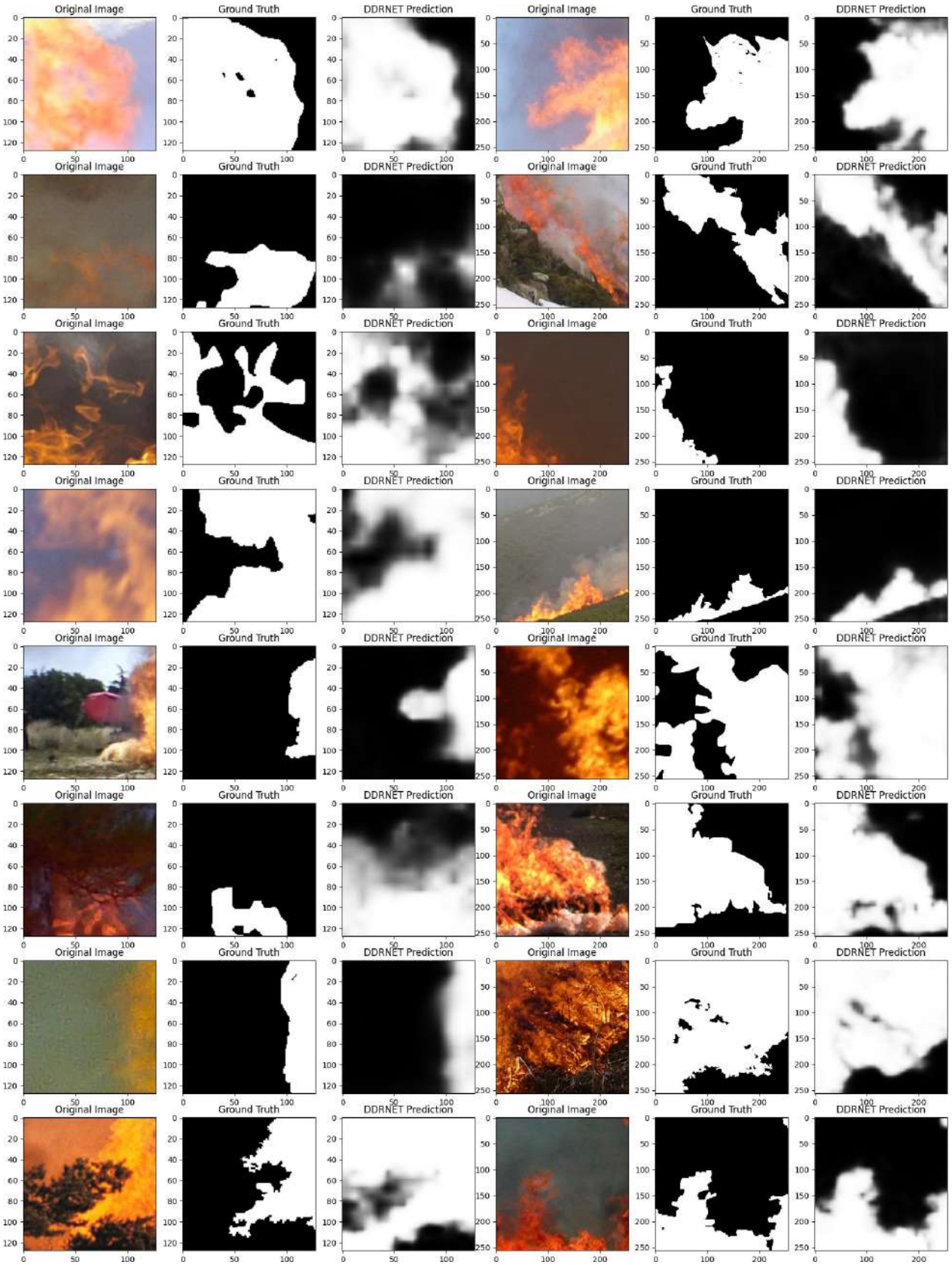

Figure 6.4.17: The original, mask and prediction for 128×128 and 256×256 image sizes respectively on the Corsican Fire dataset

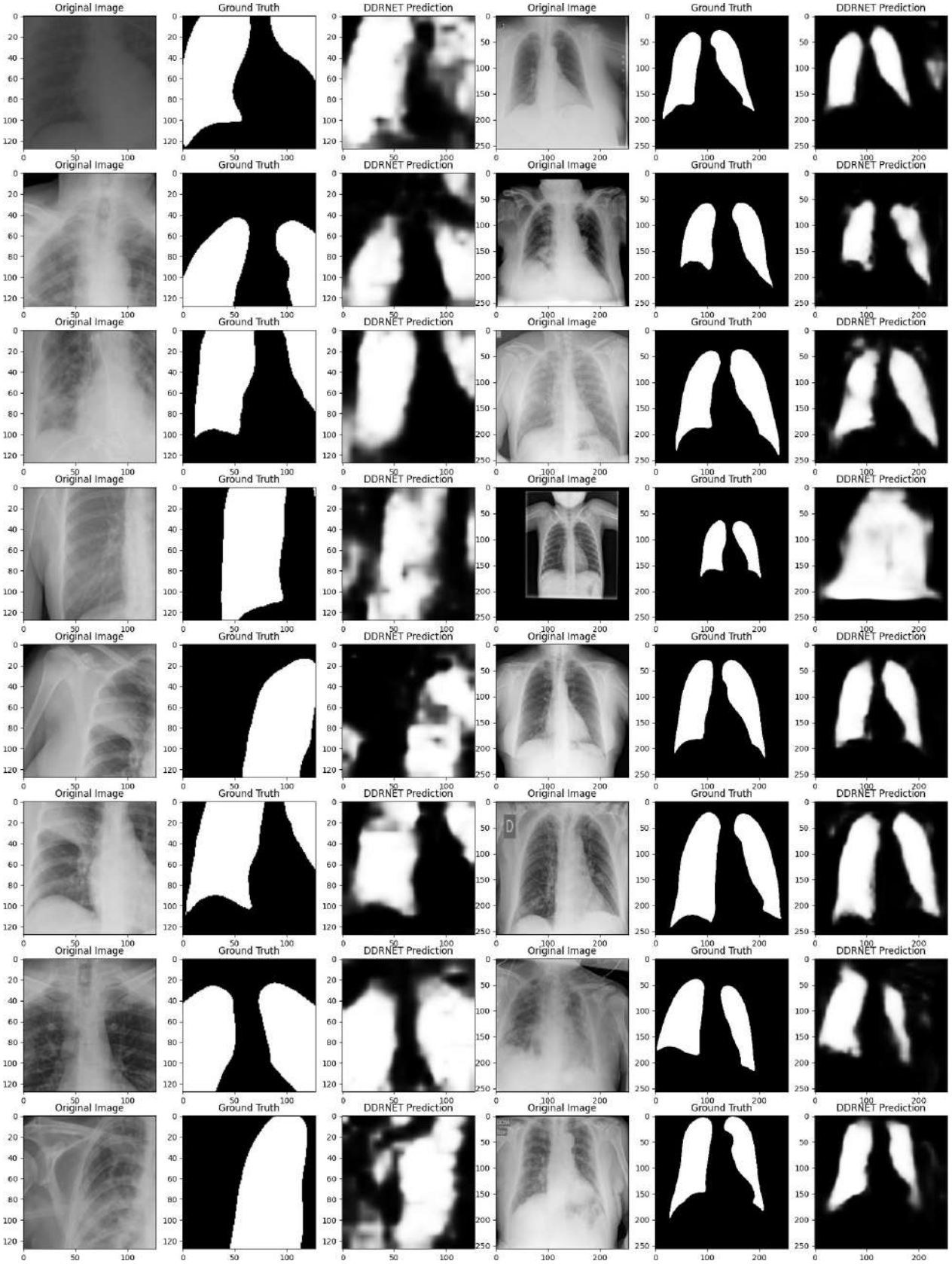

Figure 6.4.18: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively on the COVID-QU-Ex dataset

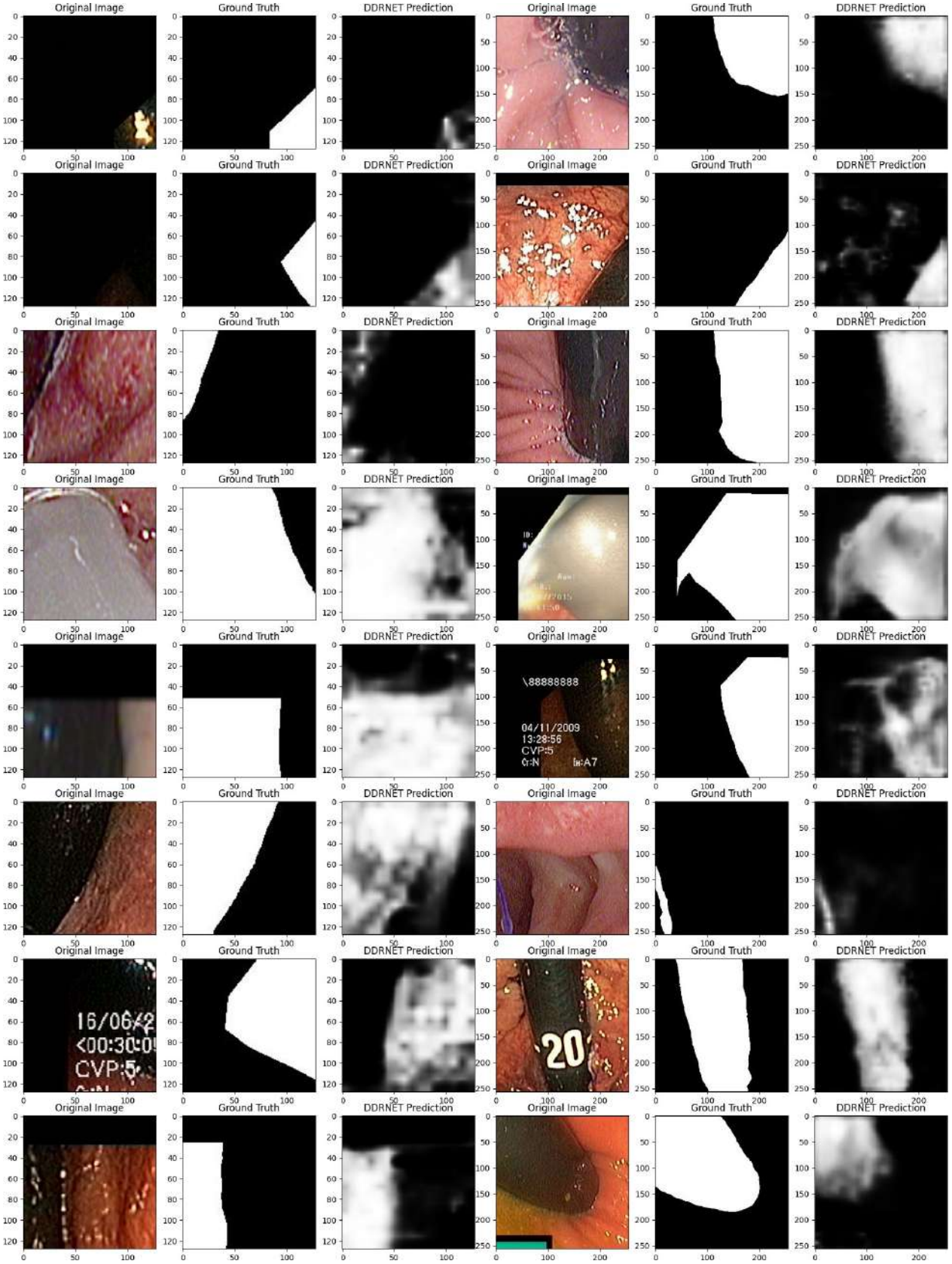

Figure 6.4.19: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Kvasir-Instrument dataset

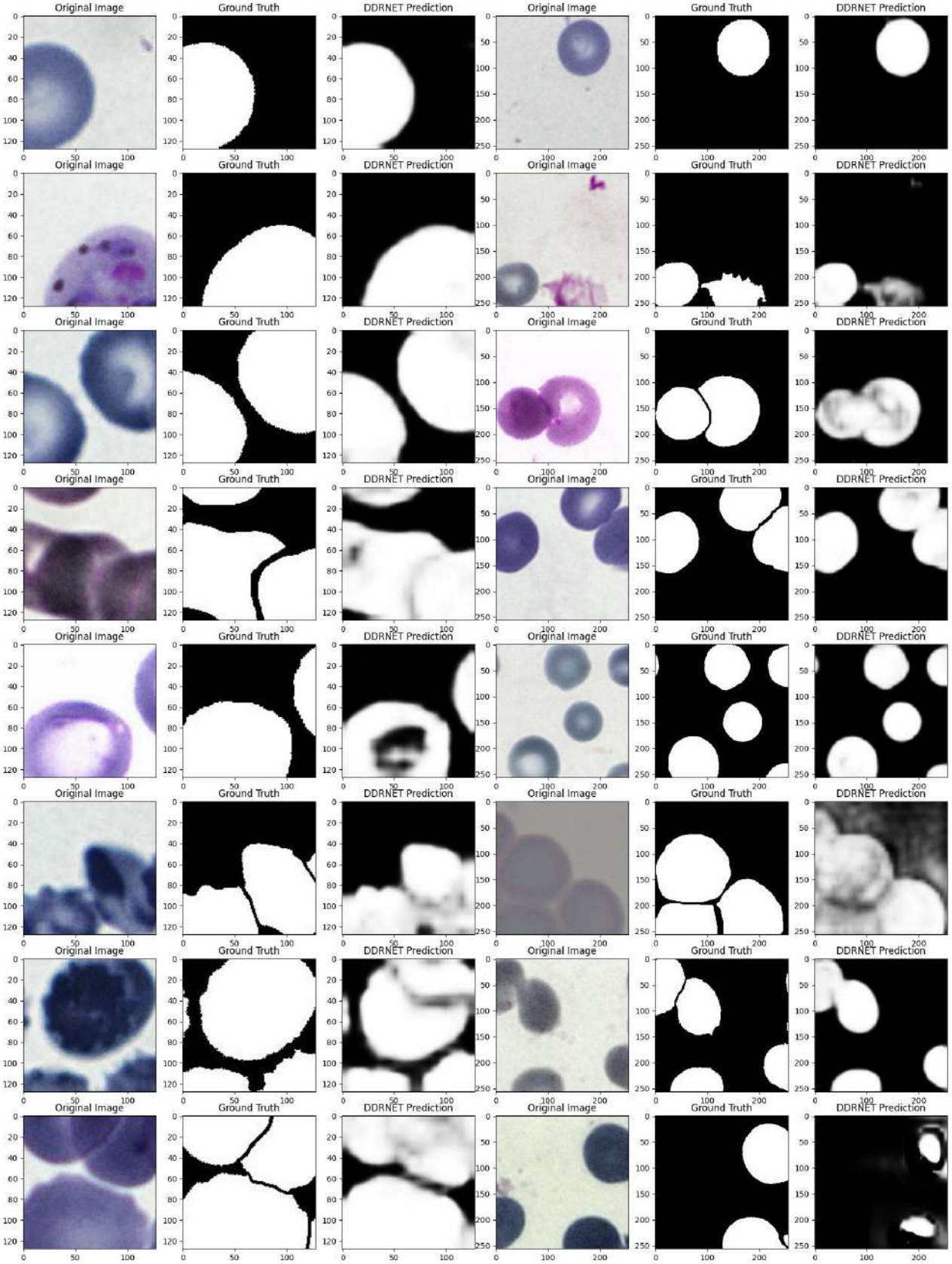

Figure 6.4.20: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Cell dataset

6.4.6 Qualitative Results of Swin Transformer

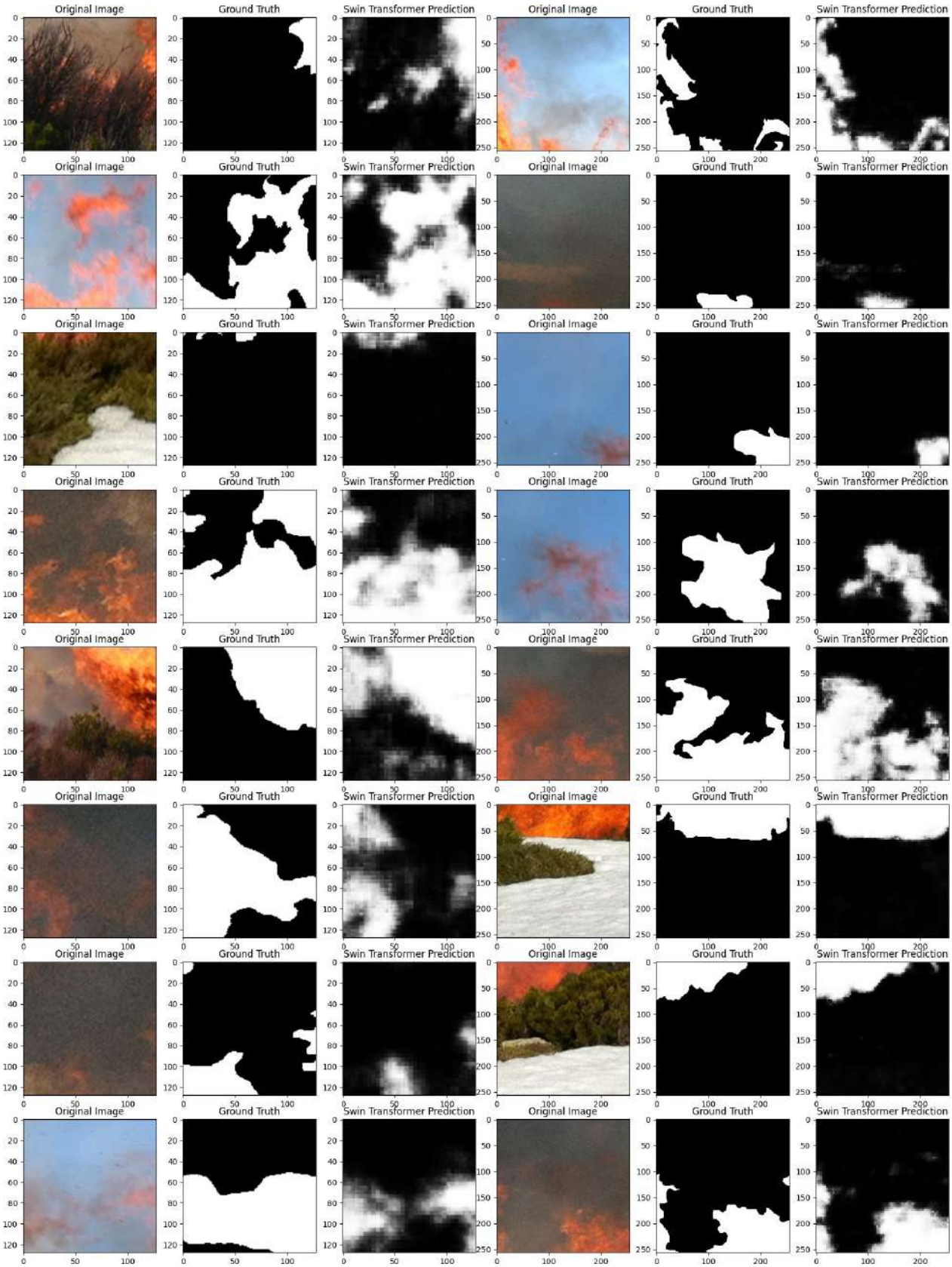

Figure 6.4.21: The original, mask and prediction for 128×128 and 256×256 image sizes respectively on the Corsican Fire dataset

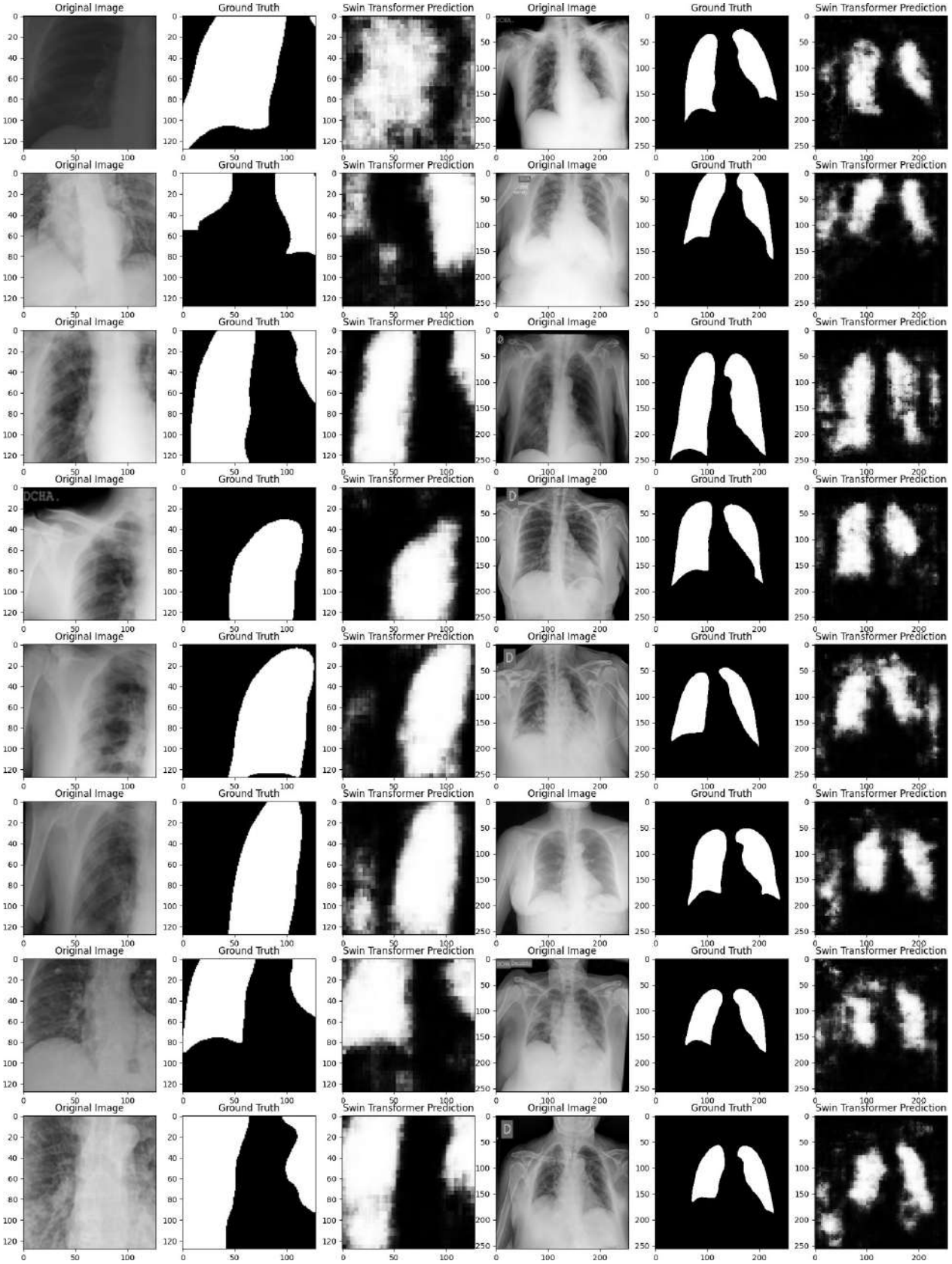

Figure 6.4.22: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively on the COVID-QU-Ex dataset

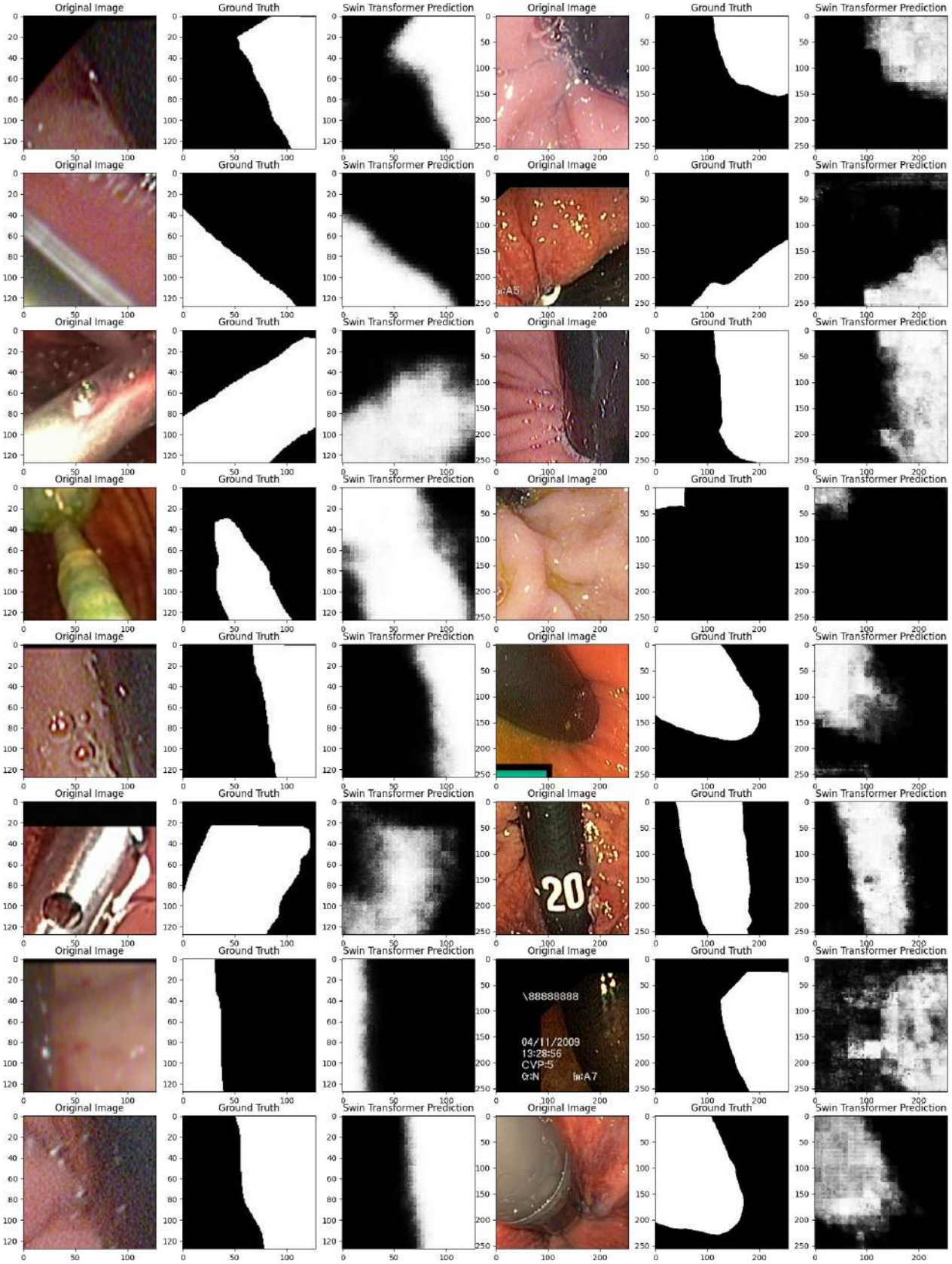

Figure 6.4.23: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Kvasir-Instrument dataset

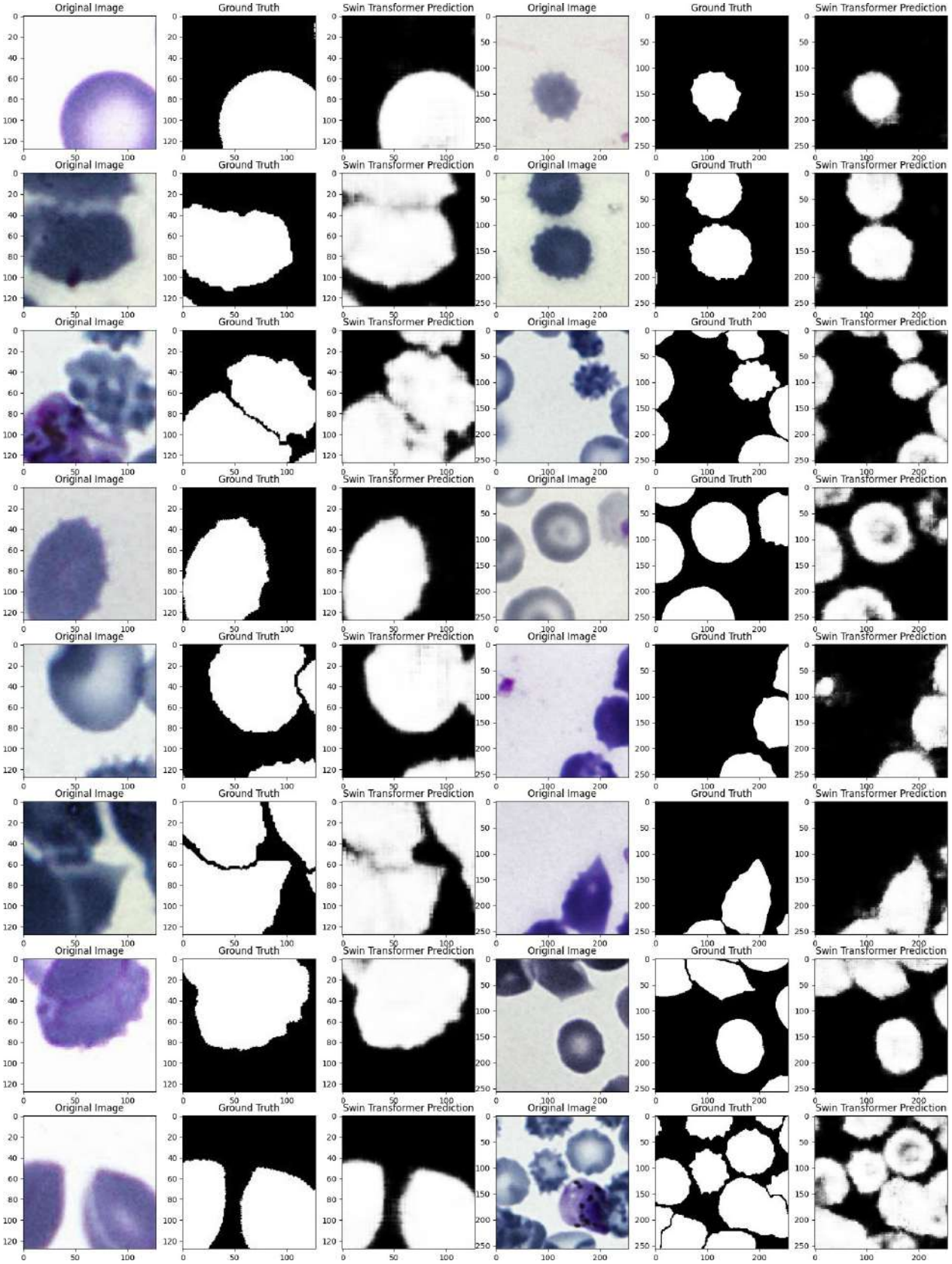

Figure 6.4.24: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Cell dataset

6.4.7 Qualitative Results of Unet3+

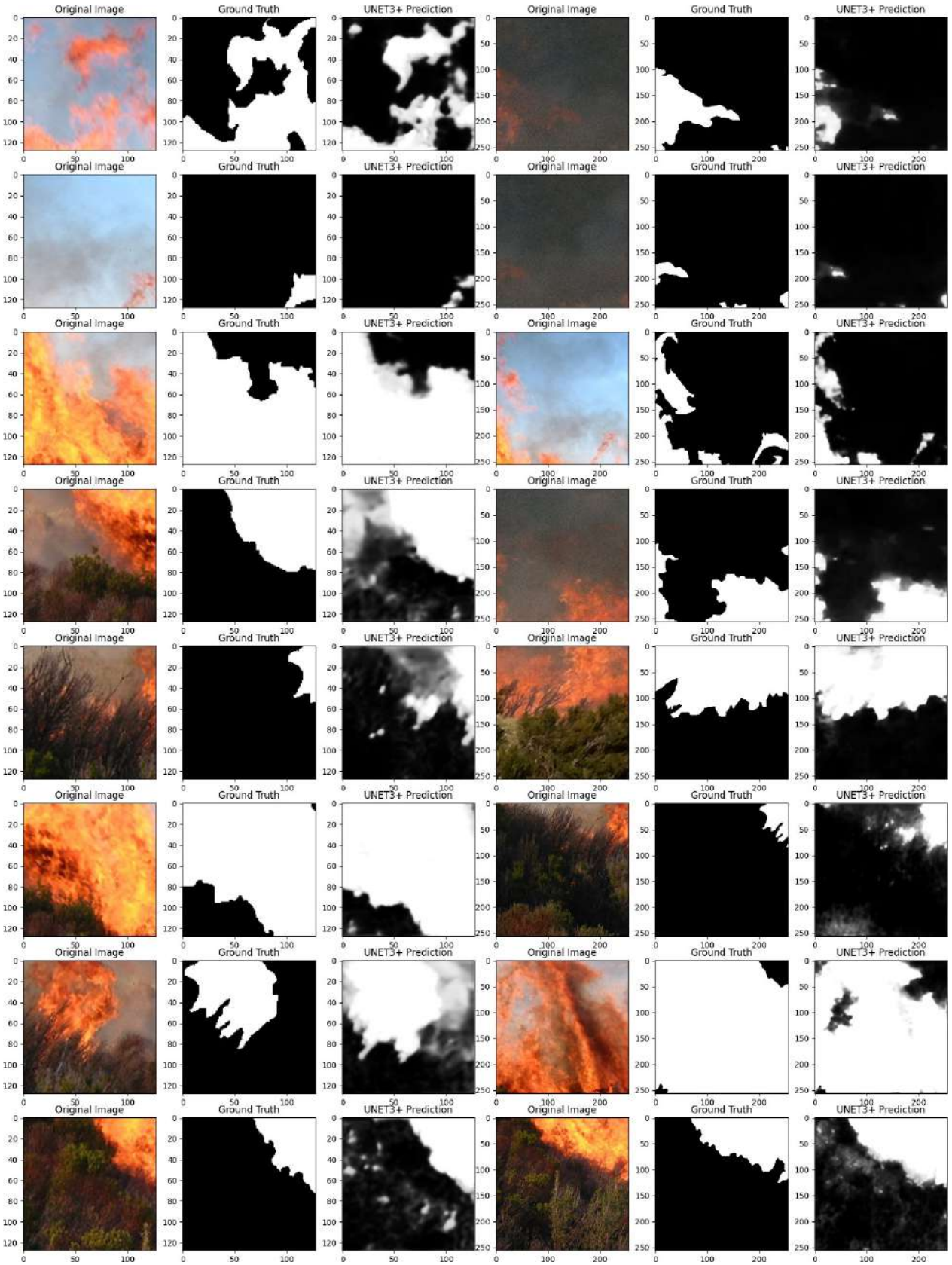

Figure 6.4.25: The original, mask and prediction for 128×128 and 256×256 image sizes respectively on the Corsican Fire dataset

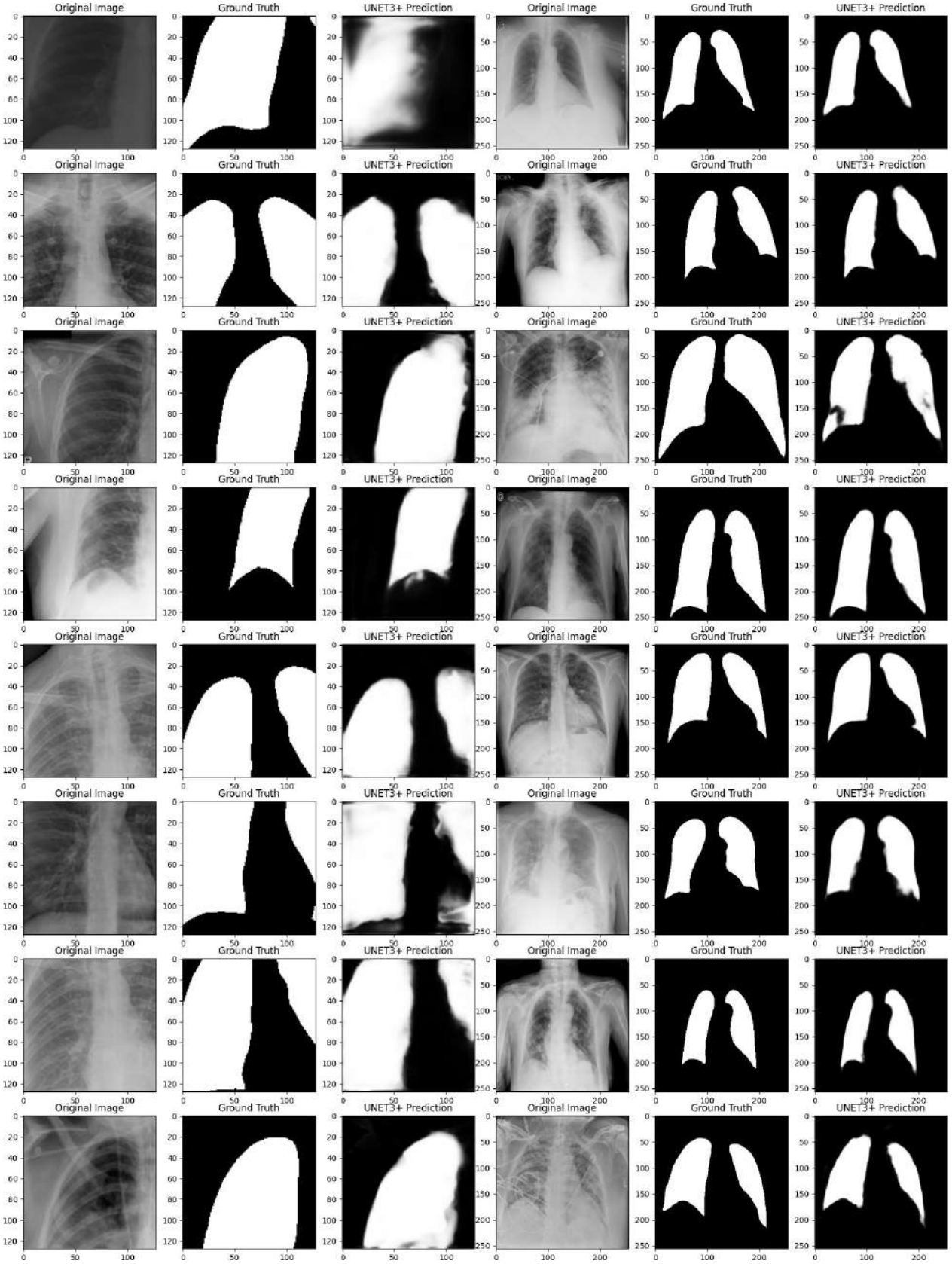

Figure 6.4.26: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively on the COVID-QU-Ex dataset

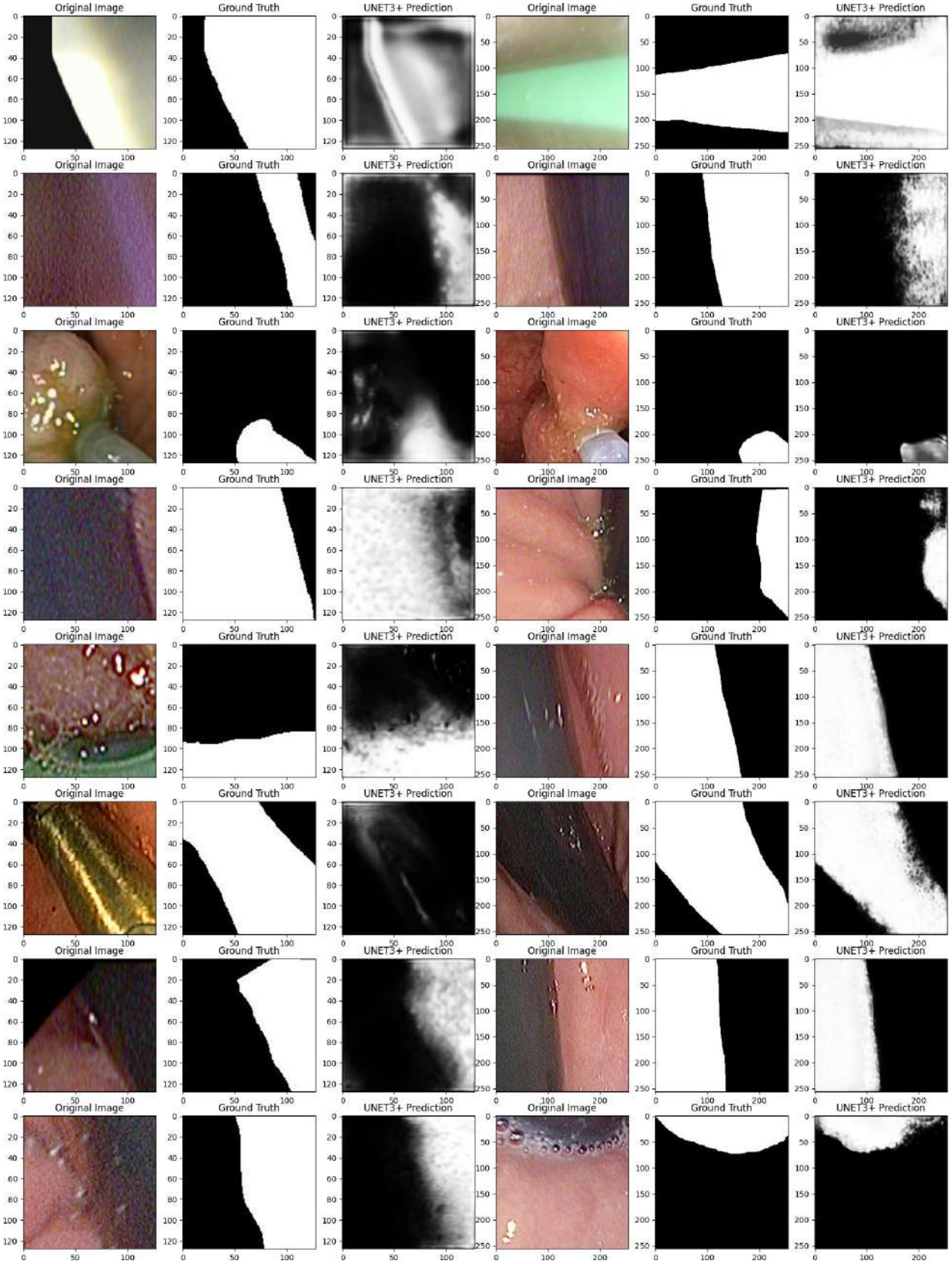

Figure 6.4.27: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Kvasir-Instrument dataset

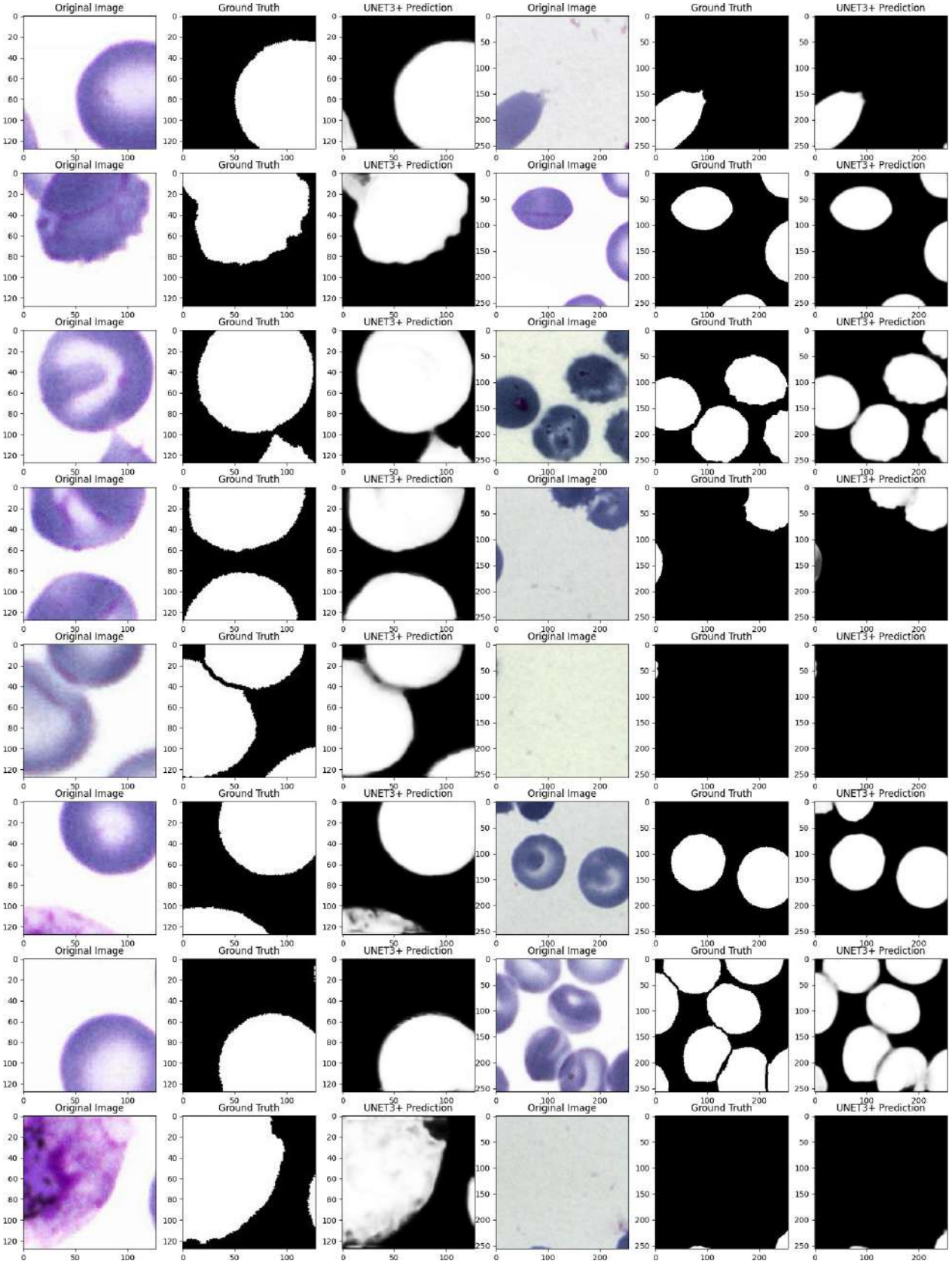

Figure 6.4.28: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Cell dataset

6.4.8 Qualitative Results of BASNet

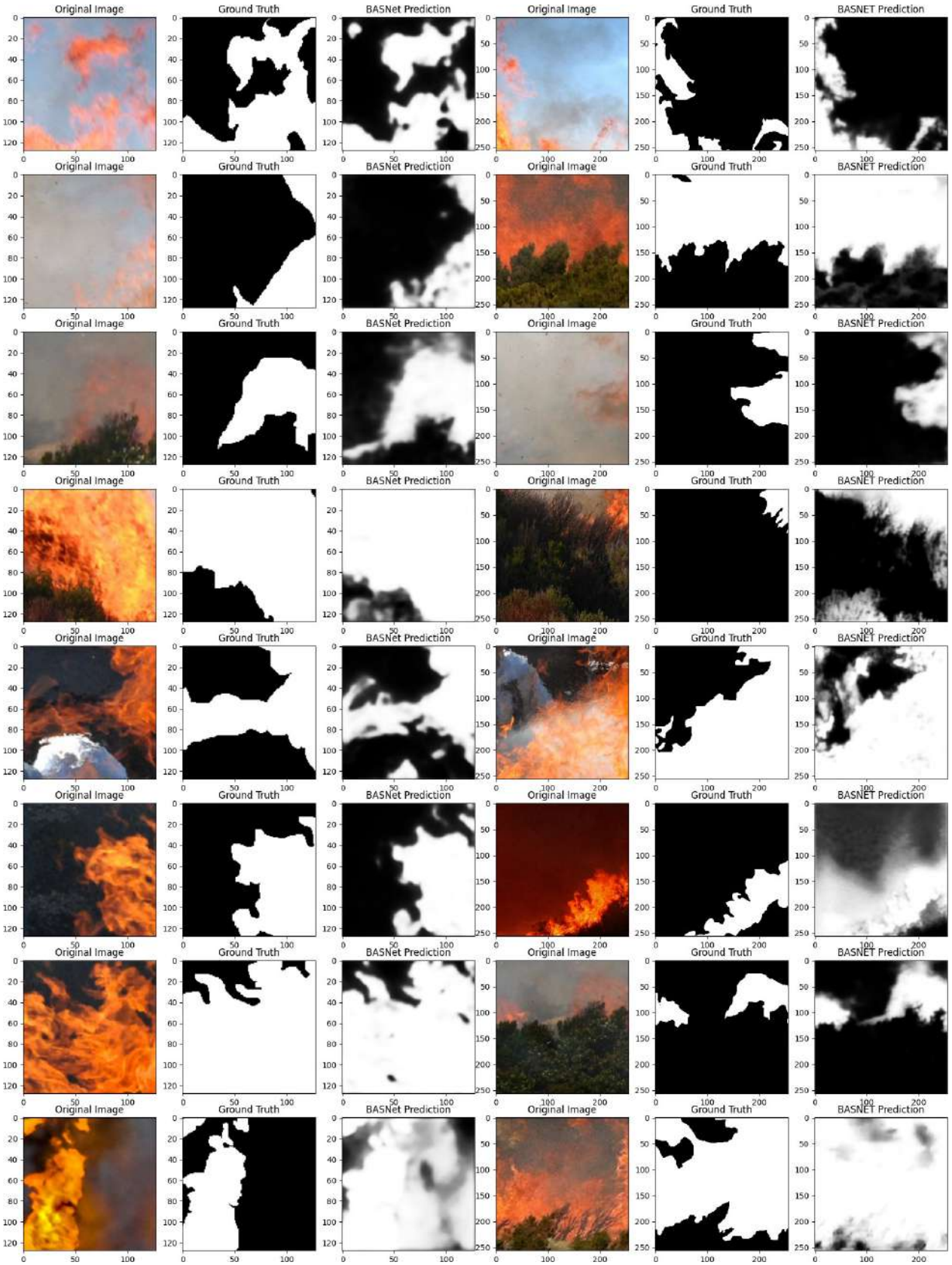

Figure 6.4.29: The original, mask and prediction for 128×128 and 256×256 image sizes respectively on the Corsican Fire dataset

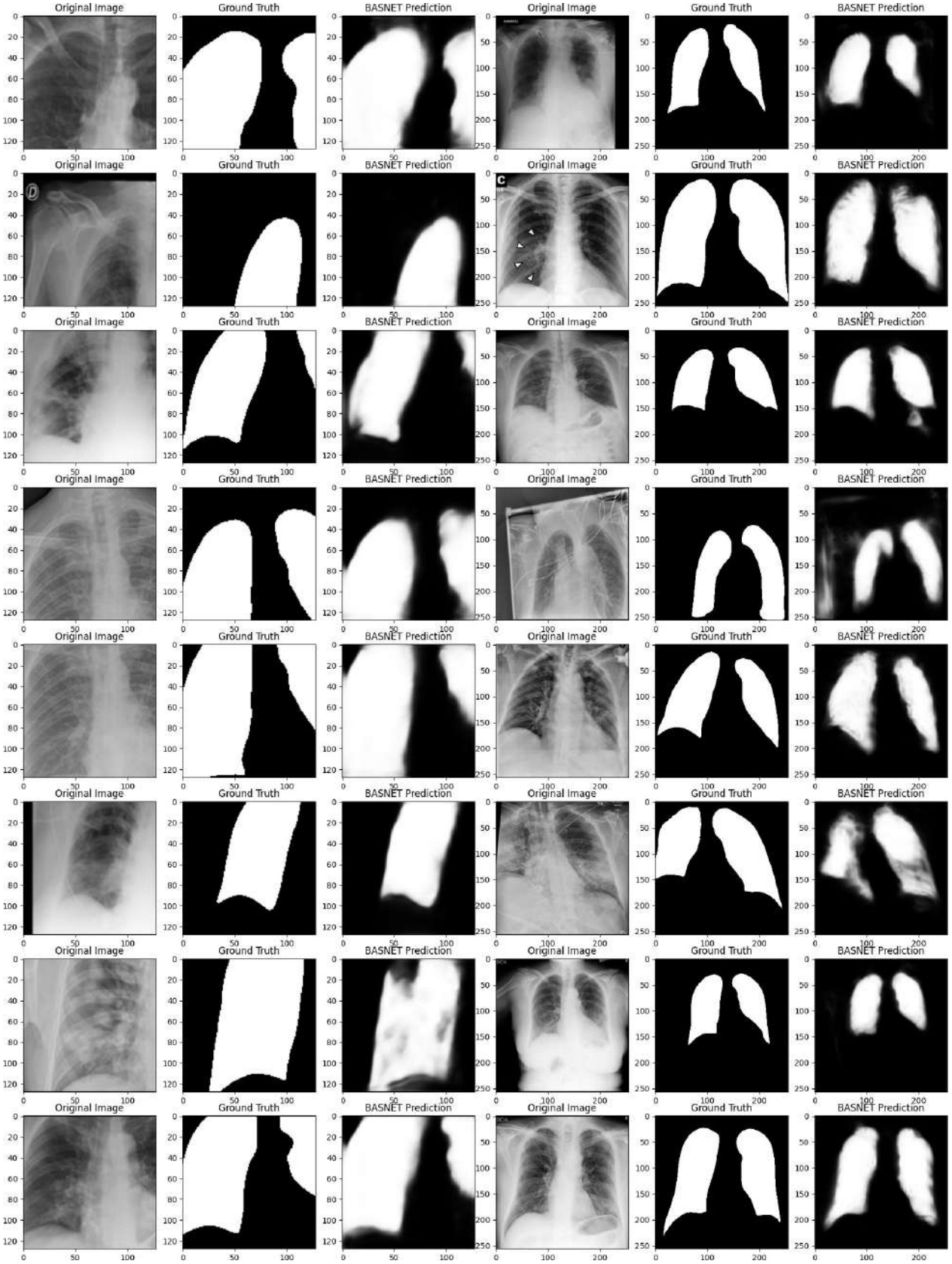

Figure 6.4.30: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively on the COVID-QU-Ex dataset

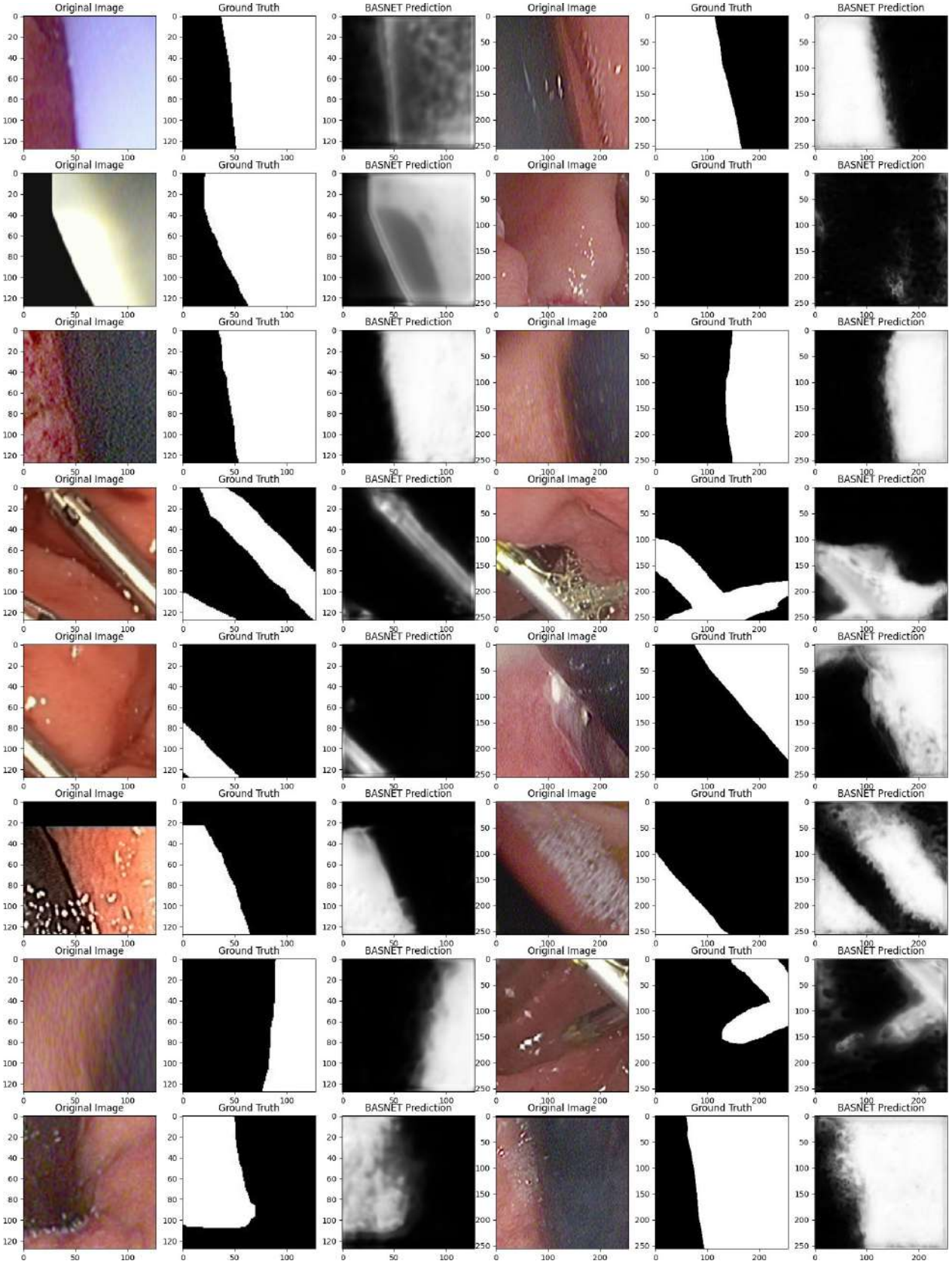

Figure 6.4.31: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Kvasir-Instrument dataset

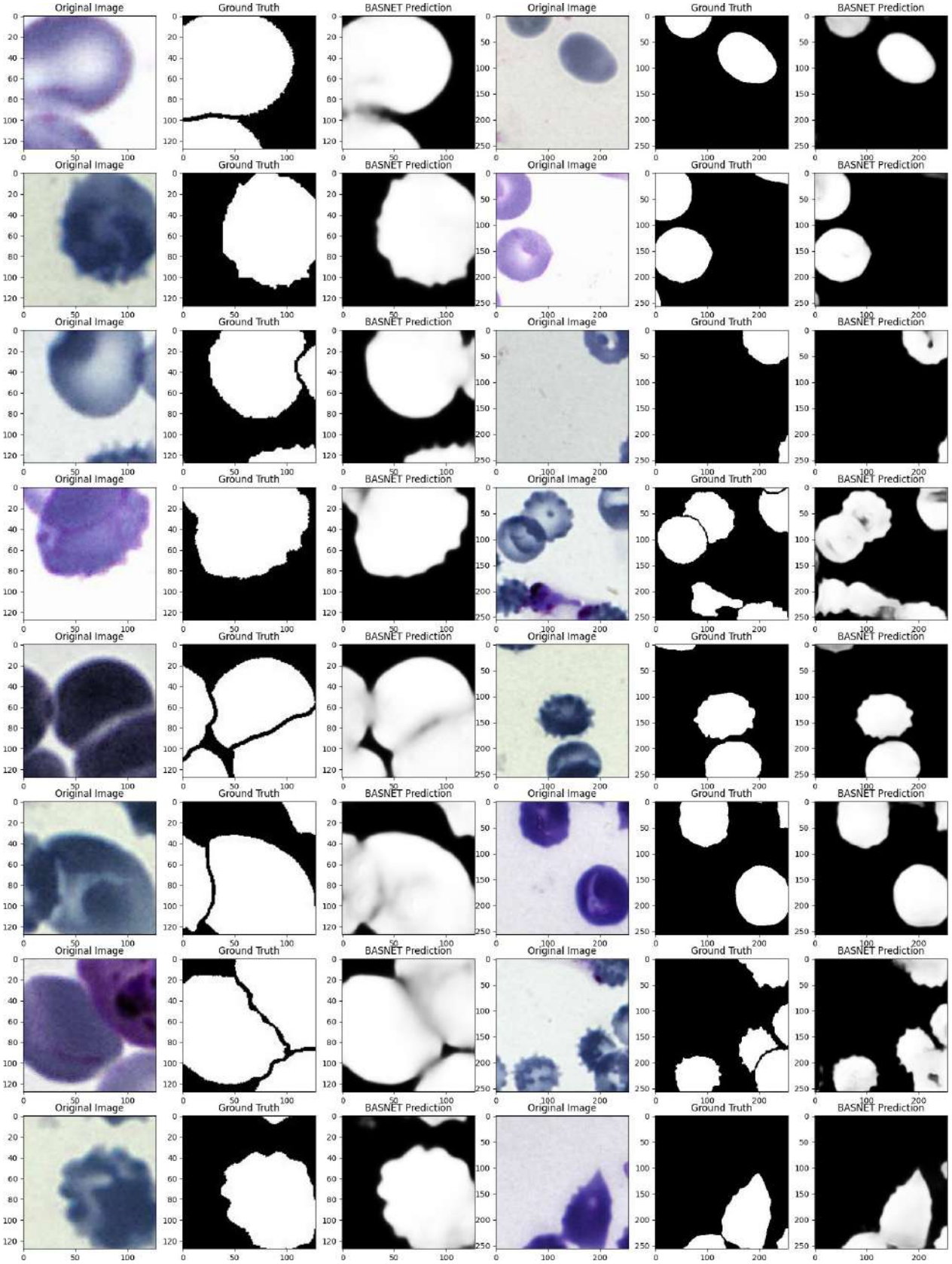

Figure 6.4.32: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Cell dataset

6.4.9 BiSeNet Qualitative Results

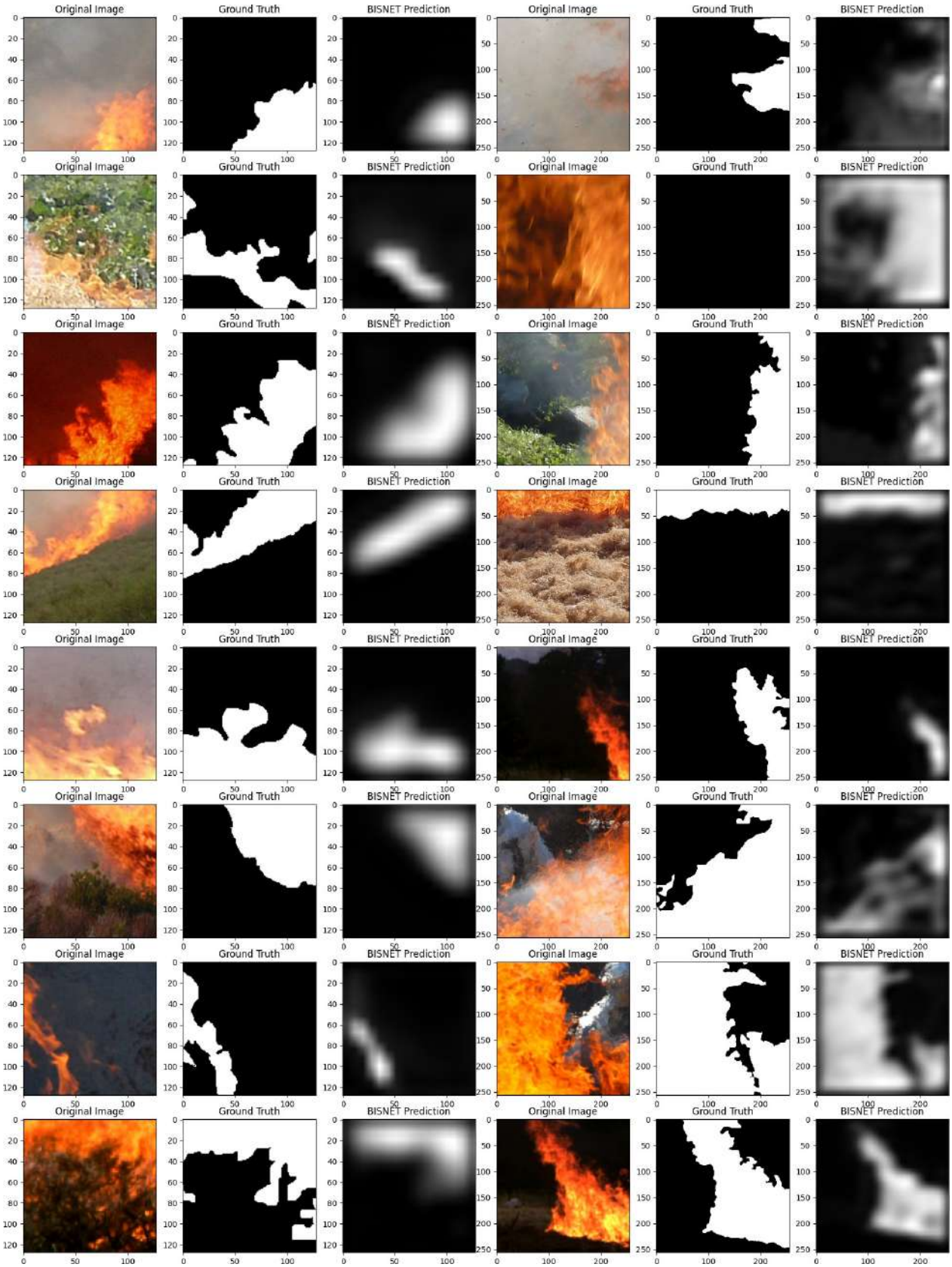

Figure 6.4.33: The original, mask and prediction for 128×128 and 256×256 image sizes respectively on the Corsican Fire dataset

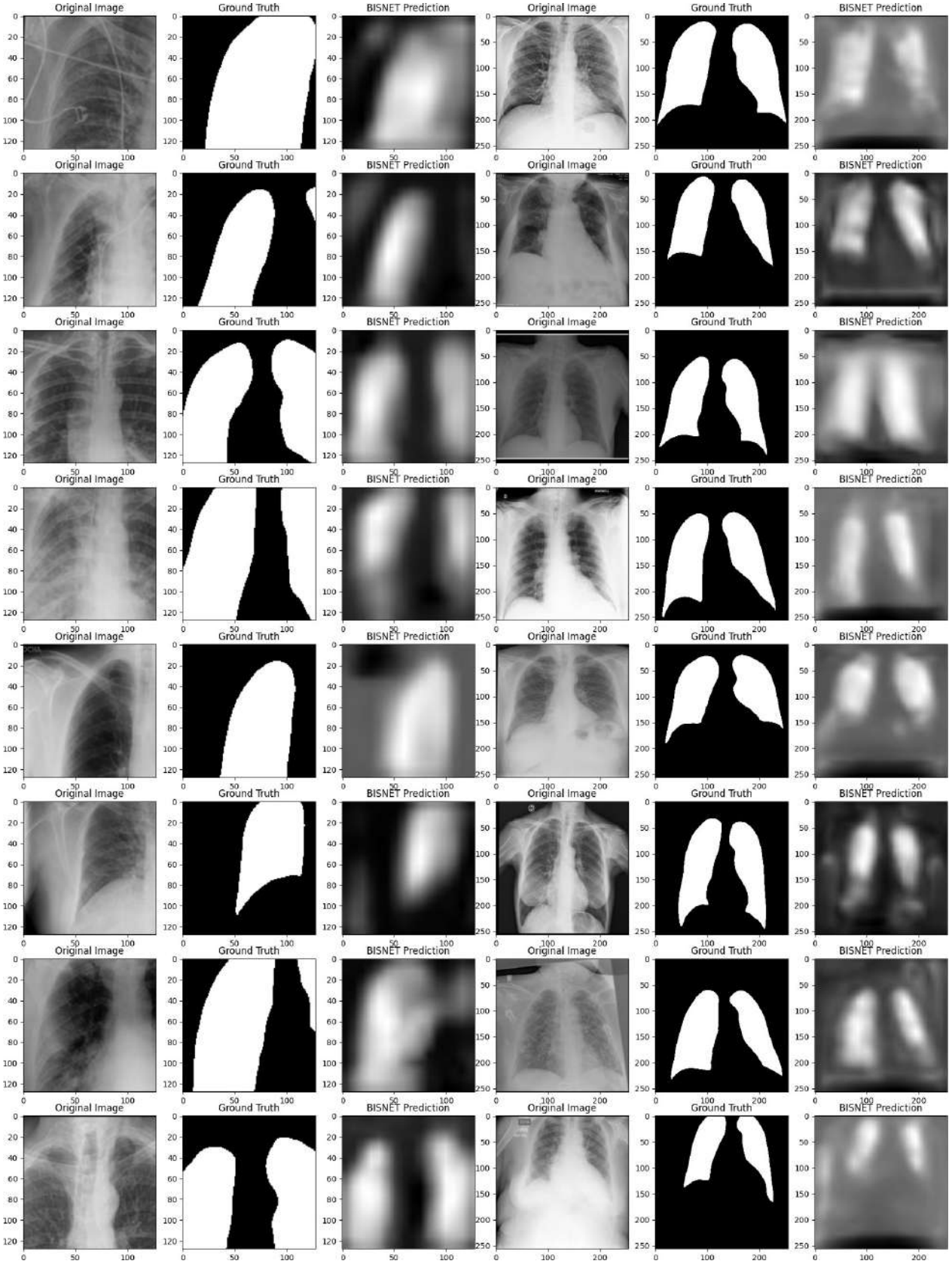

Figure 6.4.34: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively on the COVID-QU-Ex dataset

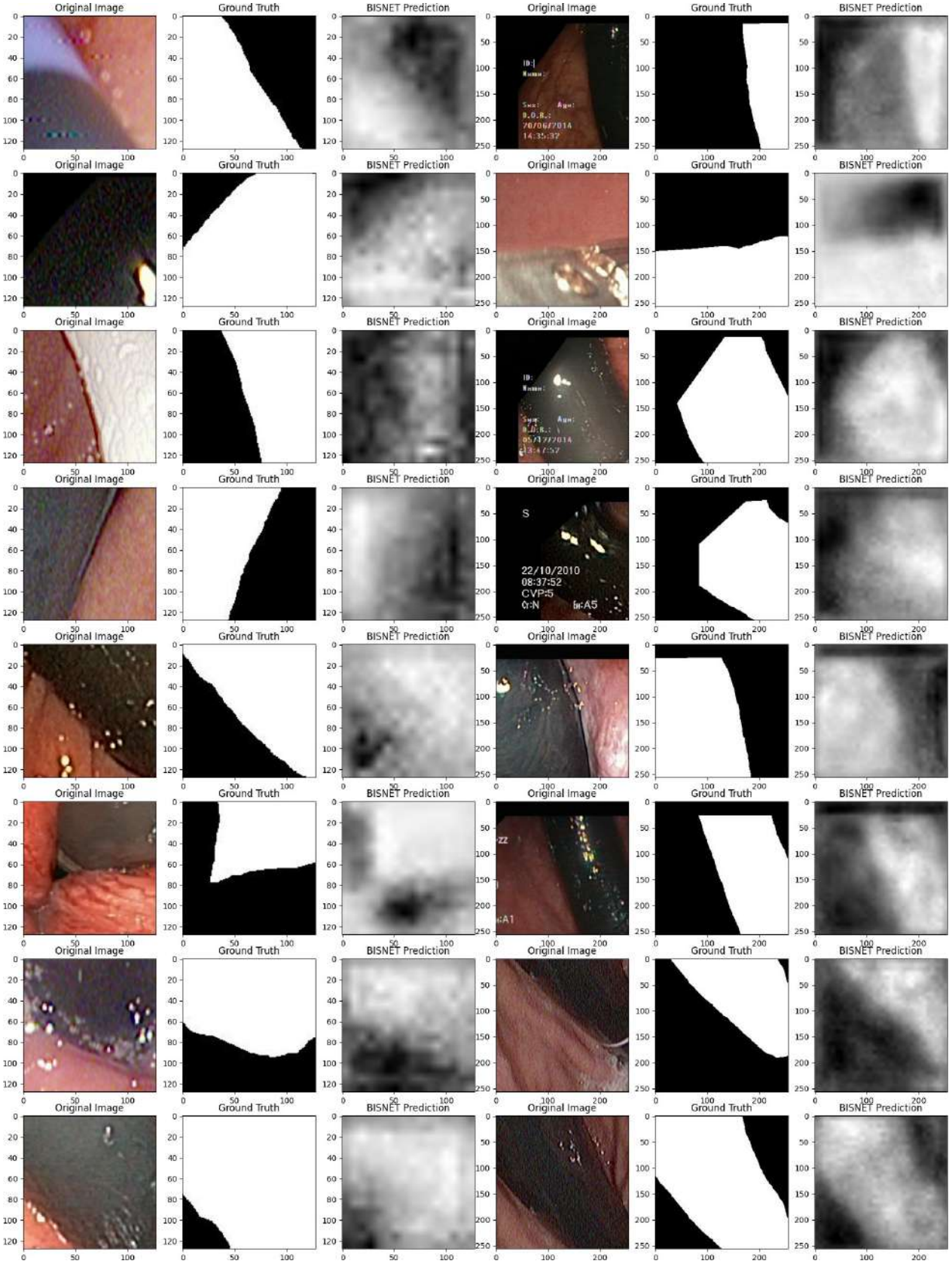

Figure 6.4.35: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Kvasir-Instrument dataset

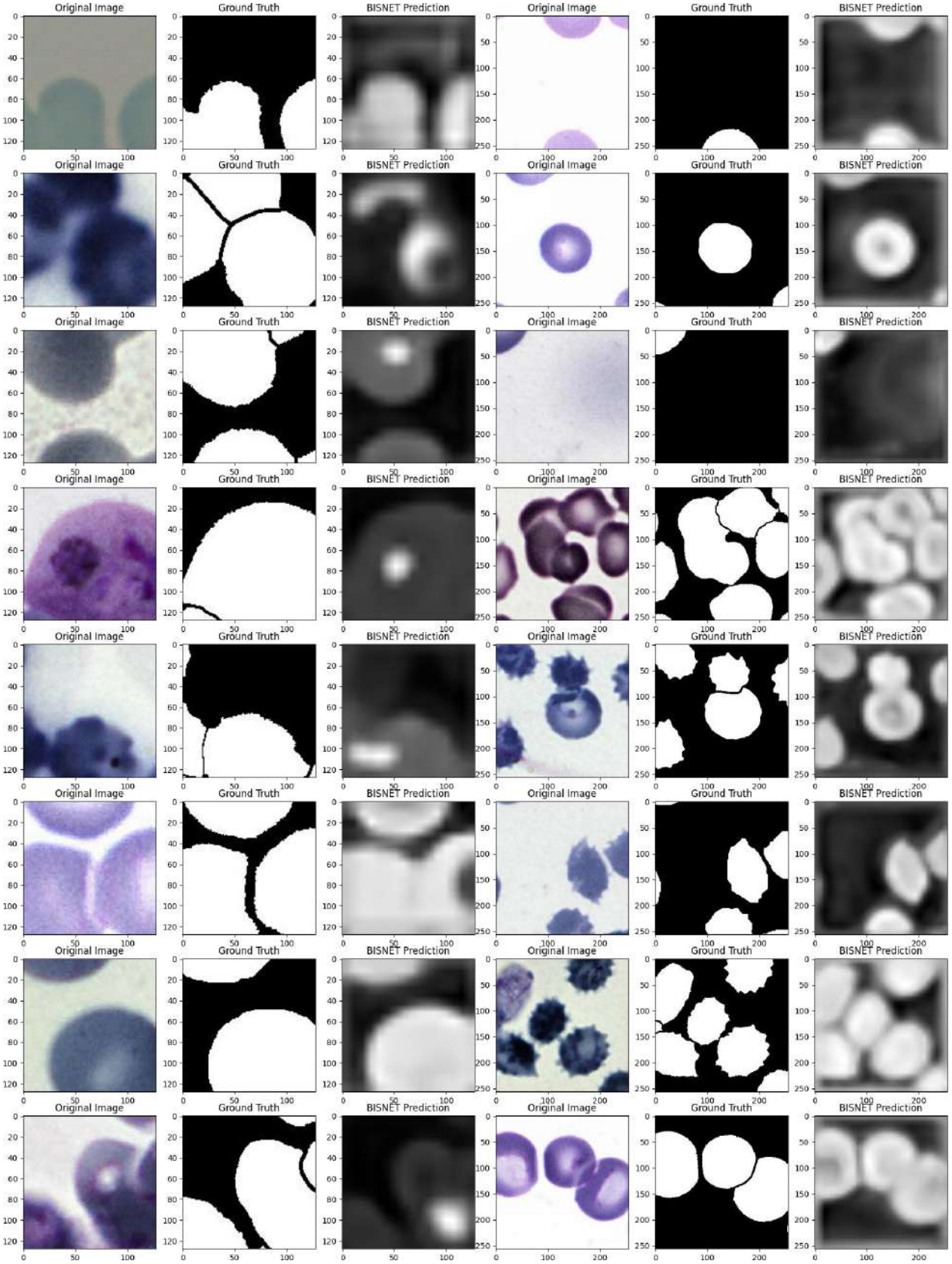

Figure 6.4.36: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Cell dataset

6.4.10 Qualitative Results of EmaNet

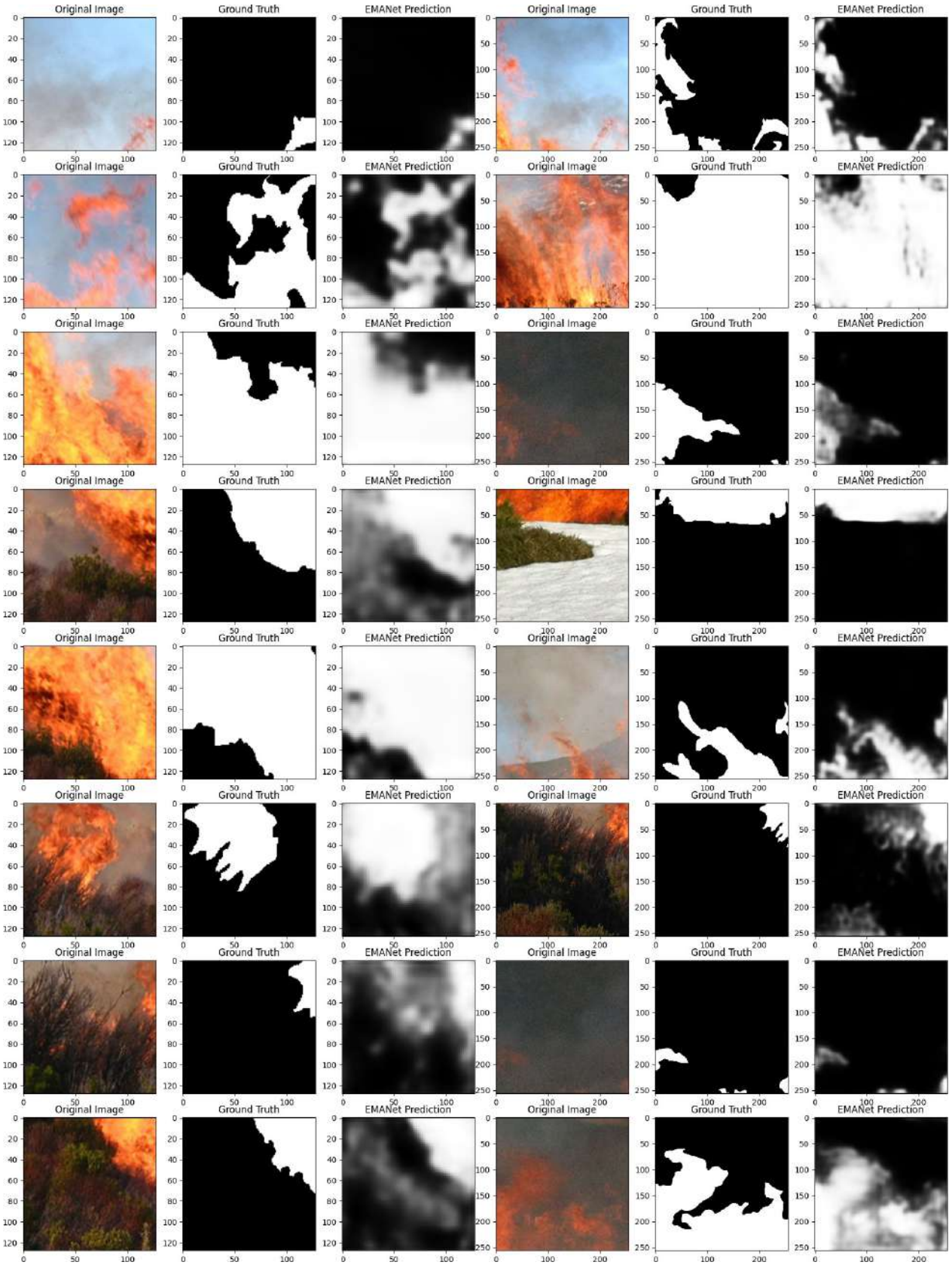

Figure 6.4.37: The original, mask and prediction for 128×128 and 256×256 image sizes respectively on the Corsican Fire dataset

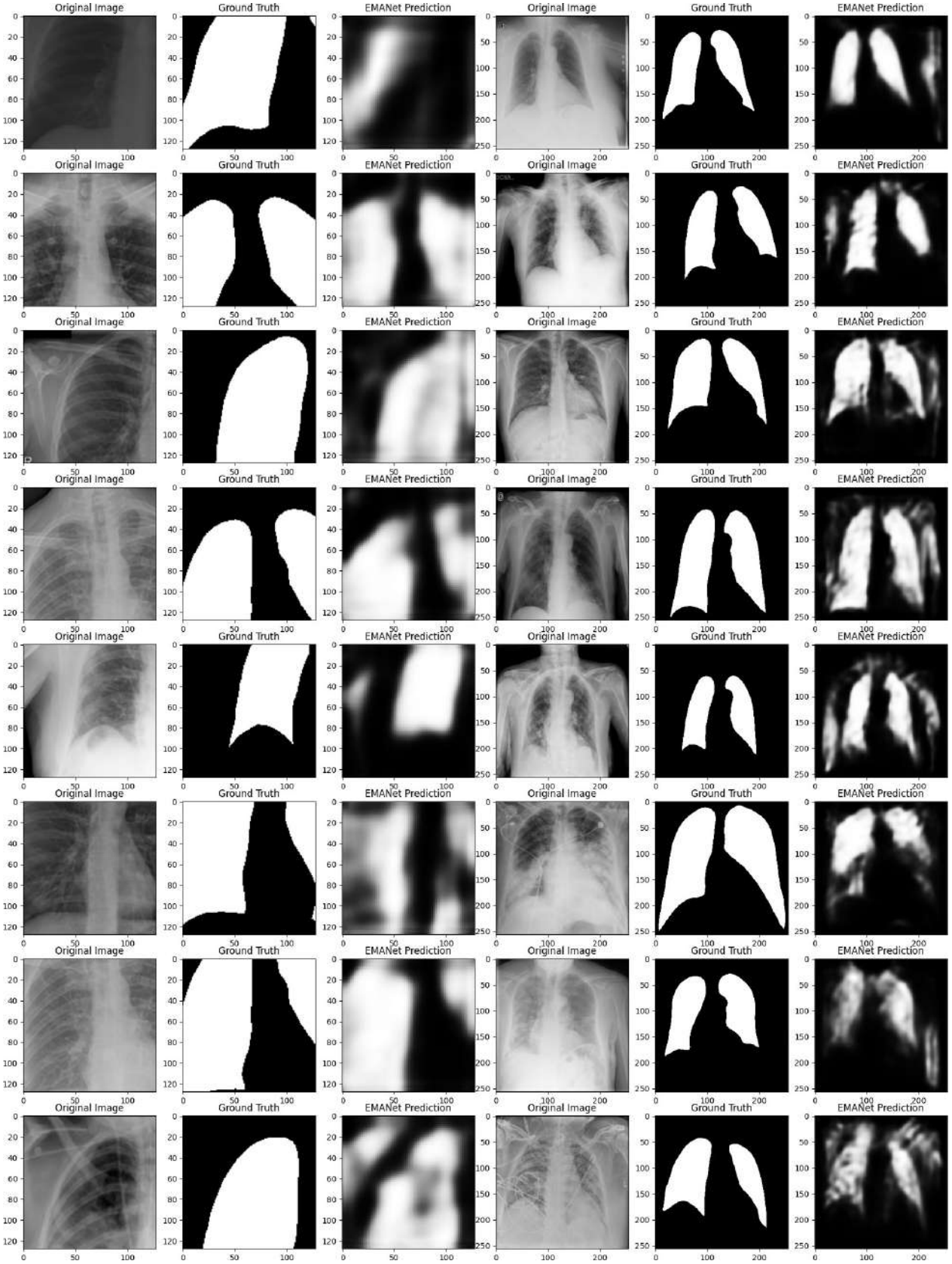

Figure 6.4.38: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively on the COVID-QU-Ex dataset

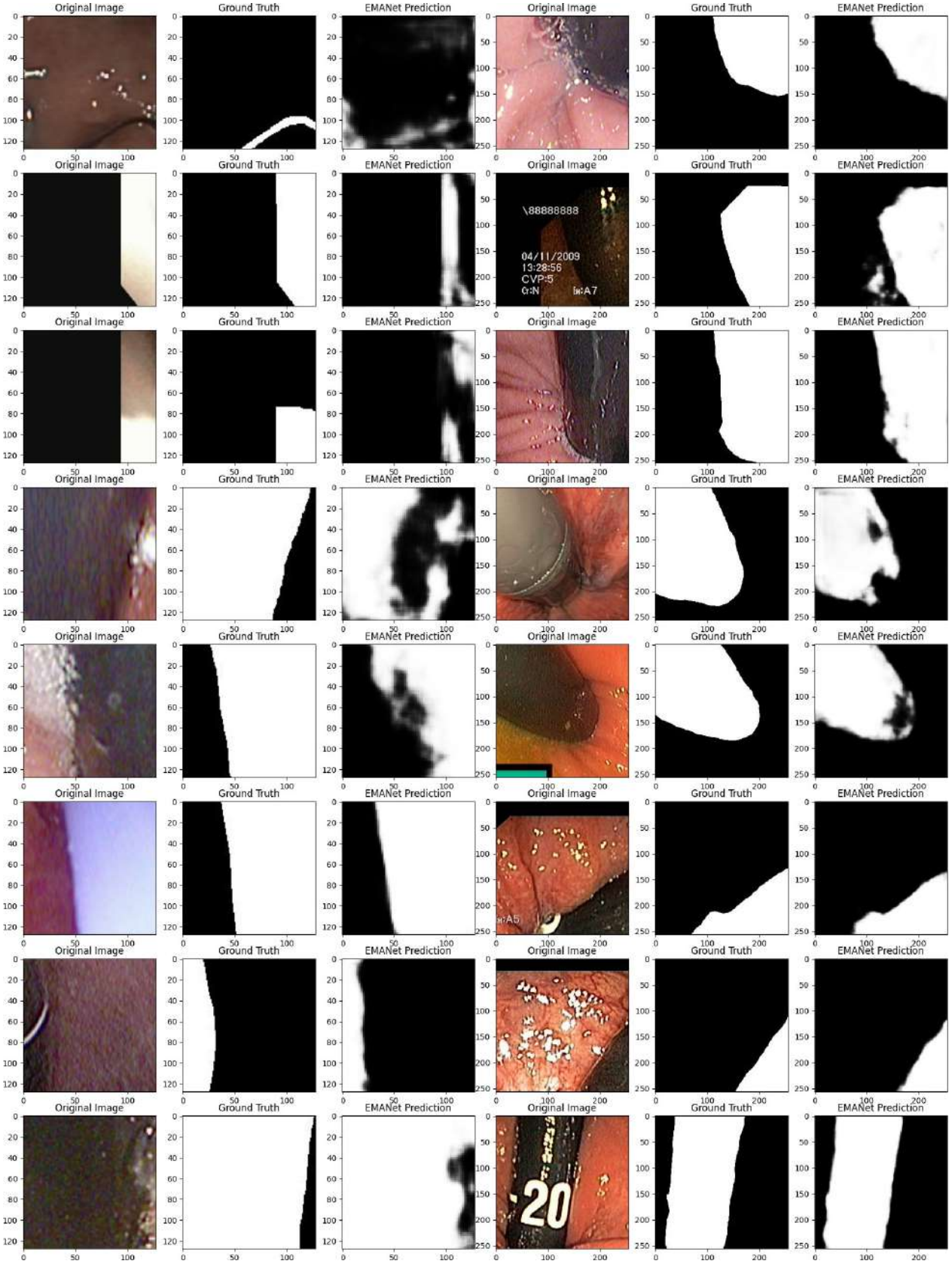

Figure 6.4.39: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Kvasir-Instrument dataset

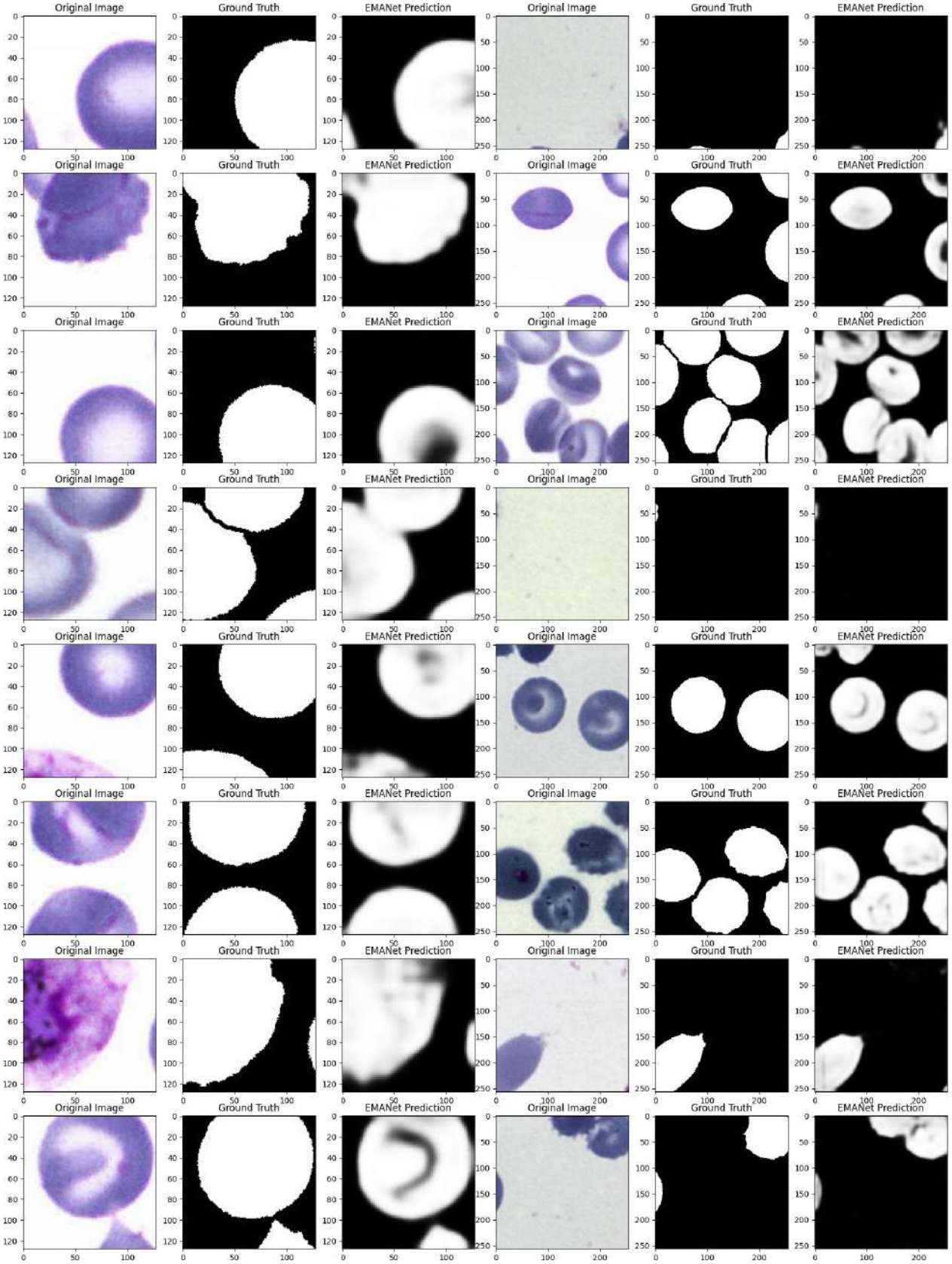

Figure 6.4.40: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Cell dataset

6.4.11 Qualitative Results of AuNet

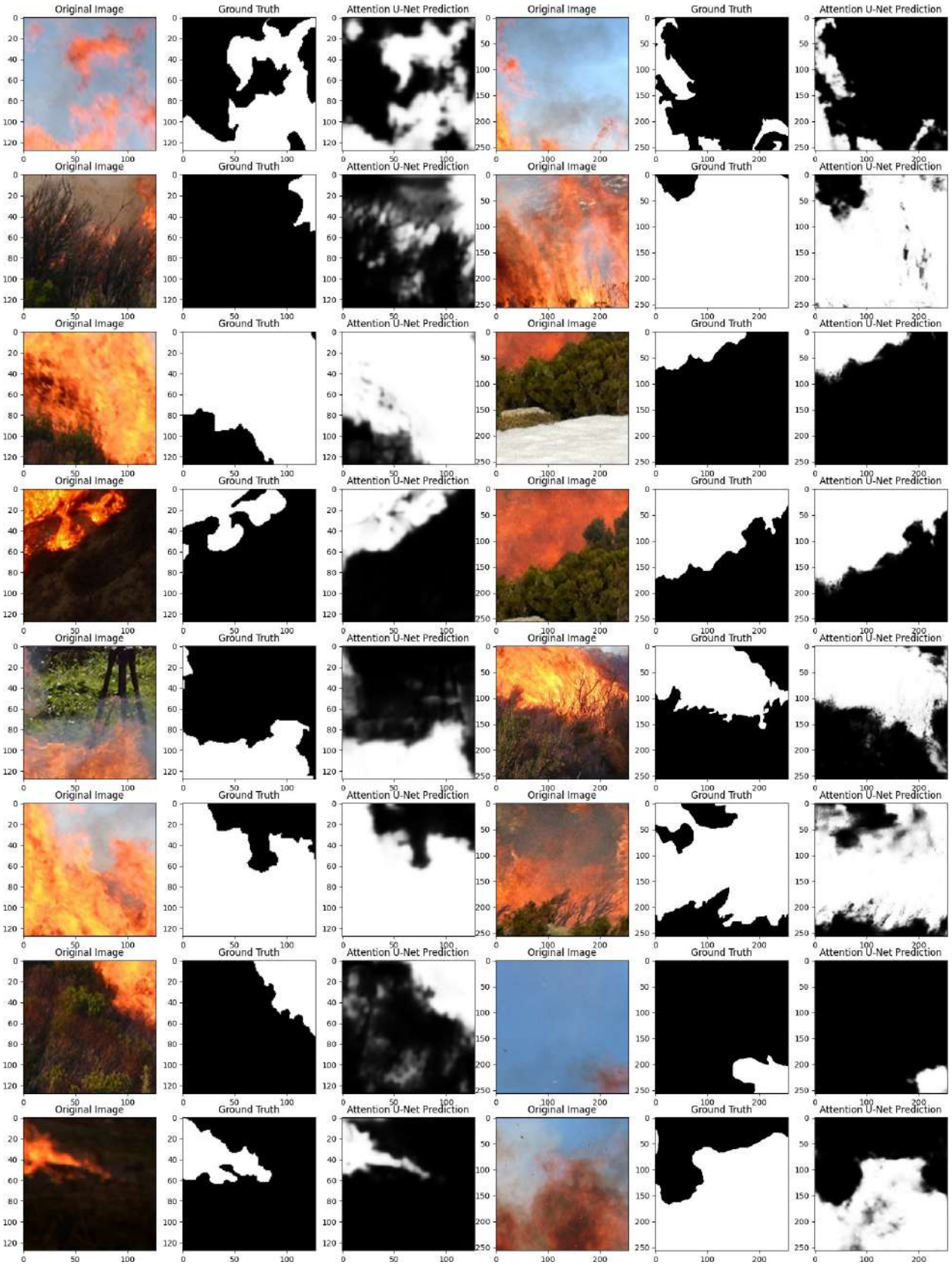

Figure 6.4.41: The original, mask and prediction for 128×128 and 256×256 image sizes respectively on the Corsican Fire dataset

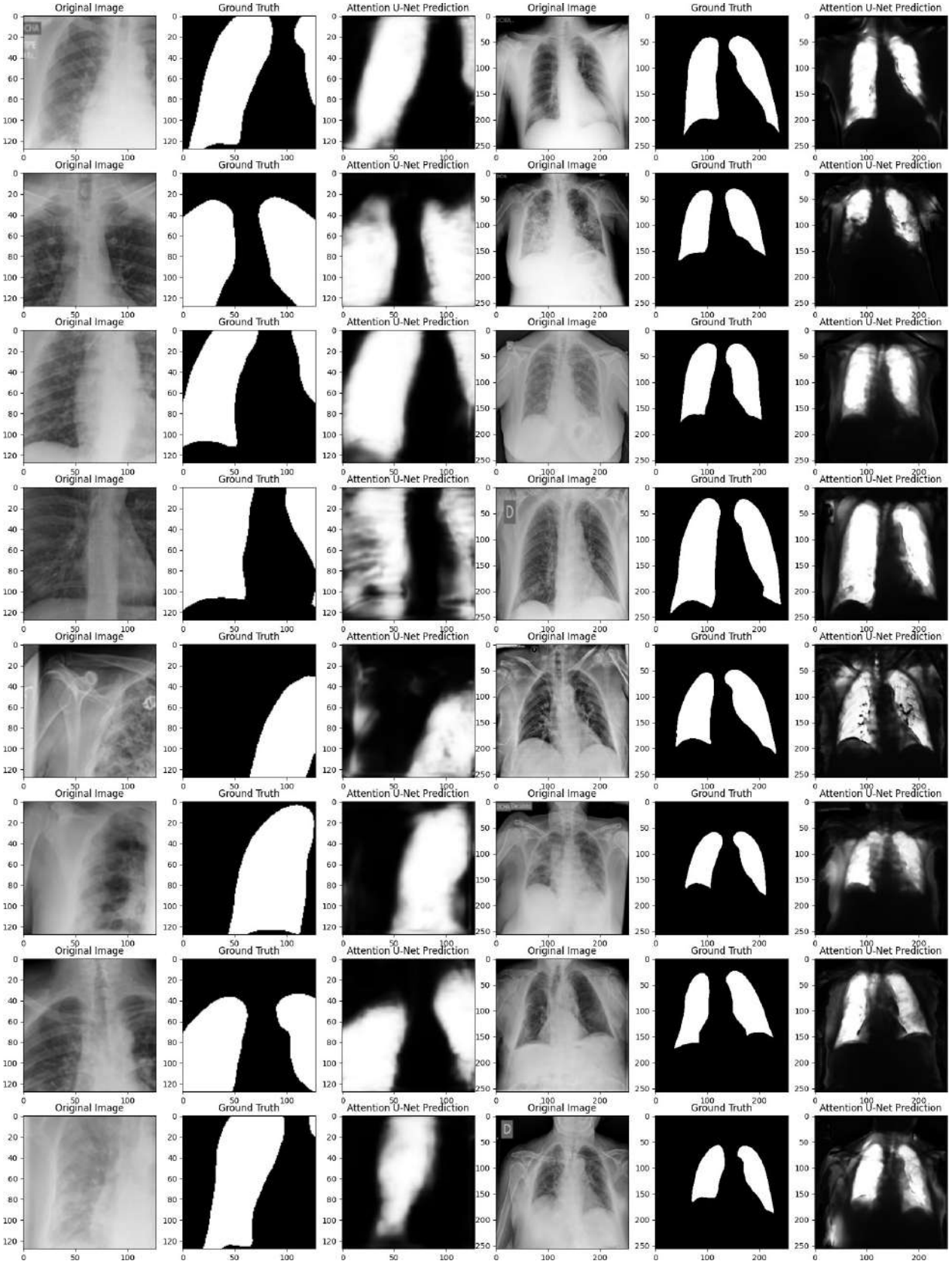

Figure 6.4.42: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively on the COVID-QU-Ex dataset

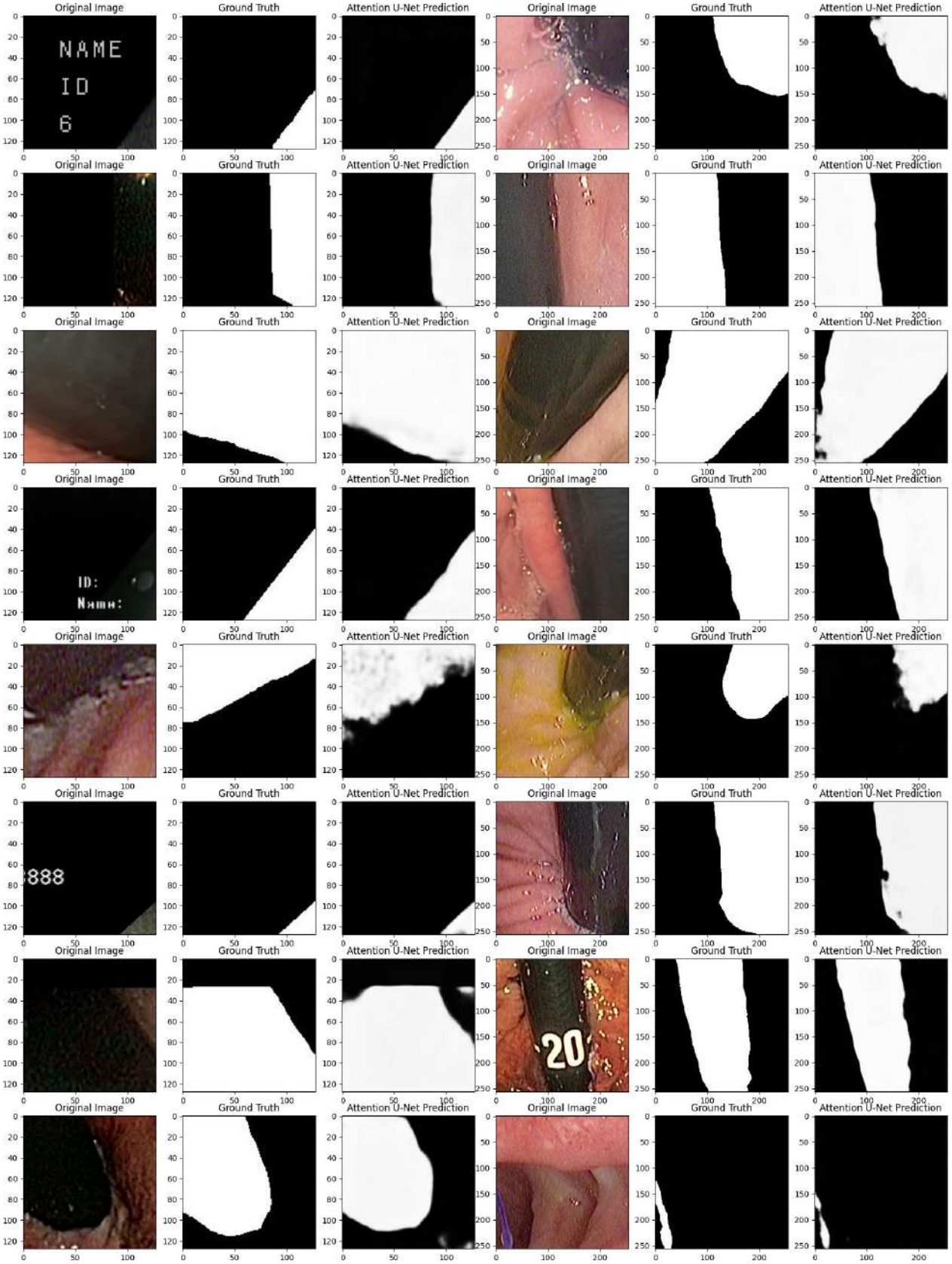

Figure 6.4.43: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Kvasir-Instrument dataset

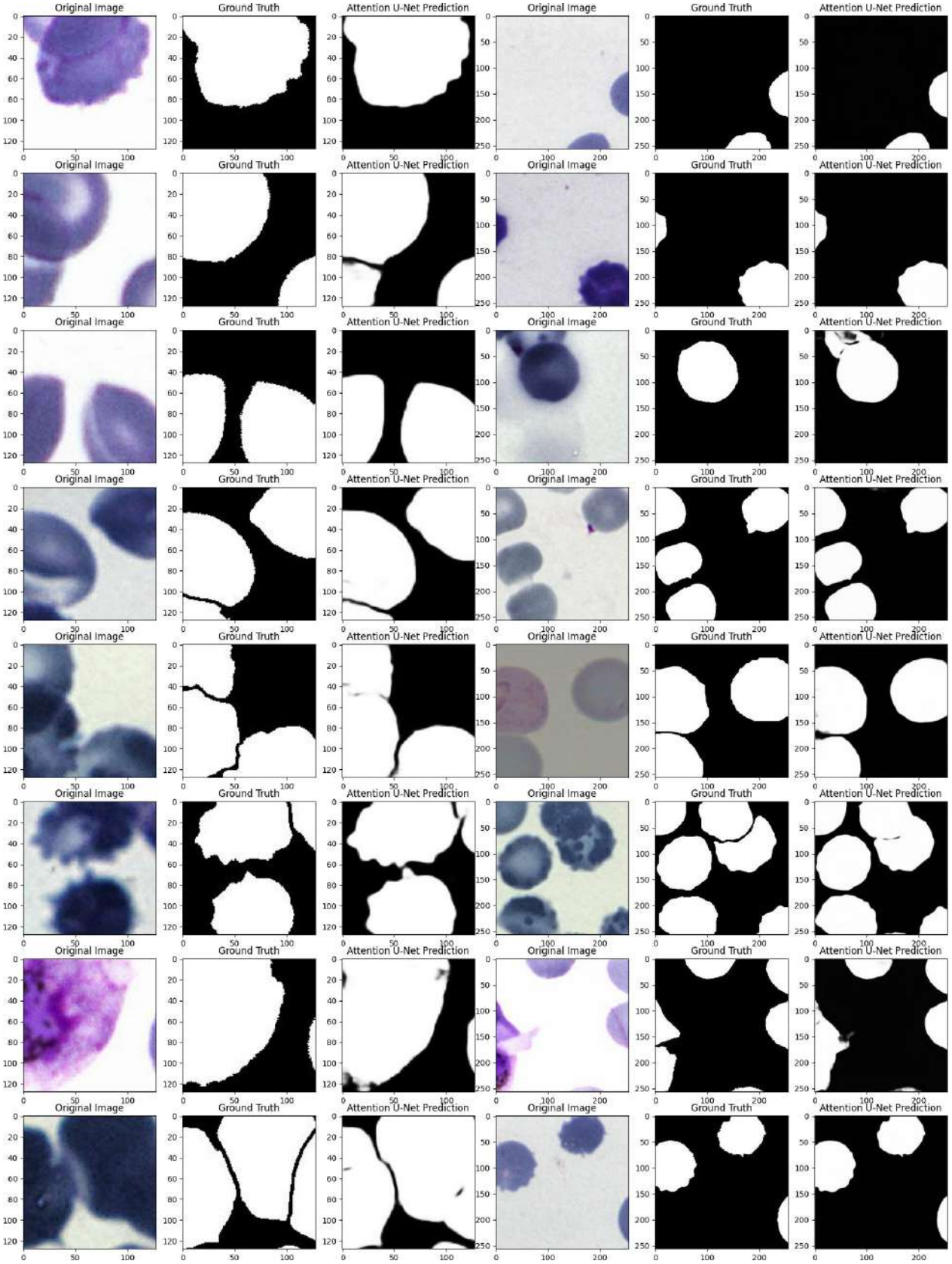

Figure 6.4.44: The original, mask, and prediction for 128×128 and 256×256 image sizes respectively in the Cell dataset

6.4.12 Commenting on Qualitative Results

Corsican Fire dataset

After training the models for 200 epochs on the Corsican Fire dataset, several of them stand out for achieving remarkable results. DeepLabv3+, HRNet, Swin Transformer, BASNet, UNet 3+ and AuNet models showed excellent performance for both $128x128$ and $256x256$ resolution. Surprisingly, they even outperform ground truth masks in terms of granularity, shape accuracy, and recording of brightness changes in fire areas. This highlights their ability to learn complex features and nuances from data.

EmaNet and DDRNet also show strong capabilities, providing accurate mask predictions. While the metrics may not rank them at the top of the table, their practical performance is remarkable. On the other hand, BiSeNet and HRNet + OCR show decent mask predictions at 256x256 resolution, but struggle at the lower $128x128$ resolution. The latest models are able to capture the general shape, but lack finer details in their projections of the mask.

Interestingly, PIDNet, despite its lower metric results, excels in mask prediction for both $128x128$ and $256x256$ resolutions. This shows that metrics alone may not fully capture a model's performance, and that there may be details that metrics overlook.

Dataset COVID-QU-Ex

Using transfer learning, the DeepLabv3+, HRNet, Swin Transformer, BASNet, UNet 3+, AuNet, EMAnet, and DDRNet models were fine-tuned on the COVID-QU-Ex dataset for lung segmentation. Remarkably, even with only 30 epochs, these models consistently demonstrated excellent performance at both 128x128 and 256x256 resolutions. In particular, they outperformed the provided masks, capturing complex lung structures, and presenting detailed predictions that even exceeded the quality of the original image. The Swin Transformer model again stood out with excellent predictions, highlighting its effectiveness in lung segmentation tasks. In contrast, the BiSeNet and HRNet + OCR models provided satisfactory results for both analyses, although they adhered more closely to the original ground truth masks without introducing more detail. These observations suggest that some models possess the ability to detect subtle patterns and structures, thereby potentially revealing insights beyond manual labeling. In essence, the transfer learning approach highlighted the adaptability of these models to diverse datasets, offering powerful segmentation capabilities and revealing finer anatomical nuances.

Kvasir-Instrument Dataset

The transfer learning strategy was then extended to a challenging dataset called Kvasir-Instrument, known for its complex medical instrument segmentation challenge. This dataset includes a wide range of instrument shapes and sizes, making it a particularly challenging task due to the complex variations inherent in medical instruments. The dataset includes medical instruments with a variety of characteristics, including shape, size, and appearance. This diversity reflects the real-world complexities encountered in medical scenarios, where tools can vary significantly. Given the need for accurate organ segmentation, this dataset presents a severe test for segmentation models due to the varied boundaries, overlapping structures, and complexities present in medical instruments.

Continuing the evaluation, the Aunet, BASNet, PIDNet, Swin Transformer and Unet3+ models showed remarkable performance in both 128x128 and 256x256 resolution. These models consistently provided excellent results, highlighting their robustness in segmenting difficult medical instrument images.

Among the other models, DeepLabv3+, HRNet, EMANet and DDRNet achieved excellent performance at 256x256 resolution, offering perfect mask predictions. However, their performance at 128x128, while still good, fell short of the excellence they showed at 256x256. This difference could arise from the increased difficulty of accurately capturing intricate details at the lower resolution.

In contrast, the Bisnet and HRNet + OCR models experienced significant difficulties on this dataset. While they succeeded in identifying the instrument shapes, their struggle to accurately locate the masks highlights the challenges posed by the varied and complex instrument shapes in the Kvasir-Instrument dataset.

Cell Dataset

Then, the transfer learning approach was extended to the cell dataset, which is specifically designed for cell segmentation, with an emphasis on circular shapes. Impressively, even with only 30 training epochs, all models, including AuNet, BASNet, PIDNet, Swin Transformer, UNet3+, DeepLabv3+, HRNet, EMANet, DDRNet, BiSeNet, HRNet + OCR, performed exceptionally well on this set data.

The cell dataset has been curated to address cell segmentation tasks, particularly targeting circular cell shapes. However, seemingly simple circular cell structures present unique challenges due to variations in size and shape that can affect accurate segmentation.

In general, these models demonstrated excellent performance, accurately capturing circular cellular structures. Importantly, their ability was consistent at both $128x128$ and $256x256$ resolutions, further highlighting their adaptability to different image sizes and segmentation complexities.

Chapter 7

Conclusions and Future Work

7.1 Conclusions

The exemplary performance achieved by models such as AuNet, EmaNet, Swin Transformer, and DeepLabv3+ on most image segmentation tasks can be attributed to a common base of core architectures. In particular, all these models exploit attentional mechanisms to enhance their segmentation capabilities. While Attention U-Net emphasizes attention during decoding, EmaNet uses an efficient point-wise gated attention, and Swin Transformer leverages self-attention based on different patches to capture both local and global context information. In addition, DeepLabv3+ incorporates Atrous Spatial Pyramid Pooling (ASPP), a specialized attention mechanism that captures multi-scale context information. These models collectively prioritize multiscale analysis through different techniques. Attention U-Net and EmaNet models via feature fusion, Swin Transformer via hierarchical self-attention and DeepLabv3+ using ASPP. This common thread of attention mechanisms and multiscale analysis enables AuNet, EmaNet, Swin Transformer, and DeepLabv3+ to excel in image segmentation tasks, highlighting the importance of such architectural choices to achieve excellent segmentation performance.

The peculiar performance pattern observed with the PIDNet model, where it performed excellently in lung segmentation during transfer learning but performed relatively poorly overall, can be attributed to several factors. The PIDNet architecture may not be optimally suited to capture the complex, irregular, and varied shapes that characterize flames. This mismatch could lead to less efficient feature extraction and segmentation in this context. The success of the PIDNet model in lung segmentation, especially with elliptical shapes such as lungs, can be attributed to the inherent advantages of its architecture in capturing specific features and patterns that align well with these shapes. One of the key components of PIDNet that probably contributes to its effectiveness in lung segmentation is the Pyramid Dilated Convolution (PDC) module. This module uses dilated convolutions with different dilation rates to capture multiscale information from the input images. In the context of lung segmentation, where elliptical shapes dominate, the ability of the PDC module to capture both local and global features is particularly important. The expansion rates allow the model to integrate the frame at various scales, which is suitable for detecting the edges and contours of lung structures.

In addition, PIDNet's attention mechanism, which enhances the model's focus on informative regions, could also contribute to its success in lung segmentation. Lungs often have distinct and well-defined boundaries, and the attention mechanism could help the model emphasize these critical regions, improving segmentation accuracy. Furthermore, the combination of skip connection and dense connection architecture allows for efficient feature propagation and reuse, which is advantageous for capturing the consistent and structured features of lung shapes.

However, it is important to note that while PIDNet excels in lung segmentation due to these architectural features, its performance may not generalize as well to other types of images with different shapes and structures, as observed in its performance on the fire database.

In this extensive study, we engaged in training deep learning models using a variety of datasets and tasks. We started by training our models on the Coriscan Fire database, which contains detailed images of fires with

various shapes and angles, from irregular to sharp and more. This initial training was crucial as it provided our models with a solid understanding of the complex and varied visual patterns present in flames. As we progressed to different datasets and tasks, such as lung segmentation in COVID-QU-EX, medical instrument segmentation in Kvasir-Instrument, and cell segmentation, the importance of our initial training on fire datasets became even more apparent. The insights gained from the fire dataset proved to be remarkably adaptable to various scenarios.

For example, familiarizing the models with rough flame shapes made it much easier for the models to compete with elliptical shapes, and thus training them on this dataset was instrumental in effectively segmenting the lungs on the COVID-QU-EX dataset. Similarly, the ability of the models to handle elongated and irregular shapes, learned from fire images, contributed to the successful segmentation of medical instruments in the Kvasir-Instrument dataset. Even in the case of cell segmentation, where circular structures predominate, exposing the models to various fire patterns enhanced their ability to accurately distinguish and segment cells.

In essence, our study highlights that training with the Coriscan Fire database provided our models with a strong foundation for recognizing and understanding a variety of visual patterns. This fundamental knowledge, gained from fire images with their complex shapes, proved invaluable in enabling our models to adapt and excel in different segmentation tasks involving a variety of shapes and structures.

7.2 Future Works

In order to extend and improve the current study on the evaluation of image segmentation models on various benchmark datasets, there are several areas that could be explored.

First, it would be beneficial to train the models on even more datasets with different characteristics to study the performance of transfer learning models with different training weights to demonstrate the generalizability of the models.

Second, investigating the use of pre-trained models for image segmentation and evaluating their performance on benchmark datasets could provide insights into how to improve the models. Pre-training models on large-scale datasets has been shown to improve model performance on various computer vision tasks.

Third, investigating the impact of various data augmentation techniques on the performance of segmentation models would provide insights into how to further improve the models. Data augmentation techniques such as rotation, scaling and inversion have been shown to improve the performance of models.

Fourth, it would be valuable to investigate the impact of different loss functions on the performance of segmentation models. While the binary cross-entropy loss function was used in the present study, other variations or combinations of loss functions should be explored to improve segmentation performance.

Fifth, conducting a study of the effect of different hyperparameters on the performance of segmentation models would provide insight into how to optimize the models. The performance of deep learning models is sensitive to the choice of hyperparameters such as learning rate, batch size, and optimizer.

Sixth, a differentiated evaluation metric, which would delve into parameters beyond the simple quantitative dimension, could include factors such as inference time, enriching the performance landscape.

Finally, studying the integration of segmentation models with other computer vision tasks, such as object detection and recognition, would be a valuable area of research. Image segmentation is often used as a preprocessing step for these tasks, and understanding the effectiveness of the models in real-world applications would be beneficial.

By exploring these areas, the present study could be extended to provide a more comprehensive understanding of image segmentation models and their performance on different datasets. These areas would also provide information on how to further optimize and improve the models for real-world applications.

Bibliography

[04] "A snake model for object tracking in natural sequences". In: *Signal processing: image communication* 19.3 (2004), pp. 219–238.

[Ala+23] Alam, T. et al. "Improving Breast Cancer Detection and Diagnosis through Semantic Segmentation Using the Unet3+ Deep Learning Framework". In: *Biomedicines* 11.6 (2023). ISSN: 2227-9059. DOI: 10.3390/biomedicines11061536. URL:

[Alh+19] Alhnaity, B. et al. "Using deep learning to predict plant growth and yield in greenhouse environments". In: *International Symposium on Advanced Technologies and Management for Innovative Greenhouses: GreenSys2019 1296*. 2019, pp. 425–432.

[Alh+21] Alhnaity, B. et al. "An autoencoder wavelet based deep neural network with attention mechanism for multi-step prediction of plant growth". In: *Information Sciences* 560 (2021), pp. 35–50.

[AZ18] Ali, N. and Zafar, B. "MSRC-v2 image dataset". In: (Aug. 2018). DOI: 10.6084/m9.figshare.6075788.v2. URL:

[AKK22] Arsenos, A., Kollias, D., and Kollias, S. "A Large Imaging Database and Novel Deep Neural Architecture for Covid-19 Diagnosis". In: *2022 IEEE 14th Image, Video, and Multidimensional Signal Processing Workshop (IVMSP)*. IEEE. 2022, pp. 1–5.

[Ars+23] Arsenos, A. et al. "Data-Driven Covid-19 Detection Through Medical Imaging". In: *2023 IEEE International Conference on Acoustics, Speech, and Signal Processing Workshops (ICASSPW)*. 2023, pp. 1–5. DOI: 10.1109/ICASSPW59220.2023.10193437.

[AXK01] Avrithis, Y., Xirouhakis, Y., and Kollias, S. "Affine-invariant curve normalization for object shape representation, classification, and retrieval". In: *Machine Vision and Applications* 13 (2001), pp. 80–94.

[BKC15] Badrinarayanan, V., Kendall, A., and Cipolla, R. *SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation*. 2015. DOI: 10.48550/ARXIV.1511.00561. URL:

[BG94] Bebis, G. and Georgiopoulos, M. "Feed-forward neural networks". In: *IEEE Potentials* 13.4 (1994), pp. 27–31. DOI: 10.1109/45.329294.

[BP21] Bird, G. and Polivoda, M. E. *Backpropagation Through Time For Networks With Long-Term Dependencies*. 2021. arXiv: 2103.15589 [cs.LG].

[Cal+18] Caliva, F. et al. "A deep learning approach to anomaly detection in nuclear reactors". In: *2018 International joint conference on neural networks (IJCNN)*. IEEE. 2018, pp. 1–8.

[Che+21] Chen, P.-C. et al. *A Simple and Effective Positional Encoding for Transformers*. 2021. arXiv: 2104.08698 [cs.CL].

[Che+20] Chen, J. et al. "Design, Optimization, and Study of Small Molecules That Target Tau Pre-mRNA and Affect Splicing". In: *Journal of the American Chemical Society* 142 (May 2020). DOI: 10.1021/jacs.0c00768.

[Che+16] Chen, L.-C. et al. *DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs*. 2016. DOI: 10.48550/ARXIV.1606.00915. URL:

[Che+17] Chen, L.-C. et al. *DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs*. 2017. arXiv: 1606.00915 [cs.CV].

[Che+18] Chen, L.-C. et al. *Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation*. 2018. DOI: 10.48550/ARXIV.1802.02611. URL:

[DLK20] De Sousa Ribeiro, F., Leontidis, G., and Kollias, S. "Introducing routing uncertainty in capsule networks". In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 6490–6502.

[De +20] De Sousa Ribeiro, F. et al. "Deep bayesian self-training". In: *Neural Computing and Applications* 32.9 (2020), pp. 4275–4291.

[Den+09] Deng, J. et al. "ImageNet: A large-scale hierarchical image database". In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009. DOI: 10.1109/CVPR.2009.5206848.

[Dos+20] Dosovitskiy, A. et al. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2020. DOI: 10.48550/ARXIV.2010.11929. URL:

[DSC21] Dubey, S. R., Singh, S. K., and Chaudhuri, B. B. *Activation Functions in Deep Learning: A Comprehensive Survey and Benchmark*. 2021. DOI: 10.48550/ARXIV.2109.14545. URL:

[HB20] Hafiz, A. M. and Bhat, G. M. "A survey on instance segmentation: state of the art". In: *International Journal of Multimedia Information Retrieval* 9.3 (July 2020), pp. 171–189. DOI: 10.1007/s13735-020-00195-x. URL:

[HA19] Haldorai, A. and Arulmurugan, R. "Supervised, Unsupervised and Reinforcement Learning -A Detailed Perspective". In: *Journal of Advanced Research in Dynamical and Control Systems* 11 (Jan. 2019), pp. 429–433.

[He+15] He, K. et al. *Deep Residual Learning for Image Recognition*. 2015. DOI: 10.48550/ARXIV.1512.03385. URL:

[Hon+21] Hong, Y. et al. *Deep Dual-resolution Networks for Real-time and Accurate Semantic Segmentation of Road Scenes*. 2021. DOI: 10.48550/ARXIV.2101.06085. URL:

[HTY18] Hua, B.-S., Tran, M.-K., and Yeung, S.-K. *Pointwise Convolutional Neural Networks*. 2018. arXiv: 1712.05245 [cs.CV].

[Hua+20] Huang, H. et al. *UNet 3+: A Full-Scale Connected UNet for Medical Image Segmentation*. 2020. DOI: 10.48550/ARXIV.2004.08790. URL:

[99] "Interactive content-based retrieval in video databases using fuzzy classification and relevance feedback". In: *Proceedings IEEE International Conference on Multimedia Computing and Systems*. Vol. 2. IEEE. 1999, pp. 954–958.

[Jha+20] Jha, D. et al. *Kvasir-Instrument: Diagnostic and therapeutic tool segmentation dataset in gastrointestinal endoscopy*. 2020. arXiv: 2011.08065 [physics.med-ph].

[Kir+18] Kirillov, A. et al. *Panoptic Segmentation*. 2018. DOI: 10.48550/ARXIV.1801.00868. URL:

[KSK19] Kollia, I., Stafylopatis, A.-G., and Kollias, S. "Predicting Parkinson's disease using latent information extracted from deep neural networks". In: *2019 International Joint Conference on Neural Networks (IJCNN)*. IEEE. 2019, pp. 1–8.

[Kol+20a] Kollias, D. et al. "Deep Transparent Prediction through Latent Representation Analysis". In: *arXiv preprint arXiv:2009.07044* (2020).

[KAK23a] Kollias, D., Arsenos, A., and Kollias, S. "A deep neural architecture for harmonizing 3-D input data analysis and decision making in medical imaging". In: *Neurocomputing* 542 (2023), p. 126244.

[KAK23b] Kollias, D., Arsenos, A., and Kollias, S. "AI-MIA: Covid-19 detection and severity analysis through medical imaging". In: *Computer Vision–ECCV 2022 Workshops: Tel Aviv, Israel, October 23–27, 2022, Proceedings, Part VII*. Springer. 2023, pp. 677–690.

[Kol+18] Kollias, D. et al. "Deep neural architectures for prediction in healthcare". In: *Complex & Intelligent Systems* 4.2 (2018), pp. 119–131.

[Kol+21] Kollias, D. et al. "Mia-cov19d: Covid-19 detection through 3-d chest ct image analysis". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision*. 2021, pp. 537–544.

[Kol+23] Kollias, D. et al. "Abaw: Valence-arousal estimation, expression recognition, action unit detection & emotional reaction intensity estimation challenges". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2023, pp. 5888–5897.

[Kol+] Kollias, D. et al. "Adaptation and contextualization of deep neural network models". In: *2017 IEEE symposium series on computational intelligence (SSCI)*. IEEE, pp. 1–8.

[Kol+20b] Kollias, D. et al. "Transparent adaptation in deep medical image diagnosis". In: *International Workshop on the Foundations of Trustworthy AI Integrating Learning, Optimization and Reasoning*. Springer. 2020, pp. 251–267.

[Kol+22] Kollias, S. et al. "Machine learning for analysis of real nuclear plant data in the frequency domain". In: *Annals of Nuclear Energy* 177 (2022), p. 109293.

[Lan+18] Lanaras, C. et al. "Super-resolution of Sentinel-2 images: Learning a globally applicable deep neural network". In: *ISPRS Journal of Photogrammetry and Remote Sensing* (Dec. 2018), pp. 305–319. ISSN: 0924-2716. DOI: 10.1016/j.isprsjprs.2018.09.018.

[Lee+20] Lee, K.-S. et al. "Evaluation of Scalability and Degree of Fine-Tuning of Deep Convolutional Neural Networks for COVID-19 Screening on Chest X-ray Images Using Explainable Deep-Learning Algorithm". In: *Journal of Personalized Medicine* 10.4 (2020). ISSN: 2075-4426. DOI: 10.3390/jpm10040213. URL:

[Li+22] Li, Q. et al. *Dite-HRNet: Dynamic Lightweight High-Resolution Network for Human Pose Estimation*. 2022. arXiv: 2204.10762 [cs.CV].

[Li+19] Li, X. et al. *Expectation-Maximization Attention Networks for Semantic Segmentation*. 2019. DOI: 10.48550/ARXIV.1907.13426. URL:

[Lin+15] Lin, T.-Y. et al. *Microsoft COCO: Common Objects in Context*. 2015. arXiv: 1405.0312 [cs.CV].

[LDY19] Liu, X., Deng, Z., and Yang, Y. "Recent progress in semantic image segmentation". In: *Artificial Intelligence Review* 52 (Aug. 2019), pp. 1–18. DOI: 10.1007/s10462-018-9641-3.

[Liu+21] Liu, Z. et al. *Swin Transformer: Hierarchical Vision Transformer using Shifted Windows*. 2021. DOI: 10.48550/ARXIV.2103.14030. URL:

[LSD14] Long, J., Shelhamer, E., and Darrell, T. *Fully Convolutional Networks for Semantic Segmentation*. 2014. DOI: 10.48550/ARXIV.1411.4038. URL:

[LTK23] Lymperopoulos, E., Tzouveli, P., and Kollias, S. "Satellite image super-resolution for forest localization". In: *2023 International Conference on Machine Intelligence for GeoAnalytics and Remote Sensing (MIGARS)*. Vol. 1. 2023, pp. 1–4. DOI: 10.1109/MIGARS57353.2023.10064552.

[Maj+22] Majid, S. et al. "Attention based CNN model for fire detection and localization in real-world images". In: *Expert Systems with Applications* 189 (2022), p. 116114. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2021.116114. URL:

[Mar+15] Martín Abadi et al. *TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems*. Software available from tensorflow.org. 2015. URL:

[Nag+11] Nagi, J. et al. "Max-pooling convolutional neural networks for vision-based hand gesture recognition". In: *2011 IEEE International Conference on Signal and Image Processing Applications (ICSIPA)*. 2011. DOI: 10.1109/ICSIPA.2011.6144164.

[NG18] Nemalidinne, S. M. and Gupta, D. "Nonsubsampled contourlet domain visible and infrared image fusion framework for fire detection using pulse coupled neural network and spatial fuzzy clustering". In: *Fire Safety Journal* 101 (2018), pp. 84–101. ISSN: 0379-7112. DOI: https://doi.org/10.1016/j.firesaf.2018.08.012. URL:

[Nos21] Nosouhian S.; Nosouhian, F. K. K. *A Review of Recurrent Neural Network Architecture for Sequence Learning: Comparison between LSTM and GRU*. 2021. DOI: https://doi.org/10.20944/preprints202107.0252.v1. URL:

[ON15] O'Shea, K. and Nash, R. *An Introduction to Convolutional Neural Networks*. 2015. DOI: 10.48550/ARXIV.1511.08458. URL:

[Okt+18] Oktay, O. et al. *Attention U-Net: Learning Where to Look for the Pancreas*. 2018. DOI: 10.48550/ARXIV.1804.03999. URL:

[96] "Optimal filter banks for signal reconstruction from noisy subband components". In: *IEEE transactions on signal processing* 44.2 (1996), pp. 212–224.

[Ots79] Otsu, N. "A Threshold Selection Method from Gray-Level Histograms". In: *IEEE Transactions on Systems, Man, and Cybernetics* 9.1 (1979), pp. 62–66. DOI: 10.1109/TSMC.1979.4310076.

[Pas+19] Paszke, A. et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019. URL:

[PK22] Psaroudakis, A. and Kollias, D. "MixAugment & Mixup: Augmentation Methods for Facial Expression Recognition". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2022, pp. 2367–2375.

[Qin+19] Qin, X. et al. "BASNet: Boundary-Aware Salient Object Detection". In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019. DOI: 10.1109/CVPR.2019.00766.

[RLK20] Ribeiro, F. D. S., Leontidis, G., and Kollias, S. "Capsule routing via variational bayes". In: *Proceedings of the AAAI Conference on Artificial Intelligence*. Vol. 34. 04. 2020, pp. 3749–3756.

[RFB15] Ronneberger, O., Fischer, P., and Brox, T. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. DOI: 10.48550/ARXIV.1505.04597. URL:

[SSW88] Sahoo, P., Soltani, S., and Wong, A. "A Survey of Thresholding Techniques". In: *Computer Vision, Graphics, and Image Processing* 41 (Feb. 1988), pp. 233–260. DOI: 10.1016/0734-189X(88)90022-9.

[STK23] Salpea, N., Tzouveli, P., and Kollias, D. "Medical Image Segmentation: A Review of Modern Architectures". In: Feb. 2023, pp. 691–708. ISBN: 978-3-031-25081-1. DOI: 10.1007/978-3-031-25082-8_47.

[Sch15] Schmidhuber, J. "Deep learning in neural networks: An overview". In: *Neural Networks* 61 (Jan. 2015), pp. 85–117. DOI: 10.1016/j.neunet.2014.09.003. URL:

[Sch+19] Schoenauer-Sebag, A. et al. *Multi-Domain Adversarial Learning*. 2019. arXiv: 1903.09239 [stat.ML].

[Sim18] Simeone, O. "A Very Brief Introduction to Machine Learning With Applications to Communication Systems". In: *IEEE Transactions on Cognitive Communications and Networking* 4.4 (2018), pp. 648–664. DOI: 10.1109/TCCN.2018.2881442.

[SVL14] Sutskever, I., Vinyals, O., and Le, Q. V. *Sequence to Sequence Learning with Neural Networks*. 2014. DOI: 10.48550/ARXIV.1409.3215. URL:

[TKS17] Tagaris, A., Kollias, D., and Stafylopatis, A. "Assessment of Parkinson's disease based on deep neural networks". In: *International Conference on Engineering Applications of Neural Networks*. Springer. 2017, pp. 391–403.

[Tag+18] Tagaris, A. et al. "Machine learning for neurodegenerative disorder diagnosis—survey of practices and launch of benchmark dataset". In: *International Journal on Artificial Intelligence Tools* 27.03 (2018), p. 1850011.

[Tah+21] Tahir, A. M. et al. "COVID-19 infection localization and severity grading from chest X-ray images". In: *Computers in Biology and Medicine* 139 (2021), p. 105002. ISSN: 0010-4825. DOI: https://doi.org/10.1016/j.compbiomed.2021.105002. URL:

[Tho16] Thoma, M. *A Survey of Semantic Segmentation*. 2016. DOI: 10.48550/ARXIV.1602.06541. URL:

[Tou+17] Toulouse, T. et al. "Computer vision for wildfire research: An evolving image dataset for processing and analysis". In: *Fire Safety Journal* 92 (2017), pp. 188–194. ISSN: 0379-7112. DOI: https://doi.org/10.1016/j.firesaf.2017.06.012. URL:

[TV07a] Triggs, B. and Verbeek, J. "Scene Segmentation with CRFs Learned from Partially Labeled Images". In: *Advances in Neural Information Processing Systems*. Ed. by J. Platt et al. Vol. 20. Curran Associates, Inc., 2007. URL:

[TV07b] Triggs, B. and Verbeek, J. "Scene Segmentation with CRFs Learned from Partially Labeled Images". In: *Advances in Neural Information Processing Systems*. Ed. by J. Platt et al. Vol. 20. Curran Associates, Inc., 2007. URL:

[Vas+17] Vaswani, A. et al. *Attention Is All You Need*. 2017. DOI: 10.48550/ARXIV.1706.03762. URL:

[Wal+03] Wallace, M. et al. "Intelligent one-stop-shop travel recommendations using an adaptive neural network and clustering of history". In: *Information Technology & Tourism* 6.3 (2003), pp. 181–193.

[Wan+19] Wang, J. et al. *Deep High-Resolution Representation Learning for Visual Recognition*. 2019. DOI: 10.48550/ARXIV.1908.07919. URL:

[Win+20] Wingate, J. et al. "Unified deep learning approach for prediction of Parkinson's disease". In: *IET Image Processing* 14.10 (2020), pp. 1980–1989.

[XXB22] Xu, J., Xiong, Z., and Bhattacharyya, S. P. *PIDNet: A Real-time Semantic Segmentation Network Inspired from PID Controller*. 2022. DOI: 10.48550/ARXIV.2206.02066. URL:

[Yu+18] Yu, C. et al. *BiSeNet: Bilateral Segmentation Network for Real-time Semantic Segmentation*. 2018. DOI: 10.48550/ARXIV.1808.00897. URL:

[YK15] Yu, F. and Koltun, V. *Multi-Scale Context Aggregation by Dilated Convolutions*. 2015. DOI: 10.48550/ARXIV.1511.07122. URL:

[Yua+19] Yuan, Y. et al. "Segmentation Transformer: Object-Contextual Representations for Semantic Segmentation". In: (2019). DOI: 10.48550/ARXIV.1909.11065. URL:

[Yua+20] Yuan, Y. et al. *SegFix: Model-Agnostic Boundary Refinement for Segmentation*. 2020. DOI: 10.48550/ARXIV.2007.04269. URL:

[Zhu+20] Zhuang, F. et al. *A Comprehensive Survey on Transfer Learning*. 2020. arXiv: 1911.02685 [cs.LG].