Karlsruher Institut für Technologie

HOCHSCHULE
RAVENSBURG-WEINGARTEN
UNIVERSITY
OF APPLIED SCIENCES

Unified Loss Merging Framework for Enhanced Semantic Segmentation

Master Thesis

Author:	Martin Samuel Lanz Matr.-No.: 33808 martin.lanz@rwu.de
First Supervisor:	Prof. Dr. rer. nat. Stefan Elser stefan.elser@rwu.de
Second Supervisor:	apl. Prof. Dr. Markus Reischl markus.reischl@kit.edu

submitted on:

July 27, 2023

SS 23

Abstract

Semantic segmentation is an important task in computer vision, aiming to assign a class label to every pixel in an image, enabling applications such as autonomous driving, medical image analysis, or facial recognition. Deep learning has significantly improved semantic segmentation performance in recent years. However, optimizing loss functions remains an open challenge due to critical limitations if used individually. This project investigates six popular loss functions and introduces a merging framework to form new combined losses, addressing these limitations and improving segmentation performance.

The project begins by providing a comprehensive overview of the fundamentals of Machine Learning (ML) and semantic segmentation, including terminology, objectives, metrics, and architectures. The literature review covers a broad range of techniques for generally improving semantic segmentation. Subsequently, the limiting factors of six loss functions are discussed, and a methodology is proposed to merge multiple losses into a single final loss, which aims to address the shortcomings of models trained with standard single losses.

A U-Net-based segmentation framework is presented to validate the approach, incorporating all theoretically described methods in code. Several experiments on multiple datasets are conducted to compare the performance of the proposed methods against baseline models trained with single losses. Quantitative and qualitative results are presented, along with an ablation study to evaluate further the impact of the presented loss merging strategies.

This unified approach demonstrates that combining multiple loss functions can significantly improve semantic segmentation performance for a whole set of loss combinations across multiple datasets. The project aims to contribute to advancing semantic segmentation research and provides a foundation for future investigations into more effective loss function design and optimization.

Acknowledgements

I want to express my sincere gratitude to several individuals who have been instrumental in the successful completion of my Master's thesis.

Firstly, my appreciation goes to Doctoral Researcher Luca Rettenberger, my principal supervisor. His guidance and regular feedback during our weekly meetings were essential to the development and direction of this thesis. His commitment to my academic progress is highly regarded.

I am also thankful to Prof. Dr. rer. nat. Stefan Elser, my professor, and first reviewer, who has provided substantial support. His contribution, including organizational effort and regular meetings, has been significant and invaluable to this project.

My gratitude extends to apl. Prof. Dr. Markus Reischl, whose constructive feedback during the midterm presentation significantly influenced the path of this research. His academic insights are appreciated.

I would also like to acknowledge my father, for his unwavering and consistent support throughout my life and studies. His patience and enduring belief in my potential have not only been a source of strength but also a continuous motivator. His support has been instrumental in every step I've taken, fostering my personal and academic growth.

Finally, I am deeply grateful to my wife. Her unwavering support, patience, and dedication throughout these years have been of utmost importance to me. Her belief in me has motivated me constantly during this journey.

In conclusion, I want to express my heartfelt thanks to everyone who contributed to this work. Your collective efforts and support have been fundamental to completing this thesis.

Contents

List of Tables

List of Figures

1. Introduction

Deep learning is a subfield of Machine Learning (ML) and has revolutionized the state-of-the-art in speech recognition [84], visual object recognition [75], object detection [310] and a lot of other domains like drug discovery [50] and genomics [318]. The following definition describes deep learning in one sentence as part of a set of definitions from [65].

> "A class of machine learning techniques that exploit many layers of non-linear information processing for supervised or unsupervised feature extraction and transformation, and for pattern analysis and classification." Li Deng, Dong Yu [65]

The critical aspects of all definitions are that a model consists of multiple layers corresponding to different levels of abstraction to learn complex relationships among data. The famous paper »*Deep Learning*« [145] describes it as a technique to discover intricate structures in large data sets by using the backpropagation algorithm to indicate how a machine should change its internal parameters that are used to compute the representation in each layer from the representation in the previous layer.

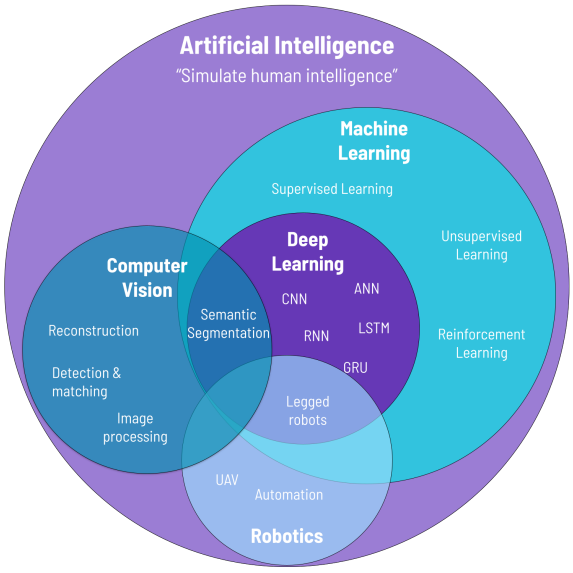


Fig. 1.1. Overview of different techniques within the large field of Artificial Intelligence (AI). The image is inspired by [117].

There are several reasons why deep learning has become as popular as it is today. One reason is the in-

creased chip processing ability which led to increased performance in parallel computing systems such as GPU clusters, or the emergence of large-scale annotated training data [65][309][49]. Additionally, there was significant research in the design of network structures and training strategies like dropout [85], data augmentation [231] or batch normalization[119], which led to several breakthroughs in object recognition [21], speech [21] or video processing [145].

One essential task relevant to this project is semantic segmentation within the ML and deep learning field. Fig. 1.1 provides an overview of AI and its most important subfields. We can see how semantic segmentation is also related to Computer Vision (CV) intersecting with Deep Learning.

The goal of semantic segmentation is to assign a class label to each pixel in an image with numerous practical applications in various fields, such as autonomous driving, where it helps the car to understand the environment around it by segmenting objects in the scene such as roads, traffic lights, pedestrians or other vehicles and assists in deciding on how to navigate [234] safely [26][47][313]. In medical imaging where it can be used to identify and segment tumors, organs, or other structures, helping doctors to diagnose diseases accurately, plan surgeries, and monitor the clinical progress [12][206][166]. In robotics, semantic segmentation can be used to help robots to interact with environments by detecting obstacles for pathfinding or for pick and place tasks [178][179][274]. Semantic segmentation can also assist in surveillance to track and detect objects such as people [125], animals, vehicles, or cargo for logistics companies [42]. It can further assist in border control [18][283], wildlife monitoring [104] or crowdmanagement [276]. In the field of Augmented Reality (AR) it can be used to place virtual objects in the real world, track the objects' movements and help to make the appearance of virtual objects seamless and more natural [250][137].

One crucial step in preparing data for deep learning, especially for techniques such as semantic segmentation, is to provide an accurately labeled dataset. The process involves assigning a label or tag to each data point which indicates which category or class it belongs to [295]. For semantic segmentation, labels consist of images with annotated pixels with a label corresponding to the object or category typically represented as integers. If no dataset is available, the labeling task requires extensive effort manually or by applying semi-automatic or automatic annotation mechanisms [203][214]. For any of these techniques, providing a very high quality of the labeled data is essential as otherwise, the model will not be capable of recognizing patterns accurately [9][187]. While labeling images for some fields is pretty straightforward and requires no specialist knowledge, the cost to obtain annotated images for medical image analysis is much higher [286]. Large labeled datasets are pretty rare; thus, extensive research has been undertaken to improve performance on small scaled or incomplete datasets [305][20][182][282][90].

Given the importance of training with limited data, this project aims to improve segmentation performance for small datasets with biomedical imagery by proposing an end-to-end training architecture with a custom loss merging functionality. The proposed pipeline further gives the user an intuition about suitable loss combinations for any input dataset by providing a final score for combined loss functions. The architecture components are set up in a way to deal with highly unbalanced datasets, which provide additional challenges which, in practice, are very common when dealing with "real world" data [45] [36].

The contribution of this project can be summarized as follows:

1. ***U-Net based segmentation architecture***: Implementing a robust segmentation architecture using

the U-Net model.

2. ***Baseline training***: Training of several baseline models on three highly unbalanced datasets of different difficulties, using six distinct loss functions.

3. ***Loss function analysis***: In-depth comparison and analysis of six standard loss functions and presentation of a method for combining them effectively.

4. ***Loss merging framework***: Development of a fully automatic loss merging experimentation framework enabling users to add, combine easily, and weight loss functions to find the most suitable combination for a given dataset.

5. ***Insightful experimentation***: Extensive tests to gain insight into suitable loss combinations and merging strategies for optimal performance.

6. ***Public availability***: Deployment as a publicly available software package, allowing others to benefit from the research and findings.

1.1. Structural Design

Chapter 1 describes the scientific problem and goals to achieve with this project. The chapter briefly reviews semantic segmentation, its importance, and its relation within the deep learning context. The chapter closes by defining the document's contributions and structural design. Chapter 2 provides an extensive review of Machine Learning (ML), semantic segmentation, and its relation to Computer Vision (CV) for the reader to get a better understanding of the research problem and approach. It further introduces several segmentation objectives, metrics, the used architectures and a standard deep learning framework. Chapter 3 includes an extensive summary of techniques for improving segmentation with a focus on custom loss functions. Chapter 4 describes the proposed methodology starting with an in-depth analysis of limiting factors for loss functions and then discussing numerous merging strategies to combine losses with the goal to improve segmentation performance. Chapter 5 introduces the experimental setup, including the dataset, parameter configurations, preprocessing steps, training, validation, testing, and monitoring settings, while the Chapter 6 presents the implementation details, such as the programming language, the deep learning framework used, management tools, the network architecture, and a custom analysis spreadsheet. Chapter 7 discusses the results achieved structured by dataset and Chapter 8 summarizes the achieved results. Chapter 9 provides an overall reflection of the main contributions, discusses their significance, and outlines their limitations to suggest directions for future work.

2. Fundamentals

This chapter introduces the terminology employed throughout this work, followed by a brief overview of Machine Learning (ML) fundamentals, including supervised, unsupervised, and reinforcement learning. The focus then shifts to semantic segmentation, exploring its connection to Computer Vision (CV), segmentation objectives, metrics, and architectures.

2.1. Terminology

This section provides the terminology used throughout this work to guarantee consistency for mathematical definitions. The terms are commonly used within ML and have been adapted to fit the context of semantic segmentation.

Input features $x \in X$: Data from the input space X which we want to use to make predictions for. In the case of semantic segmentation, every input x can be defined as a tensor of shape $([C, H, W])$ where C is the number of channels such as red-green-blue (RGB), H the height and W the width of the image. If we use batches of images, we describe them as a tensor of $([N, C, H, W])$ where N is the number of batches.

Output labels $y \in Y$: Targets from the label space which we want to predict. Labels consist of tensors of shape $([N, H, W])$ where N is the batch size, H is the height, and W is the image's width. For training, we usually convert the tensor $([N, H, W])$ into a one-hot-encoded format of shape $([N, D, H, W])$ where D stands for the number of distinct classes.

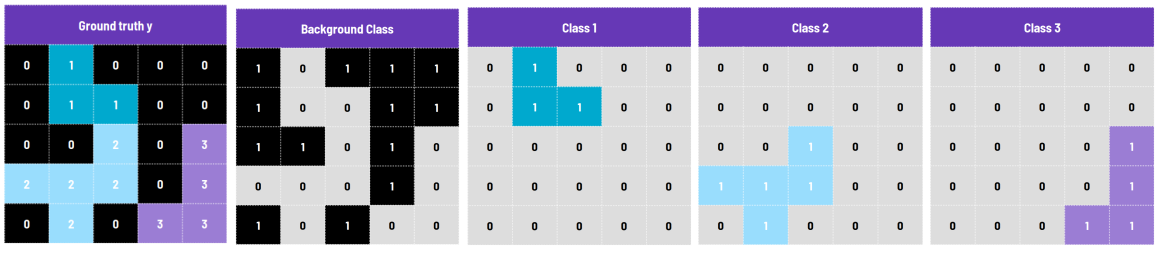


Fig. 2.1. Example of integer-encoded output label (a) transferred to a one-hot-encoded label (b-e) of shape $([N, D, H, W])$. In this example, the batch size $N = 1$

Probability distribution $\mathbb{P}(XY)$: Unknown distribution of input features and output labels. A subset from this distribution would be a dataset $D = \{(x_i, y_i)\}_{i=1}^{m} \in (X, Y) \sim \mathbb{P}(X, Y)$.

(a) (b)

Fig. 2.2. Example of sample (a) label (b) pair $(x_1, y_1) \in (X, Y)$ of the Medaka fish dataset with four classes (Bulbus, Ventricle, Atrium, and Background)

Model $h_\theta : X \to \hat{Y}$: Function to map features from the input space X to the prediction space $\hat{Y}$ depending on θ. Within semantic segmentation, h_θ consists of a Convolutional Neural Network (CNN).

Parameters $\theta \in \Theta$: Weights of the model which are intended to be optimized during the training phase of the model.

Predictions $\hat{y} \in \hat{Y}$: Outputs of the model h_θ. The value typically consists of scores inferred from the model h_θ. For segmentation tasks, the output shape $\hat{Y}$ is usually of the form $([N, D, H, W])$ such as Y, and is then optionally transformed into probabilities as defined in Section B.1.

Loss function $l : \hat{Y} \times Y \to \mathbb{R}_+$: Also known as cost function measures the difference between the prediction $\hat{Y}$ and the output label Y. The goal of a machine learning model is to minimize the value of the loss function by providing a gradient or direction for the model to make.

2.2. Machine Learning

Machine Learning (ML) is a subfield of Artificial Intelligence (AI). The paper »*Machine learning: Trends, perspectives, and prospects*«[123] provides an excellent summary of ML and defines it as the question of »*How to build computers that improve automatically through experience*«. The paper starts by outlining its importance as one of today's most rapidly growing technical fields and as a method of choice in Computer Vision (CV), Natural Language Processing (NLP), Speech Recognition (SR), or Robot Control (RC). It formally defines ML as a learning problem to improve some measure of performance when executing some task through some training experience. Further, it describes the three most important paradigms of ML known as Supervised Learning (SL), Unsupervised Learning (UL), and Reinforcement Learning (RL).

2.2.1. Supervised Learning

Supervised Learning (SL) techniques are nowadays the most widely used methods [123]. Those methods are based on the principle known as Empirical Risk Minimization (ERM) from statistical learning theory [268]. It is defined as an approach of the »*expected risk*« as

$$R(h_\theta) = \mathbb{E}[L(h(X), Y)] = \int L(h(X), Y) d\mathbb{P}(X, Y) \tag{2.1}$$

with the »*empirical risk*« as

$$R_\delta(h_\theta) = \int l(h_\theta(X), Y) d\mathbb{P}_\delta(X, Y) = \frac{1}{m} \sum_{i=1}^{m} l(h_\theta(x_i), y_i) \tag{2.2}$$

The idea is that even if we do not know the real joint distribution of a sample, label pairs $\mathbb{P}(X, Y)$, we still can approach it with some known training data. The general goal of a SL algorithm is to find a function $h \in H$ which models the relationship between a random feature vector X and a random target vector Y, also called output label, by following the joint distribution $\mathbb{P}(X, Y)$. L is the loss function which penalizes the differences between prediction $h(x)$ and Y. $\mathbb{E}[L(h(x), y)]$ is the expected loss if we obtain samples from the same distribution $\mathbb{P}(x, y)$ as the training data [302]. The difference between 2.1 and 2.2 is the distribution from where the data is sampled. While the former samples the data from the real unknown distribution $\mathbb{P}(X, Y)$, the latter is an approximation obtained from the distribution $\mathbb{P}_\delta(X, Y)$ wich represents samples from a dataset $DS = \{(x_i, y_i)\}_{i=1}^{m}$.

The inputs of the dataset DS can have numerous forms, such as simple vectors, DNA sequences, images, or graphs [123]. For SL methods, the output y typically is either a category used for classification or a real value for regression methods. The following section will describe some of the most essential SL methods and briefly outline how to optimize the »*empirical risk*« defined as 2.2.

Supervised Learning Models

Linear Regression (LR) is a supervised technique used to model the relationship between a dependent and one or more independent variables. The dependent variable is a real value. Some applications for LR are the prediction of stock prices for financial institutes [24], sales volume [254], customer lifetime value [168] or churn rates [97] for sales and marketing departments [170], predicting patient outcomes [149] or healthcare costs [241], modeling the relationship between environmental variables such as temperature [291], humidity [265] or precipitation [181] and their impact on natural systems for environmental scientists [229]. Mathematically, the linear regression model can be expressed as

$$h_\theta(x) = \sum_{j=1}^{n} \theta_j x_j = \theta^T x \tag{2.3}$$

where x is the dependent variable, $h_\theta(x)$ the independent variables and n the number of features. θ are the coefficients or parameters. As a loss function, we could use the squared error or l_1 loss defined as

$$l_1(h_\theta(x), y) = \sum_{i=1}^{m} (\theta^T x_i - y)^2 \tag{2.4}$$

or the absolute loss or l_2 as

$$l_2(h_\theta(x), y) = \sum_{i=1}^{m} |\theta^T x_i - y| \tag{2.5}$$

which is less sensitive to outliers [253][245]. Note that m is the number of samples in the dataset.

Logistic Regression (LogR) is another supervised technique but used for classification problems. The dependent variable is based on a logistic model, which typically estimates the probability of an outcome of two classes. There are many domains where LogR can be applied. Some of them are fraud detection [217], customer churn prediction [64], medical diagnosis [259] or sentiment analysis [202]. For LogR, we can use the same linear model as defined in 2.3 but with a different loss function as

$$\begin{aligned} l_{\log}(h_\theta(x), y) &= \sum_{i=1}^{m} y_i \log(\theta^T x_i) + (1 - y_i) \log(1 - (\theta^T x_i)) \\ &= \sum_{i=1}^{m} y_i \theta^T x_i - \log(1 + \exp(\theta^T x_i)) \end{aligned} \tag{2.6}$$

which is also called the log-likelihood function [103].

Tree Based Methods (TBM) are also SL methods which can be either used for classification or regression tasks. TBM can be used with ensemble methods to improve performance. The core idea of ensembles is that, on average, they perform better than individual models. New predictions are then made by averaging [103] or voting [59]. Averaging is used for regression and voting for classification tasks. The advantage of decision trees is that they are easy to create [8] and interpret [136], that they require fewer data preparation, and that they are insensitive against outliers or missing data [252]. Formally the model of TBM can be described as

$$h_\theta(x) = \sum_{m=1}^{M} c_m \mathbb{1}\{x \in R_m\} \tag{2.7}$$

The model consists of a sum of constants c, if x is within a region R_m of the decision tree. The constants c_m here depend on the task and can be the proportion of each class $c_m = \frac{1}{N_m} \sum_{x_i \in R_m} \mathbb{1}\{y_i = k\}$ for Classification Trees (CTs) or the mean of all labels $c_m = \frac{1}{N_m} \sum_{x_i \in R_m} y_i$ for Regression Trees (RT) [103]. There are numerous popular variants of decision trees such as Random Forests (RF) [62], Gradient Boost (GB) [130] or Ada Boost (AB) [289].

K-nearest Neighbors (KNN) is a non-parametric SL algorithm. KNN can be used for either classification or regression tasks. The predictions for KNN are made based on the voting of the majority class among its k nearest neighbors in the training set [257]. The advantages of KNN are that it is straightforward and easy to understand, that it can achieve high accuracy by using different distance metrics [263], that it can handle non-linear and complex data sets by using kernel functions and that it adapts to changing data distributions by updating nearest neighbors dynamically [306]. Some disadvantages are that it is slow and computationally expensive for large datasets with many classes [263], its sensitivity to irrelevant features [306], and that it suffers from the curse of dimensionality [141]. Formally the model can be described as

$$h(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i \tag{2.8}$$

where $N_k(x)$ is the neihborhod of x defined by the k closest points x_i in the data set [103]. This model type can be used for classification where $y = \{-1, +1\}$ or regression where y is a real value.

Support Vector Machines (SVMs) are also SL models that can achieve good performance and generalization by using different kernels to handle different types of data and problems. The core idea of SVMs is to find a linear or non-linear function that maps the input data to a higher-dimensional space where a separating hyperplane with a maximal margin can be found. The hyperplane is defined by a set of points, called support vectors which lie on the boundary of the margin [197] [106]. SVMs have been extensively used in multiple domains such as text categorization [256], face detection [191] or neuroimaging analysis [198]. There are numerous types of SVMs. The soft margin support vector machine in primal form is a minimization of the objective

$$\begin{aligned} \min_{\theta,b} \quad & \frac{1}{2}||\theta||^2 + C\sum_{i=1}^{m}\xi_i \\ \text{subject to} \quad & y_i(\theta^T f(x_i) + b) \geq 1 - \xi_i \quad i = 1, \ldots, m \\ & \xi_i \geq 0 \qquad\qquad\quad i = 1, \ldots, m \end{aligned} \tag{2.9}$$

where C permits training instances to be inside the margin or on the wrong side of the separating hyperplane if increased and can thus be considered a regularization term [13]. $f(x)$ is the feature function projecting the inputs into a high feature space where a linear decision surface is constructed [61].

Neural Networks (NNs) have been invented originally by Frank Rosenblatt (1963) as the multi-layer perceptron [210]. Deep learning as a subset of ML uses NN with many layers and has revolutionized the state-of-the-art in numerous fields as described in Chapter 1. Generally, a NN is a universal function approximator consisting of piecewise linear components [53]. Formally the model of a simple one layer Feed Forward Network (FFN) can be described as

$$\begin{aligned} z_i &= \phi(c_{0i} + u_i^T X),\ i = 1, \ldots, I, \\ p_k &= b_{0k} + \theta_k^T Z,\ k = 1, \ldots, K, \\ h_k(X) &= g_k(P),\ k = 1, \ldots, K, \end{aligned} \tag{2.10}$$

where the matrices $Z = (z_1, z_2, \ldots, z_I)$ and $P = (p_1, p_2, \ldots, p_K)$ join the individual components, K represents the number of target classes, and ϕ the activation function of the network responsible for its non-linearity [103]. $c_{0i}, i = 1, \ldots, I$ and $b_{0k}, k = 1, \ldots, K$ are the bias terms which provides greater flexibility. g_k is the output function. We can use the softmax function described in B.1 for a classification task. We would then have $g_k(P) = \sigma(P)$. On the other hand, we can use $g_k(P) = P_k$ [103] for a regression task.

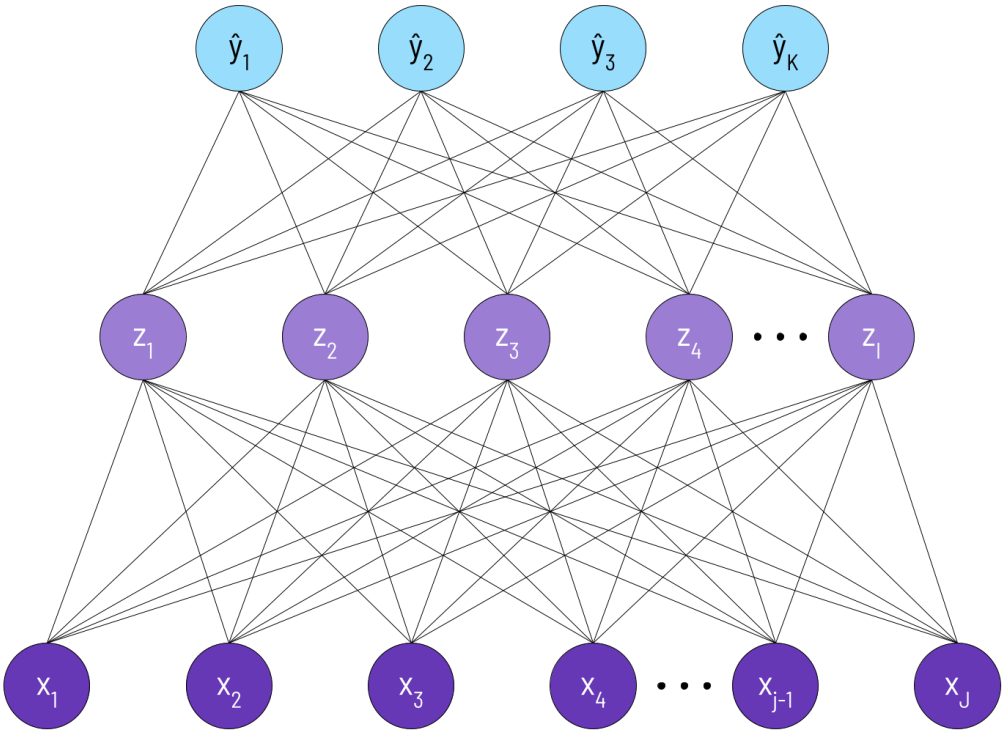


Fig. 2.3. Feed forward network with one hidden layer. The illustration depicted is inspired by the schematic described in [103].

Name	Formula
Sigmoid	$\phi(z) = \frac{1}{1+e^{-z}}$
Tanh	$\phi(z) = \frac{e^{2z}-1}{e^{2z}+1}$
Relu	$\phi(z) = \max\{0, z\}$

Tab. 2.1. List of some popular activation functions used in NNs

Numerous activation functions are available for ϕ, so the choice depends on data distribution, network architecture, task objective, and more [188]. See 2.1 for a list of some popular activation functions. The ReLU activation function, for example, is widely used in hidden layers due to its simplicity and effectiveness [146][73]. Generally, it is essential to add non-linearity to the NN to make the model learn, represent and process any data and any arbitrary complex function to map inputs to outputs [228], which on the other hand, requires minimizing an NP-hard high-dimensional non-convex objective [150]. This can be thought of as finding a global minimum in a chaotic, highly non-convex landscape as depicted in Fig. 2.4.

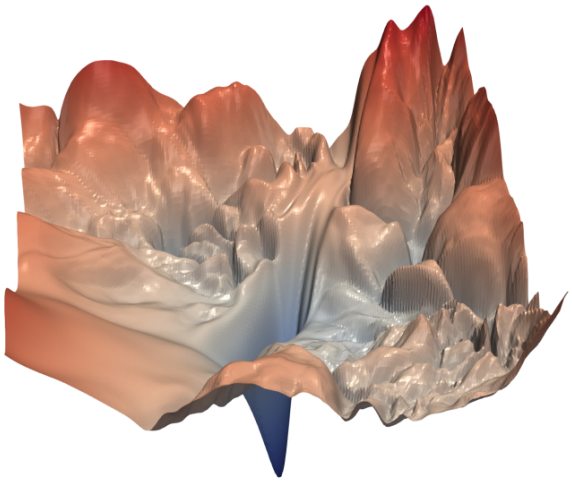


Fig. 2.4. Highly non-convex landscape describing the complex objective of a Neural Network. The image is taken from [150].

This section was a brief review of important SL methods. As we can see from the definition of the »*empirical risk*« 2.2, all supervised methods require a model h, a set of labeled data, and some loss function typically chosen based on the task to solve, which can be a classification or regression task. Additionally, the equation 2.2 must be minimized, which leads to optimization theory briefly described in the next section.

Optimization

As we need to minimize the »*empirical risk*« as defined in 2.2 we can write the final objective as

$$\min_{\theta} J(\theta) = \frac{1}{m} \sum_{i=1}^{m} l(h_\theta(x_i), y_i) \tag{2.11}$$

which formally means that we want to find parameters θ that minimize the average losses of the training data.

The optimization techniques to solve this problem directly depend on the algorithm. While for some fundamental methods, such as linear regression, we can use analytic solutions, NN requires an optimization technique called gradient descent which minimizes the objective by updating the parameters in the opposite direction of the gradient of the objective function concerning the parameters θ [212]. Intuitively we can think of trying to find a path to reach at least a local minimum on a non-convex surface as illustrated in Fig. 2.4.

One of the most critical parameters in deep learning optimization is the learning rate, which controls the size of the steps taken during each iteration of the optimization process [92]. A low learning rate can result in slower convergence, while a high learning rate might hinder finding an appropriate minimum. In practice, adaptive learning rate methods can automatically adjust the learning rate during training [300].

Gradient descent is a widely-used optimization method in deep learning, and numerous variants have been developed to improve its performance. One such variant is stochastic gradient descent (SGD), which estimates the gradient instead of calculating the true gradient. This approach often leads to faster convergence compared to traditional gradient descent [29].

2.2.2. Unsupervised Learning

A prominent subject within ML is ULs, which explores data to recognize inherent patterns without the guidance of predefined output labels. Two big fields of UL are Clustering [174] and Dimensionality Reduction [240]. Some applications of Clustering are Recommender Systems (RSs) [163], Customer Segregation (CS) [224] and Targetet Marketing (TM) [196] and Dimensionality Reduction includes topics as Feature Elicitation (FE) [30] Structure Discovery (SD) [273] Meaningful Compression (MC) [83] or Big data Visualization (BDV) [132]. Numerous methods combine supervised with unsupervised techniques, such as semi-supervised, self-supervised, or weakly supervised. While semi-supervised learning uses labeled and unlabeled data, self-supervised learning attempts to generate labeled data independently, and weakly-supervised learning uses only partial information about the label [219]. As this project focuses on SL techniques, please refer to the keywords and papers mentioned above for further information.

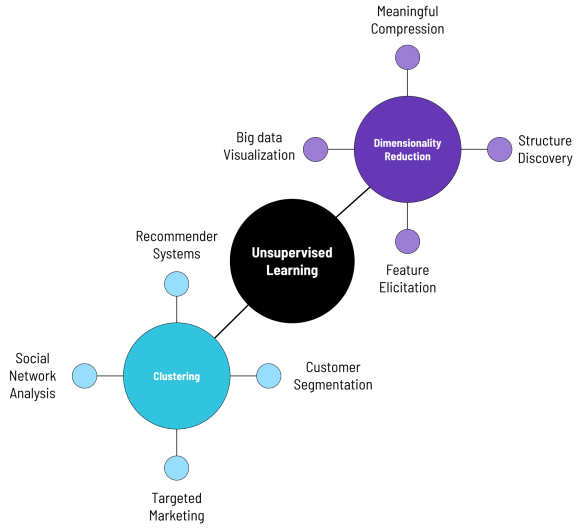


Fig. 2.5. Applications of Unsupervised Learning (UL). The image is inspired from [233]

2.2.3. Reinforcement Learning

Another large field of ML is Reinforcement Learning (RL), which relies on using training data to evaluate actions to take. In RL, the correct action leads to the choice of subsequent actions. While the agent[1] is not

[1]The intelligent agent is a term used extensively in the literature. The book [213] defines AI as the study of agents that receive percepts from the environment and perform actions.

told what actions to complete, it tries to discover what action can produce a maximum feedback signal also known as reward [95]. RL, therefore, is a trial-and-error mechanism that learns through feedback [121].

Fig. 2.6 illustrates a typical concept of RL with an agent connected to its environment via perception and action. On every step, the agent receives as input i an indication of the current state of the environment. Subsequently, the agent selects an action a, which changes the state of the environment. This state transition is then communicated to the agent through a scalar reinforcement signal r. The goal for the agent is to choose actions that increase the long-run sum of values of the reinforcement signal [126].

Numerous algorithms correspond to RL. Those include dynamic programming [40], Monte Carlo methods [99], Q-Learning [101], TD-Learning [204] or Sarsa algorithms [319].

An important domain where RL can be applied is in robotics, where RL can help how to grasp objects [124], interact with humans [6] or other robots [169], or navigate through difficult environments [159]. Another domain is finance, where RL can assist in portfolio management [122], trading [67], or risk management [122]. RL can further assist in healthcare with providing diagnosis [128], treatment recommendation [93], drug discovery [315] or be used for surgical robots [207]. In the gaming field, RL can help create intelligent agents that can play games such as chess [237], Go [236], poker [107] or even video games [225]. Additionally, RL can be applied to CV tasks such as feature detection [39], image segmentation [216], object recognition [192] or tracking [299]. Chapter 3 will describe in more detail of how RL methods can improve segmentation results.

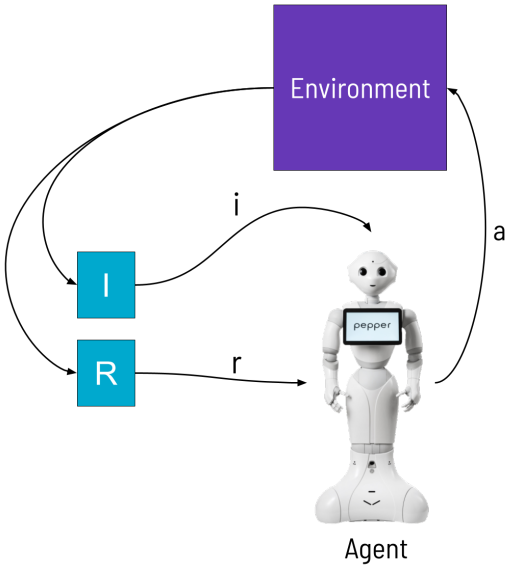


Fig. 2.6. Typical concept of Reinforcement Learning (RL) algorithm.

2.3. Semantic Segmentation

This section offers a comprehensive overview of semantic segmentation using deep learning techniques. Semantic segmentation is a pixel-level classification algorithm that is crucial in comprehensively understanding images. According to [88], many applications benefit from deriving insights from visual data. Notably, semantic segmentation has potential applications in a diverse range of fields, including but not limited to:

- Autonomous driving [77] [81] [258] [234] [26]
- Human-machine interaction [189]
- Computational photography [294]
- Image search engines [275]
- Augmented reality [137] [303]
- Medical imaging and diagnostics [12] [135] [72]
- Facial recognition [175] [134]

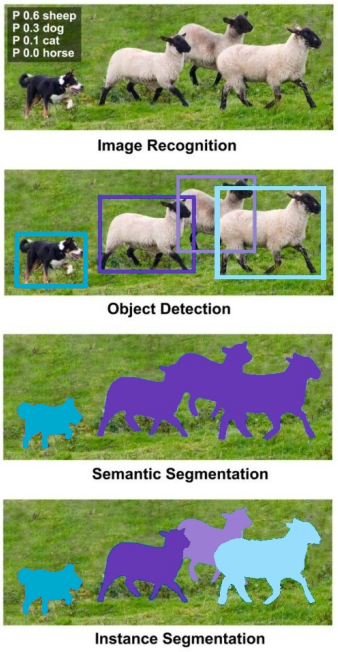


Fig. 2.7. Semantic segmentation within the context of scene understanding. Image from [34].

Fig. 2.7 illustrates four categories of scene understanding. *Image recognition* provides a probability of objects detected within an image but does not give any information about their exact location. *Object detection*, on the other hand, tries to locate and draw bounding boxes around every object in the image. *Semantic segmentation* classifies pixels into preset categories. In the example shown, sheep and dogs are distinguished as separate categories. Finally, *instance segmentation*, similar to semantic segmentation, can differentiate individual objects within the same class.

2.3.1. Segmentation within Computer Vision

As illustrated in Fig. 1.1, semantic segmentation is a subfield of Computer Vision (CV) and Machine Learning (ML). CV, as a multidisciplinary field, encompasses a wide range of tasks and applications. Shapiro and Stockman define the objective of computer vision: »*The goal of computer vision is to make valuable decisions about real physical objects and scenes based on sensed images*« [226].

The authors describe Sensing, Encoded Information, Representations, and Algorithms as critical aspects of computer vision. *Sensing* refers to how sensors acquire images and encode properties such as material, shape, illumination, and spatial relationships. *Encoded Information* pertains to extracting information from images for a deeper understanding, including the geometry, texture, motion, or identity of objects. *Representations* involve how images are represented for further use, considering their parts, properties, or relationships. *Algorithms* describe the methods used to extract or process image information.[226]

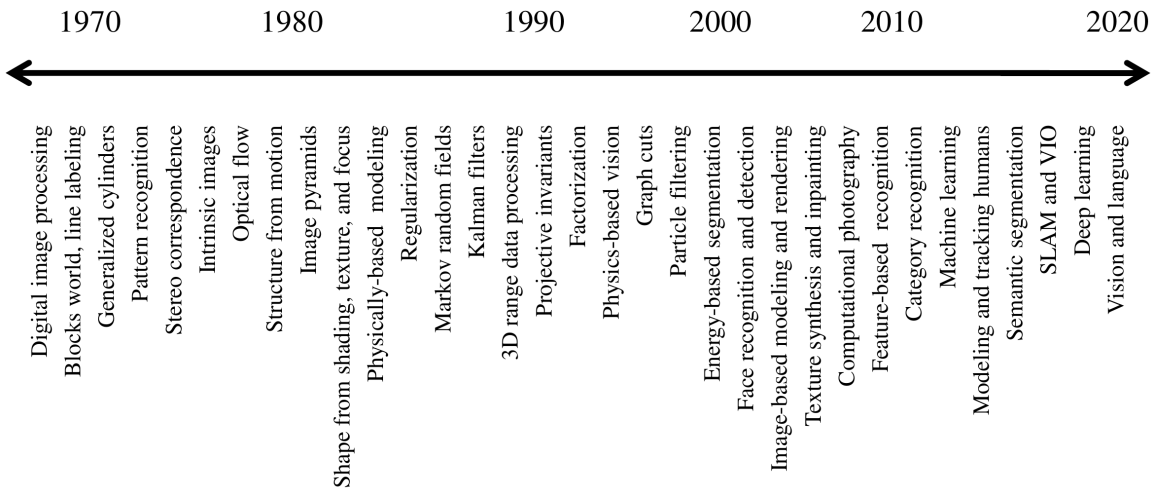


Fig. 2.8. Timeline of CV research. Image from [249]

Many algorithms used in computer vision were developed in studies dating back to the 1970s. [249] provides a fascinating timeline of the most important research topics in computer vision, as shown in Fig. 2.8. A significant breakthrough occurred in 2006 when Hinton introduced the Deep Belief Network with multiple layers of Restricted Boltzmann Machines. The author further states that the availability of large, public datasets and the advancement of parallel GPU computing contributed to the tremendous growth of deep learning.

The following chapter delves into some fundamental concepts of semantic segmentation, which, according to the timeline in Fig. 2.8, is a relatively new field within computer vision.

2.3.2. Segmentation Objectives

As briefly mentioned above, and as illustrated in Fig. 2.7, semantic segmentation aims to classify each pixel in an input image into a specific class. Applying the empirical risk introduced earlier to semantic segmentation, we have

$$\min_{\theta} J(\theta) = \frac{1}{m} \sum_{i=1}^{m} l(h_\theta(x_i), y_i) \tag{2.12}$$

where m is the number of images in the dataset, h_θ the model, which typically consists of some form of CNN architecture, x the input image, y the output label and l, the loss function which quantifies the difference between a prediction and the ground truth.

Numerous loss functions are suitable for semantic segmentation, which can be divided into different categories [164]. Subsequent sections offer an in-depth exploration of six notable loss functions, with two representatives from each category: distribution-based, region-based, and boundary-based, summarized in the following table 2.2.

Types	Names
Distribution-based	Cross-Entropy Loss (CE), Focal Loss (FL)
Region-based	Dice Loss (DL), Tversky Loss (TL)
Boundary-based	Hausdorff Loss (HL), Boundary Loss (BL)

Tab. 2.2. Types of loss functions

Distribution-based

Distribution-based loss functions focus on minimizing the dissimilarity of two distributions [164]. A principal loss function of this type is the Cross-Entropy Loss (CE). Suppose a given segmentation task has two classes, we can use the Binary Cross Entropy loss (BCE) derived from the Bernoulli distribution, and for a multi-class problem, we would use the Categorical Cross Entropy Loss (CCE) derived from Multinoulli distribution [120].

Cross-Entropy Loss (CE)

The binary cross-entropy loss can be formally defined as

$$CE_b = -y \log(\hat{y}) - (1-y) \log(1-\hat{y}) \tag{2.13}$$

For the case of multi-class classification, we would use the categorical cross-entropy loss defined as

$$CE_{mc} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{c=1}^{C} y_{i,c} \cdot \log(\sigma_{i,c}) \tag{2.14}$$

where N is the number of pixels, C is the number of classes, and σ_i is the normalized pixel with the softmax activation function as defined in B.1.

The CE or some variations are widely used for semantic segmentation. Prior studies claimed it works best in equal data distributions among classes [120]. If the data is highly unbalanced, it can lead to the over-representation of larger objects in the loss, thus resulting in lower performance of smaller objects [292].

Focal Loss (FL)

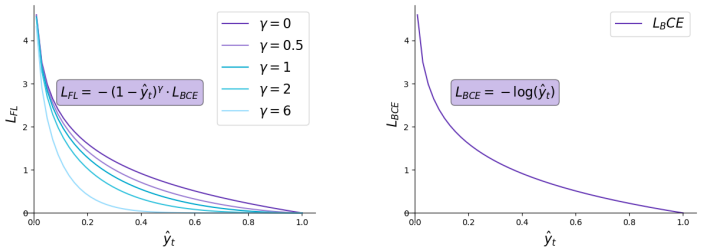
The FL [157] is a variant of the cross entropy loss. It provides excellent performance for highly imbalanced datasets by down-weighting easy examples and focusing on learning harder ones. The focal loss for binary classification is formally defined as

$$FL_b = -\alpha(1-\hat{y}_t)^{\gamma} \cdot L_{BCE} \tag{2.15}$$

and for multi-class classification as

$$FL_{mc} = -\alpha(1-(\sigma_{i,c}))^{\gamma} \cdot L_{CCE} \tag{2.16}$$

The α and β parameters control how hard wrong predictions get penalized. Fig. 2.9 provides an example and the relation of FL to CE.



(a) (b)

Fig. 2.9. FL (a) as proposed in [157] with different values of γ. Whenever $\gamma = 0$ focal loss turns into the regular CE (b). Increasing γ reduces the loss for well-classified examples, e.g., $\hat{y}_t > 0.6$ leading to a higher focus on misclassifications for hard examples.

Region-based

Region-based loss functions primarily focus on the segmented regions within an image to ensure that the predicted output accurately captures the distinct areas or objects of interest. Region-based loss functions are also considered a prevalent type used in computer vision tasks such as semantic segmentation or object detection. The main difference between region-based and distribution-based loss functions is how they measure the difference between $\hat{y}$ and y. While region-based loss functions penalize incorrect classifications of individual pixels or regions, distribution-based loss functions penalize differences between the distribution of predicted class labels and ground truth labels.

Dice Loss (DL)

A popular region-based loss function for medical segmentation is the Dice Loss (DL) based on the Dice Similarity Coefficient (DSC). The DSC is defined as

$$DSC = \frac{2|\hat{y} \cap y|}{|\hat{y}| + |y|} \tag{2.17}$$

where $\hat{y}$ is the set of predicted pixels and y the set of actual pixels. The numerator represents two times the cardinality of the intersection of the predicted and the actual set while the denominator is the sum of the predicted and actual cardinality.

As the DSC is defined as a discrete set operation that is not differentiable [308], it cannot be directly used for training. To obtain a probabilistic version of the DSC, an approximation known as the DL was designed. The DL penalizes the pixel-wise mismatch between a predicted map $\hat{y}$ and its corresponding ground truth y for each class j. It is defined as

$$DL = 1 - \frac{2\sum_{i=1}^{P}\sum_{j=1}^{C} y_{i,j}\hat{y}_{i,j} + \epsilon}{\sum_{i=1}^{P}\sum_{j=1}^{C}(y_{i,j} + \hat{y}_{i,j}) + \epsilon} \tag{2.18}$$

where P is the number of pixels, C the number of classes and ϵ a smoothing constant to avoid division by zero.

There are numerous variations of the dice loss. [247] from 2017 introduces the Generalized Dice Loss (GDL) which has the form

$$GDL = 1 - 2\frac{\sum_{l=1}^{2} w_l \sum_n y_{ln}\hat{y}_{ln}}{\sum_{l=1}^{2} w_l \sum_n y_{ln} + \hat{y}_{ln}} \tag{2.19}$$

where w_l can provide invariance to specific label set properties by correcting the contribution of each label by the inverse of its volume. This technique can reduce the correlation between region size and dice score, which will be further discussed in Section 4.1.

Tversky Loss (TL)

The TL, presented by [218] was inspired by the Tversky index [261] from 1977 defined as

$$S(\hat{y}, y, \alpha, \beta) = \frac{|\hat{y} \cap y|}{|\hat{y} \cap y| + \alpha|\hat{y} \setminus y| + \beta|\hat{y} \setminus y|} \tag{2.20}$$

where $\hat{y}$ and y are sets of predicted and ground truth labels, respectively, and α and β are hyperparameters that control the trade-off between false positives and false negatives.

The TL aims to address the problem of high-precision, low-recall segmentations that can occur in highly imbalanced datasets. See Fig. 2.10. The authors claim this is an undesired property, especially in computer-aided diagnosis or clinical decision support systems where high recall is often more important than high precision.

The TL is defined as follows:

$$TL(\alpha, \beta) = \frac{\sum_{i=1}^{N} \sigma_{1i} y_{1i}}{\sum_{i=1}^{N} \sigma_{1i} y_{1i} + \alpha \sum_{i=1}^{N} \sigma_{1i} y_{0i} + \beta \sum_{i=1}^{N} \sigma_{0i} y_{1i}} \tag{2.21}$$

where σ_{0i} defined in B.1 is the probability of pixel i to be part of the background region and σ_{1i} the probability of pixel i to be part of the foreground. y_{1i} is 1 for a foreground pixel and 0 for a background pixel. y_{0i} would be 0 for a foreground pixel and 1 for a background pixel. With $\alpha = \beta = 1$ the second term of the denominator would calculate the false positive rate and the second term the false negative rate. The hyperparameters α and β control the balance between false positives and false negatives, with higher values of β emphasizing false negatives and increasing recall.

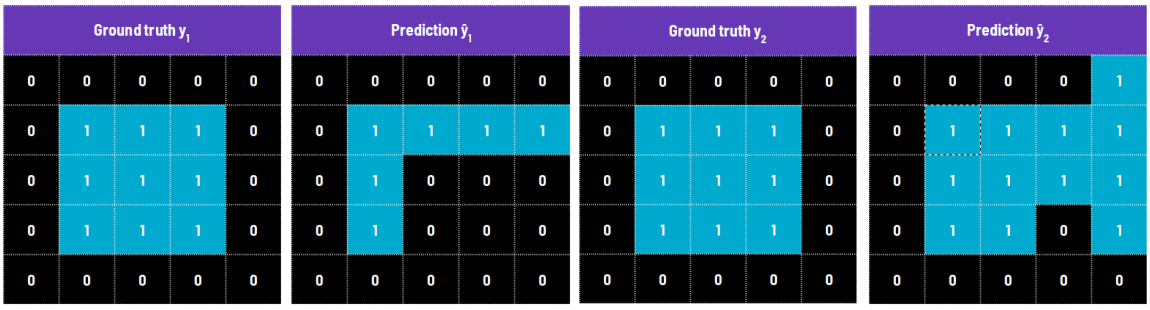


Fig. 2.10. Images (a) and (b) exhibit high precision 0.83 with a low false positive rate and low recall 0.55 with a high false negative rate, whereas images (c) and (d) show the opposite situation with low precision 0.61 and a high false positive rate and high recall 0.88 with a low false negative rate. In heavily imbalanced datasets with a much smaller foreground class than the background class, it is common to encounter situations with high precision and low recall, resulting in many false negative classifications. This result is a critical problem in tasks such as medical image analysis, where detecting abnormalities is more important than achieving high precision [200].

Boundary-based

Boundary-based loss functions represent a specialized type of loss functions that offers supplementary distance information, addressing the limitations of region-based and distribution-based loss functions. This section comprehensively analyzes two prominent boundary-based loss functions: BL and Hausdorff Loss (HL).

Hausdorff Loss (HL)

The Hausdorff loss was presented in the paper »*Reducing the Hausdorff Distance in Medical Image Segmentation with Convolutional Neural Networks*«[129] aiming to reduce an approximation of the Hausdorf Distance (HD). The authors claim that the HD is one of the most informative and useful objective criteria as an indicator for the largest segmentation error. The HD was first introduced in the book called »*Grundzüge der Mengenlehre*« by Felix Hausdorff in the year 1914. Applied to the definitions of y and $\hat{y}$ presented in Section 2.1, defining their corresponding boundaries as Y_b and $\hat{Y}_b$ the calculation for the one-sided HD from the pointset $\hat{Y}_b$ to Y_b is defined as follows [208]:

$$hd(\hat{Y}_b, Y_b) = \max_{\hat{y}_b \in \hat{Y}_b} \min_{y_b \in Y_b} ||\hat{y}_b - y_b||_2 \tag{2.22}$$

and from the set Y_b to $\hat{Y}_b$ as

$$hd(Y_b, \hat{Y}_b) = \max_{y_b \in Y_b} \min_{\hat{y}_b \in \hat{Y}_b} ||\hat{y}_b - y_b||_2 \tag{2.23}$$

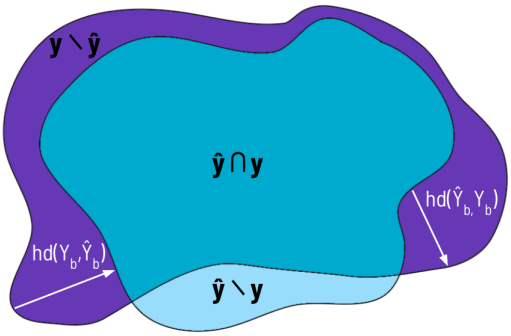


Fig. 2.11. Visual concept of the Hausdorff distance and some related notation. The two distances $hd(\hat{Y}_b, Y_b)$ and $hd(Y_b, \hat{Y}_b)$ represent the two largest segmentation errors from both the prediction boundary and output label boundary perspective. The contour additionally displays **false positives** as $\hat{y} \setminus y$, **false negatives** as $y \setminus \hat{y}$ and **true positives** as $\hat{y} \cap y$. The image is inspired by the visuals of [129]

When calculating $hd(\hat{Y}_b, Y_b)$, we can intuitively say that for every point $\hat{y}_{b,i}$ on the boundary $\hat{Y}$ we calculate the closest distance to the boundary Y_b and then chose the largest value from all those closest distances. As we can see in the Fig. 2.11, $hd(y_b, \hat{y}_b) \neq hd(\hat{y}_b, y_b)$ which should be true in most cases. A bidirectional version of HD, is defined as [129]:

$$HD(Y_b, \hat{Y}_b) = \max(hd(\hat{Y}_b, Y_b), hd(Y_b, \hat{Y}_b)) \tag{2.24}$$

The calculation of the HD for segmentation purposes is shown to be done with the help of the distance transform method [129]. The distance transform technique calculates the distance to a curve or a set of points using a two-pass raster algorithm [249]. Adjusted to the notation of the current work, the formal definition for the distance transform applied to the prediction $\hat{y}$ is defined as:

$$D_{\hat{y}}(i,j) = \min_{k,l\ \hat{y}[k,l]=1} d(i-k, j-l) \tag{2.25}$$

or for y as

$$D_{y}(i,j) = \min_{k,l\ y[k,l]=1} d(i-k, j-l) \tag{2.26}$$

Note that we assume that the input, whether from y or $\hat{y}$, is provided as a binary image. The equations 2.25 and 2.26 calculate the distance for every pixel to the nearest pixel with the value 1. Fig. 2.12 provides an example calculation for $D_{\hat{y}}(i,j)$ and $D_{y}(i,j)$ with input shape of $([1,1,5,5])$ for y and $\hat{y}$.

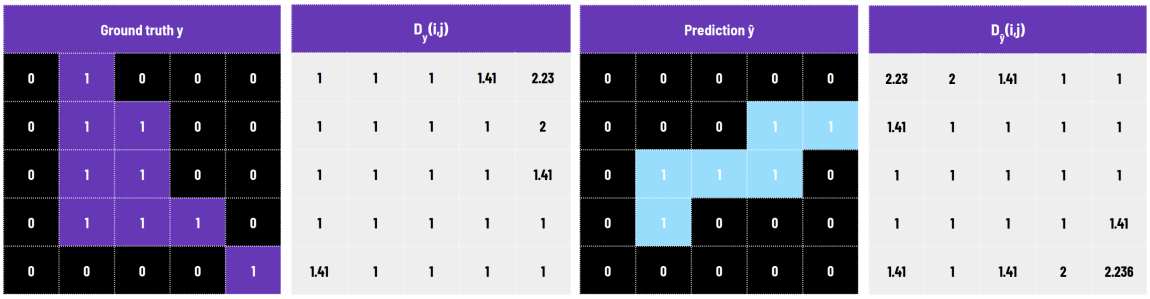


Fig. 2.12. Distance transform calculation for ground truth y (a) and $\hat{y}$ (c). We can see that D_y (b) and $D_{\hat{y}}$ (d) contain values that consist of distances. The position of these distances corresponds to the same position of the input y and $\hat{y}$ and the distances indicate how far this pixel was from the closest pixel with the value 1.

For the final calculation of the HD for segmentation, we first have to subtract y from $\hat{y}$ or vice versa and then take the absolute value from the result. We still input the binary maps of y and $\hat{y}$. The result will highlight false positives and false negatives as a new binary map defined as $|y - \hat{y}|$ or in set notation $y \setminus \hat{y} \cup \hat{y} \setminus y$. See Fig. 2.13 or Fig. 2.11 for further information. To finalize we need calculate $hd_D(Y_b, \hat{Y}_b)$ as

$$hd_D(Y_b, \hat{Y}_b) = \max_{\Omega}(|y - \hat{y}| \circ D_{\hat{y}}(i, j)) \tag{2.27}$$

and $hd_D(\hat{Y}_b, Y_b)$ as

$$hd_D(\hat{Y}_b, Y_b) = \max_{\Omega}(|y - \hat{y}| \circ D_y(i, j)) \tag{2.28}$$

The subscript D on hd_D indicates that we use the distance transform for this type of calculation. $\circ$ stands for the Hadamard product, an element-wise product of two matrices of the same size [113]. Ω represents the grid where the image is established, indicating that the maximum value pertains to all the pixels. Fig. 2.13 provides the result of the example calculation from above. For the bidirectional version, we could take 2.24 as the final loss definition.

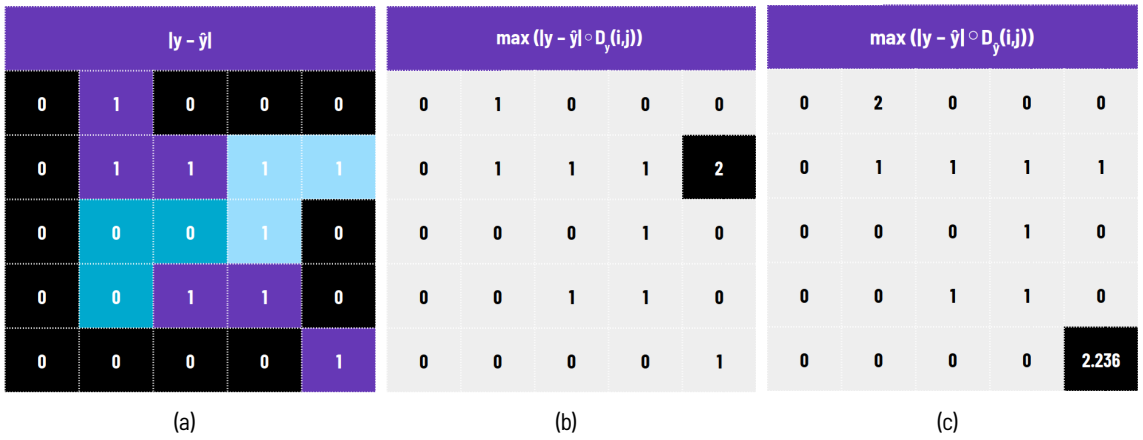


Fig. 2.13. Image (a): **False negatives**: Values predicted mistakenly as zero. **False positives**: Values predicted mistakenly as 1. **True positives**: Correctly predicted values.The values containing one can also be described in set notation as $y \setminus \hat{y} \cup \hat{y} \setminus y$. Image (b-c): Result of the final calculation of $hd_D(Y_b, \hat{Y}_b)$ and $hd_D(\hat{Y}_b, Y_b)$.

Even if reducing the HD directly sounds plausible, the definition 2.24 has some drawbacks. The authors of [129] describe some disadvantages due to the limited focus on the largest distance between the boundaries, which can lead to unstable behavior in other regions. The paper refers to other works that attempted to use the HD where most studies focus on restricted tasks such as face detection or object matching. To improve stability by focusing on the entire region of $y \setminus \hat{y} \cup \hat{y} \setminus y$, the paper proposes the following loss:

$$HL(\hat{y}, y) = \frac{1}{|\Omega|} \sum_{\Omega} ((y - \hat{y})^2 \circ (D_y^\alpha + D_{\hat{y}}^\alpha)) \tag{2.29}$$

which smoothly penalizes larger segmentation errors but takes the entire boundary into account. The parameter α determines how strongly large errors are getting penalized.

A proposed alternative for this project, where instead of raising the distance transform D_y and $D_{\hat{y}}$ to the power of the hyperparameter α and then adding both values, uses the absolute values of their sum to guarantee the necessary positive values while providing a lower weight on the right factor of the equation above, is additionally suggested after providing promising results after some initial tests. Formally, this variant is defined as

$$HL_1(\hat{y}, y) = \frac{1}{|\Omega|} \sum_{\Omega} ((y - \hat{y})^2 \circ |D_y + D_{\hat{y}}|) \tag{2.30}$$

Another disadvantage is the high computational cost for $D_{\hat{y}}(i, j)$. It cannot be precalculated as $D_y(i, j)$ as the prediction $\hat{y}$ changes during training. The paper, therefore, suggests a »*one-sided*« alternative defined as:

$$HL_{os} = (y, \hat{y}) = \frac{1}{\Omega} \sum_{\Omega} ((y - \hat{y})^2 \circ D_y^\alpha) \tag{2.31}$$

Boundary Loss (BL)

The paper »*Boundary loss for highly unbalanced segmentation*«[133] introduces a new BLs. The general goal

of the paper is to face the problems of unbalanced segmentation, which is relatively common in medical image analysis. The suggested boundary loss is supposed to provide additional information complementary to regional losses. The authors claim that the main challenge is building a differentiable function considering the regions between Y_b and $\hat{Y}_b$. Those regions are equivalent to $y \setminus \hat{y} \cup \hat{y} \setminus y$, which consists of the union of false negatives and false positives. From now on, we define $\Delta S = y \setminus \hat{y} \cup \hat{y} \setminus y$, which is equivalent to the union of the dark violet and light cyan regions in Fig. 2.11.

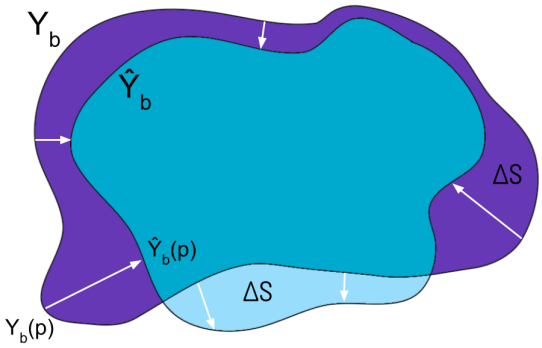


Fig. 2.14. Schematic representation of the normals indicated as white arrows perpendicular to Y_b at $Y_b(p)$ and its intersection at $\hat{Y}_b(p)$.

The loss proposed in [133] is inspired by graph-based optimization techniques [31] and attempts to approach

$$Dist(Y_b, \hat{Y}_b) = \int_{Y_b} ||Y_b(p) - \hat{Y}_b(p)||^2 dp \tag{2.32}$$

with

$$Dist(Y_b, \hat{Y}_b) \approx 2 \int_{\Delta S} D_y(q) dq \tag{2.33}$$

where $Y_b(p)$ and $\hat{Y}_b(p)$ denote two corresponding points on the boundary connected by the normal to Y_b at p and its intersection $\hat{Y}_b(p)$. See Fig. 2.14. As the expression 2.32 cannot be directly used for a loss due to local differential computations of the normal [133], it can be approximated using the distance transform $D_y(q)$ where q refers to any pixel on the ground truth y. The distance transforms $D_y(q)$, therefore, calculates the distance from q to the closest point on the boundary Y_b.

The final loss is then calculated based on a level set representation [2] $\varphi_y(q)$ which is calculated from the distance transform $D_y(q)$ as:

$$\varphi_y(q) \begin{cases} -D_y(q) & \text{if } q \in y \\ D_y(q) & \text{otherwise} \end{cases} \tag{2.34}$$

[2] A level set representation is a method of representing objects and boundaries in a computer simulation or image processing using a level set function. The level set function assigns a numerical value to each point in a given space, such as an image or a three-dimensional model. Points inside or on the boundary have one value, while points outside the object have another. The points where the value of the function changes from one to another are called level sets. They are the contours of the object or the boundary.

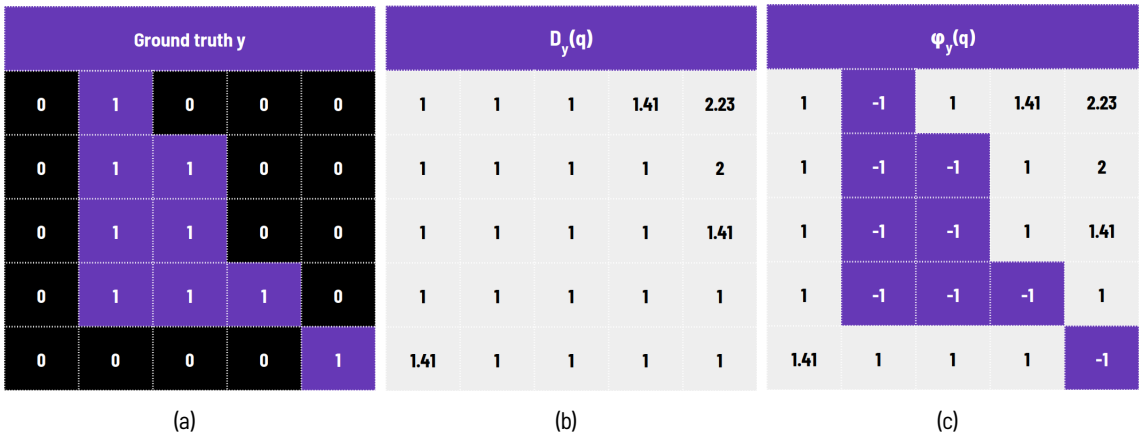


Fig. 2.15. Example calculation of $\varphi_y(q)$ (c) from y (a) with equation 2.34. Image (b) depicts the distance transform $D_y(q)$.

which then leads to

$$Dist(Y_b, \hat{Y}_b) = \int \varphi_y(q)\hat{y}dq - \int \varphi_y(q)ydq = BL \tag{2.35}$$

which has the major advantage that the level set function $\varphi_y(q)$ can be precomputed from the ground truth as it does not change during training.

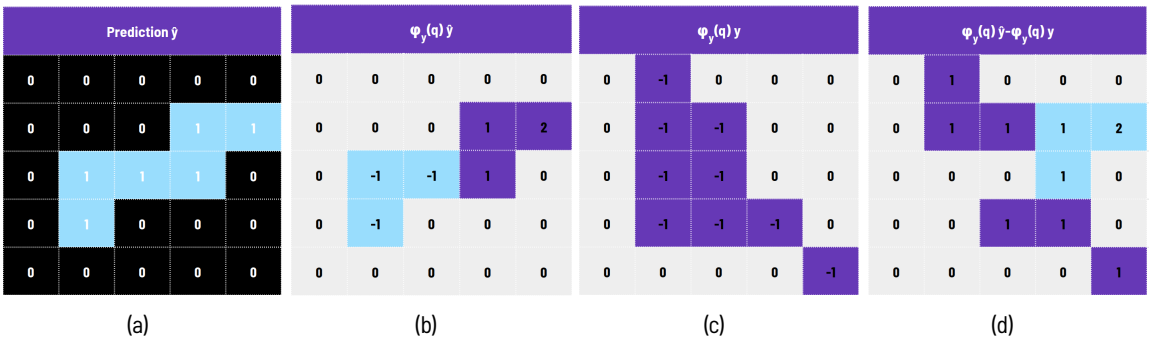


Fig. 2.16. Final calculation of BL by precomputing the level set representation $\varphi_y(q)$ from the ground truth y and multiplying it with the prediction $\hat{y}$ (b) and with the ground truth y (c). To obtain the final boundary loss, the integrals $\int \varphi_y(q)\hat{y}dq$ and $\int \varphi_y(q)ydq$ are subtracted from each other (d).

2.3.3. Segmentation Metrics

Evaluating the performance of semantic segmentation models is critical to developing and optimizing practical algorithms for image understanding tasks. To assess the quality of a model's predictions, it is essential to utilize an appropriate measure to quantify the performance of the segmentation results. In this section,

a variety of segmentation metrics commonly employed in classification tasks are explored. These metrics are the basis for comparing different models, understanding their strengths and weaknesses, and guiding further improvements. The following subsections will discuss each metric's underlying concepts, mathematical formulations, and practical applications, focusing on their relevance to semantic segmentation tasks.

Confusion Matrix

Several of the following metrics use parts of the confusion matrix, a table used to describe the performance of a classification algorithm. The confusion matrix displays the number of correct and incorrect predictions made by the classifier. For semantic segmentation, the performance of each $\hat{y} \in \hat{Y}$ can be represented as a confusion matrix that summarizes all pixels within $\hat{y}$ as correct or incorrectly labeled. In the following, the rows of the confusion matrix represent the predicted class labels, while the columns represent the true (actual) class labels. The first cell in the first row represents the count of correctly classified as positive predictions. In contrast, the second cell of the first row indicates the count of incorrectly classified as positive. The first cell in the second row represents the count of predictions incorrectly classified as negative, while the second cell correctly classified samples as negative. Even if confusion matrices exist for multi-class classification tasks, this work examines the binary case, which can be extended for multi-class classification. The confusion matrix for a binary classification problem has the following structure:

	Actual Positive	Actual Negative
Predicted Positive	True Positive	False Positve
Predicted Negative	False Negative	True Negative

Tab. 2.3. Structure of binary confusion matrix where the rows represent the predicted class labels while the cells represent the actual class. Each cell in the matrix represents the count of predictions where the true class is the column label, and the predicted class is the row label.

Related to semantic segmentation, the values within the cells represent numbers of pixels described as follows:

- True Positive (TP): Number of correctly predicted pixels of the positive class
- False Positive (FP): Number of incorrectly predicted pixels of the positive class
- False Negative (FN): Number of incorrectly predicted pixels of the negative class
- True Negative (TN): Number of correctly predicted pixels of the negative class

(a) Output label (b) Color coded prediction

Fig. 2.17. Output label (a) and example of a 25-pixel visualization where each pixel is color-coded to indicate a **True Positive**, **False Positive**, **False Negative** and **True Negative** prediction (b) resulting in a confusion matrix of $\begin{bmatrix} 5 & 5 \\ 4 & 11 \end{bmatrix}$.

The following sections will describe some common metrics derived from the confusion matrix.

Precision (PPV), Recall (TPR)

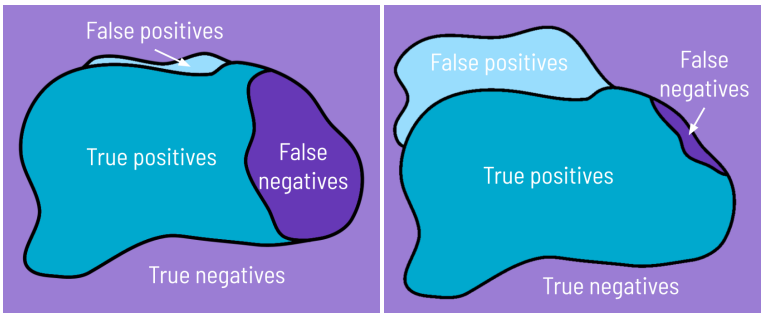
Precision and recall, also known as Positive Predictive Value (PPV) and True Positive Rate (TPR) are complementary metrics used to evaluate the performance of a classification task. The difference is that precision focuses on the number of false positives in addition to the true positives, while recall focuses on false negatives additionally to true positives.

The formal definition of precision and recall is as follows:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.36}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.37}$$

Intuitively, the perfect precision would have no false positives leading to a score of 1, while the perfect recall would have no false negatives, also leading to a score of 1. It is important to note that precision and recall are particularly relevant for the positive or foreground class in the context of semantic segmentation. Fig. 2.18 illustrates two cases: one with high precision and low recall and the other with low precision and high recall. Based on the specific requirements and priorities of the segmentation task, a user might prefer one case over the other.



(a) High precision low recall (b) Low precision high recall

Fig. 2.18. Image (a) depicts a situation with very high precision. The false positives are very low. False negatives are high, which indicates a low recall. The situation in image (b) is the opposite, with a low precision and many false positives and a high recall with few false negatives.

Specificity (TNR) and Negative Predictive Value (NPV)

Specificity, also known as True Negative Rate (TNR) and Negative predictive value (NPV), are complementary metrics akin to precision and recall. However, these metrics emphasize true negatives instead of focusing on true positives. Specificity and NPV are particularly useful in scenarios where the accurate identification of negative or background instances is crucial, such as in medical diagnostics [194] or anomaly detection [108].

Specificity and negative predictive value are formally defined as follows:

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{2.38}$$

$$\text{Negative predictive value} = \frac{TN}{TN + FN} \tag{2.39}$$

Intuitively, a perfect specificity score indicates no false positives, while a perfect NPV implies no false negatives. In semantic segmentation, these metrics highlight the model's performance in correctly identifying negative or background classes.

Intersection over Union (IoU)

The Intersection over Union (IoU), also known as the Jaccard index (JI), is a similarity measure used to compare two sample data objects. It is defined as the ratio of the intersection size to the union size [271].

Originally, the IoU was described in set notation as:

$$IOU = \frac{|A \cap B|}{|A \cup B|} = \frac{|A \cap B|}{|A| + |B| - |A \cap B|} \tag{2.40}$$

where A and B represent two sets being compared.

In the context of confusion matrix components, the IoU is defined as:

$$IOU = \frac{TP}{TP + FP + FN} \tag{2.41}$$

Dice Similarity Coefficient (DSC)

The DSC, also known as the Søerensen-Dice index (SDI), was originally defined in set notation to describe the similarity between two samples. It is a widely used measure for evaluating image segmentation algorithms, particularly within the medical field [43]. Given two sets, A and B, the DSC is defined as

$$DSC = \frac{2|A \cap B|}{|A| + |B|} \tag{2.42}$$

In terms of true positives (TP), false positives (FP), and false negatives (FN), the DSC can be expressed as

$$DSC = \frac{2 \cdot TP}{2 \cdot TP + FP + FN} \tag{2.43}$$

Notably, true positives are counted twice in this measure, resulting in the DSC having a higher value for the same number of correctly classified pixels than the IoU metric.

Accuracy (Acc)

Accuracy measures the proportion of correctly classified pixels in an image defined as

$$\text{Accuracy} = \frac{\text{Number of correctly classified pixels}}{\text{Total number of pixels}} \tag{2.44}$$

where the number of correctly classified pixels also includes True Negative predictions. In terms of confusion matrix components, this metric can also be defined as

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.45}$$

Multiclass Classification

Previously, we discussed metric calculations for binary classification tasks involving only two classes. This concept can be extended to multi-class classification problems, where the output labels y can be transformed from integers to one-hot-encoded format, as demonstrated in Fig. 2.1. With the predictions $\hat{y}$ provided in one-hot-encoded format, as shown in Fig. B.1, each class can be treated as a binary classification problem. To obtain a final metric score, we can calculate individual confusion matrices for each class and average the results. This approach is known as `macro averaging` and will be employed for this project.

An alternative method, called `micro averaging`, involves calculating a single confusion matrix for all classes and using it to compute a single metric score. The following equations demonstrate the difference between these approaches for the Intersection over Union (IoU) metric.

$$\text{Micro IoU} = \frac{\sum_{i=1}^{D} TP_i}{\sum_{i=1}^{D} (TP_i + FP_i + FN_i)} \tag{2.46}$$

$$\text{Macro IoU} = \frac{1}{D} \sum_{i=1}^{D} \frac{TP_i}{(TP_i + FP_i + FN_i)} \tag{2.47}$$

Here, D represents the number of classes, while TP_i, FP_i, and FN_i denote the true positives, false positives, and false negatives for class i, respectively. Fig. 2.19 further explores the differences between these approaches using an example that demonstrates metric calculation in an imbalanced situation.

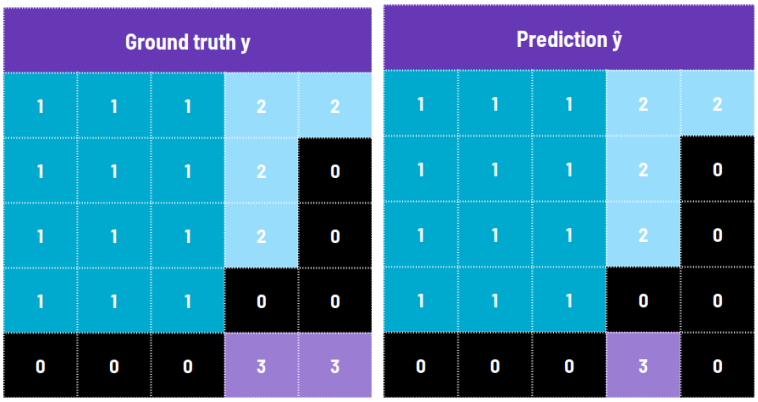


Fig. 2.19. The IoU using the micro average yields a value of 0.923, whereas the macro IoU average results in 0.843. This significantly lower value for the 'macro' average primarily stems from the false negative prediction in pixel $\hat{Y}_{5,5}$, which leads to an IoU score of 0.5 for class three. According to equation 2.47, this prediction has a much larger impact on the overall result than in the micro IoU, where the most prevalent class absorbs mispredictions for smaller classes. The prevalent class could be either the background or a foreground class with a considerably higher pixel count on average.

2.3.4. Segmentation Architectures

The architecture employed in this project is derived from the U-Net, which is based on the Encoder-Decoder Architecture (EDA). Introduced by [209] and illustrated in Fig. 2.20, the U-Net comprises two distinct paths: an encoding (contracting) path and a decoding (expanding) path, each composed of five layers. The encoding path incorporates pooling and convolutional operations along batch normalization and ReLU activation. It is responsible for capturing the context and reducing the spatial dimensions of the feature maps while increasing the number of channels. The decoding path upsamples the feature maps and concatenates them with the corresponding feature maps from the encoding path to provide high-resolution spatial information. After concatenation, the convolutions are applied to refine the feature maps. In this example, the final layer is a 1x1 convolution that maps the feature maps to the desired number of output classes resulting in a binary segmentation mask of shape 2x256x256 where each channel corresponds to one of the two classes in the binary segmentation.[209]

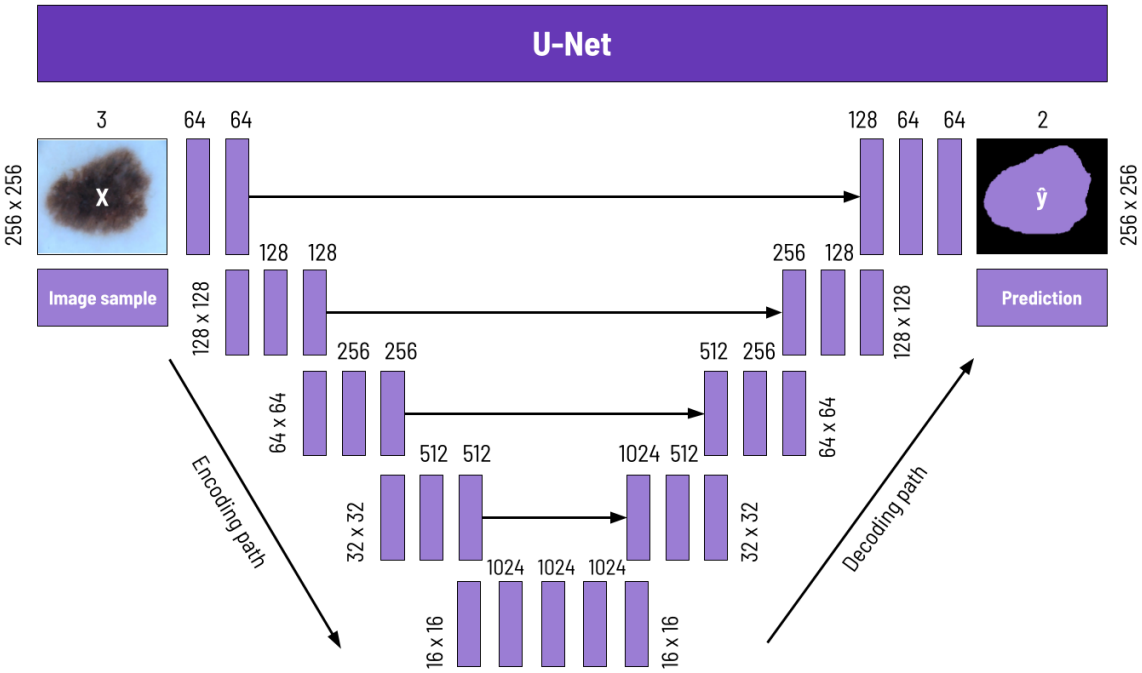


Fig. 2.20. U-Net model architecture proposed by [209]

Besides the U-Net architecture, there is a wide variety of further models. [264] provides an excellent survey on deep learning architectures for semantic segmentation such as the Spatial Pyramid Pooling (SPP) [264] or the DeepLabv3+ Architecture[52].

2.3.5. Segmentation Framework

A segmentation framework follows a similar implementation as a standard deep learning framework presented as a model production cycle in Fig. 2.21, which entails configuration, preprocessing, training, valida-

tion, testing, and monitoring.

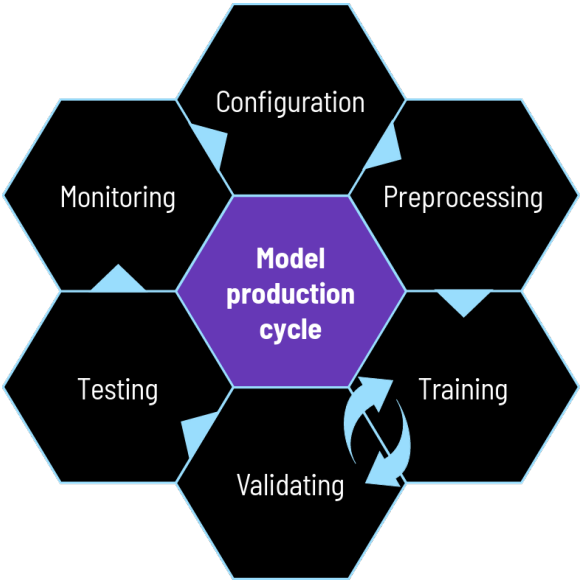


Fig. 2.21. The model production cycle iteratively preprocesses, trains, validates, tests, and logs the results of each model using a specified configuration.

1. ***Configuration***: The configuration generally entails setting up the parameters used to train the model. There are various ways to efficiently search for hyperparameters, with the selection ultimately depending on several factors such as dataset size, computational resources, and chosen network architecture.

2. ***Preprocessing***: The preprocessing step involves preparing the data for input into the model, which includes tasks such as data augmentation, normalization, and encoding of categorical features. This step ensures that the data is in a suitable training, validation, and testing format.

3. ***Training***: The training process begins by iterating through all samples in the training set. For each input sample x, the model generates a prediction $\hat{y}$ and calculates a loss $l(\hat{y}, y)$, which is used to adjust the models' weights via backpropagation which typically involves the optimization technique called gradient descent as discussed earlier. An optimizer such as the Adam optimizer can effectively determine an appropriate learning rate, also called step size for gradient descent, as it utilizes an adaptive learning rate for each parameter during the backpropagation process. Once this process is done for every sample, label pair within the training set, the model will be validated in the next step.

4. ***Validation***: After completing the training steps, the validation step is carried out to assess the trained models' performance by generating predictions and testing them against the corresponding output labels from the validation set. No model parameter updates occur during this step. Once the validation steps are finished, the process begins anew with the training set. This cycle repeats until all epochs have been completed. Epochs refer to a single pass through the entire training dataset while

training a model.

5. ***Testing***: Finally, the testing step is performed on the test data, which has been previously separated from the entire dataset. The test data results are crucial for evaluating a model's performance, as they reveal its generalization capabilities on unseen data. Like the validation step, the model parameters remain unchanged during the test step.

6. ***Monitoring***: Monitoring is a crucial part of the model production cycle as it helps track the progress of the training, validation, and testing processes, allowing for the detection of potential issues or improvements. This step includes logging performance metrics, visualizing model performance, and saving intermediate results for later analysis.

3. Literature Review

The literature review presented in this chapter focuses specifically on the topic of custom loss functions, which is directly aligned with the primary focus of this thesis. More specifically, this chapter provides a comprehensive overview of some of the most recent publications in this area, focusing on studies that, like this one, advocate for the use of loss function combinations or loss adjustments, to create a more precise learning signal than what standard loss functions can offer. While there is a wide range of literature available on loss functions and strategies for their enhancement, to the best of the authors' knowledge, no unified framework exists that systematically combines a predetermined set of prominent loss types in the manner suggested in this work.

For a more general review of semantic segmentation improvement strategies beyond the scope of this thesis, please refer to Appendix D: »*Extended Literature Review*«.

»*Relax and focus on brain tumor segmentation*« [280] from 2022, presents a CNN for fully automatic brain tumor segmentation. The authors of the paper acknowledge that one of the major challenges to accurately segmenting brain tumors is the great variation in location, size, structure, and morphological appearance, as well as the severe data imbalance that exist between brain tumors and non-tumor tissue, as well as among the different sub-regions within the brain tumor itself.

To address these challenges, the authors propose a hybrid model which incorporates several key components:

- A dynamic combination of the FL and DL specifically designed to focus more on the smaller tumor sub-regions with more complex structures during the training process.
- A relaxation of boundary constraints for the inner tumor regions in a coarse-to-fine fashion to better recognize the overall structure of the brain tumor and the morphological relationship among different tumor sub-regions.
- A symmetric attention branch highlights the possible location of the brain tumor from the asymmetric features caused by the growth and expansion of abnormal brain tissues.

The authors of the paper claim that the proposed hybrid model can balance the learning capacity of the model between spatial details and high-level morphological features and can improve the overall segmentation performance of brain tumors. The model is validated on the publicly available brain tumor dataset from real patients called BRATS 2019 [177]. The experimental results show that their model improves the overall segmentation performance compared to the top approaches in the BRATS 2019 challenge.

»*PolyLoss: A Polynomial Expansion Perspective of Classification Loss Functions*« [147] from 2022, presents a new framework named PolyLoss whose goal is to understand better and design loss functions for Deep Neural Networks (DNNs). The authors propose that commonly used classification loss functions such as CE and FL can be decomposed into a series of weighted polynomial functions. This framework allows for the importance of different polynomial bases to be adjusted depending on the target task and dataset. It demonstrates that better results can be achieved by adjusting these weights than with the commonly used loss functions. They also propose a simple and effective Poly-1 loss formulation that outperforms CE and FL on various tasks by only introducing one extra hyperparameter and adding one line of code.

»*Distribution-Aware Margin Calibration for Semantic Segmentation in Images*« [296] from 2022, describes a method to improve the performance of semantic segmentation models. The authors propose using a »*margin calibration*« method, which can be used as a learning objective to improve the generalization of the IoU metric over the data distribution. The method theoretically ensures better segmentation performance in terms of IoU. It has been evaluated on seven image datasets, showing substantial improvements in the IoU score over other learning objectives using deep segmentation models. In contrast, traditional loss functions, such as CE, tend to have a poor indicator of model quality in semantic segmentation because the minimization of it does not guarantee a higher IoU score, which is more commonly used in semantic segmentation, specifically for sketching the contours of interest regions.

»*blob loss: instance imbalance aware loss functions for semantic segmentation*« [139] from 2022, presents a new loss function called »*blob loss*« for semantic segmentation tasks, which aims to improve instance-level detection metrics, such as F1 score and sensitivity.[1] The authors argue that traditional loss functions, such as the DL, do not take instance imbalance[2] within a class into account, which can lead to sub-optimal results when the object of interest is highly heterogeneous. This is particularly relevant in medical applications, such as disease progression monitoring, where detecting small-scale lesions is critical. The proposed method is based on converting any loss function into a »*blob loss*« by adding a term that represents the instance imbalance. This term is calculated as the ratio of the number of pixels belonging to the smaller instances to the number of pixels belonging to the more significant instances. The »*blob loss*« is designed to work with semantic segmentation problems where the instances are connected components within a class.

The proposed method is evaluated on five complex 3D semantic segmentation tasks that feature pronounced instance heterogeneity in terms of texture and morphology. These tasks include multiple sclerosis lesions, liver tumors, and microscopy segmentation. The evaluation results showed that compared to the traditional soft DL, the »*blob loss*« achieved an average 2% improvement for microscopy segmentation tasks considering F1 score and 5% improvement for MS lesions.

The paper argues that the proposed »*blob loss*« method addresses the issue of instance imbalance in semantic segmentation tasks, which is not adequately reflected in traditional loss functions such as DL, and it

[1] $F1 = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$

[2] Instance imbalance within a class occurs when there is a disproportionate number of instances with specific characteristics compared to others. Characteristics in this context could be different textures or morphological differences like size or shape [139]. For example, in a forest image dataset, there might be many trees with leaves but only a few without. Similarly, there could be significant and small tumors of the same class in medical imaging. This imbalance can lead to poor overall performance as models may struggle to identify less common instances accurately.

can improve the performance of the model in detecting small scale lesions in medical applications.

»*Topology-Aware Segmentation Using Discrete Morse Theory*« [114] from 2021 proposes a new loss function incorporating topological information from the image segmentation task. The loss function is based on the concept of DMT, a mathematical framework that uses the gradient vector field of the likelihood function to identify critical points and structures in the image. These structures, composed of 1D and 2D manifold pieces, reveal the topological information captured by the likelihood function. For more detailed information on DMT, see Fig. 3.1.

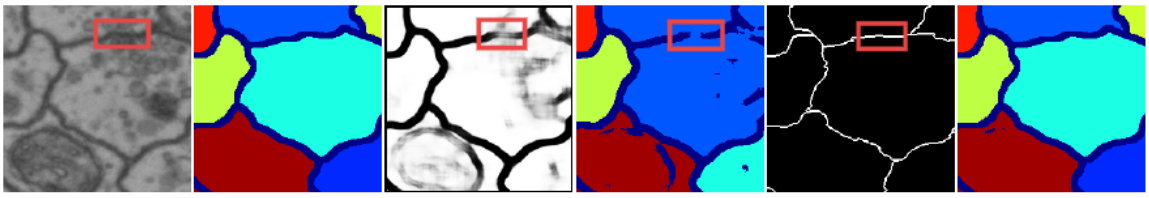


Fig. 3.1. From left to right: Sample, Ground truth, Likelihood map, Baseline prediction, Critical structure with DMT loss applied and final result. Image from [114]

The method identifies these critical structures, called Morse structures, as essential for topological accuracy. The loss function assigns a higher penalty to predictions that deviate from these structures. This way, the network is encouraged to produce predictions more consistent with the topological information.

The loss function is designed to correct false negatives, where a structure is a tree structure missed in the segmentation, and false positives, where a structure is a »*halluzination*« of the model and should be removed. By enforcing higher penalties along the identified Morse structures, the network is forced to predict correctly near these topologically difficult locations and address the sampling bias issue [3].

In summary, the new loss function proposed in the paper uses the concept of discrete Morse theory to identify global structures and critical points in the image. It incorporates this topological information into the training process by assigning a higher penalty to predictions that deviate from these structures. This approach improves the network's performance, especially near topologically challenging locations, and ensures that the network can produce segmentations with correct topology.

»*Tilted Cross Entropy (TCE): Promoting Fairness in Semantic Segmentation*« [248] from 2021, proposes a new loss function called Tilted cross-entropy (TCE) for semantic segmentation in order to minimize performance disparities among target classes and promote fairness. The TCE is inspired by the recently introduced Tilted empirical risk minimization (TERM)[154] and is adapted to the semantic segmentation setting. The authors demonstrate through quantitative and qualitative performance analyses that the proposed TCE can improve the low-performing classes of CityScapes [60] and ADE20k [312] datasets and also results in improved overall fairness.

»*Rethinking the Dice Loss for Deep Learning Lesion Segmentation in Medical Images*« [308] from 2021

[3] Sampling bias is a problem that occurs when the sample used in a study or survey is not representative of the population being studied, leading to inaccurate or misleading results [176].

presents four limitations called *Distance insensitivity*, *Boundary insensitivity*, *Region-size sensitivity* and *Insensitivity to FP / FN weight*, for the DL and how to overcome them. This section provides a detailed review of these limitations.

One significant weakness is the »*insensitivity to the distance*« between non-overlapping regions, as highlighted by [308]. This means that regardless of how far these regions are apart, the loss will remain constant even if, intuitively, a closer region should yield a lower loss. A computational example using the DL is illustrated in Fig. 3.2.

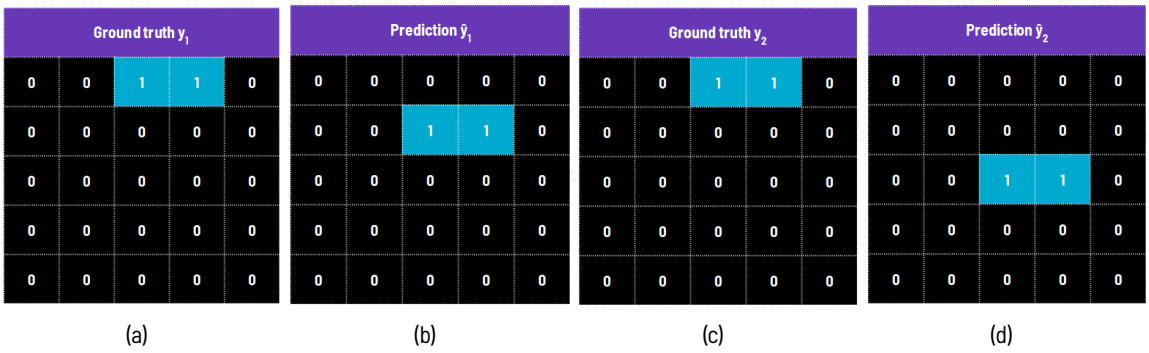


Fig. 3.2. The images show the insensitivity to distance of the DL. Image (a) and (c) depict the ground truths y_1, y_2, and image (b) and (d) the corresponding predictions $\hat{y}_1, \hat{y}_2$. It can be seen that even if $\hat{y}_1$ is closer to the ground truth than $\hat{y}_2$ the result for both predictions is the same with $DL_1 = DL_2 = 1 - \frac{2 \cdot 0}{2 \cdot 0 + 2 + 2} = 1.0$.

An additional limitation of the DL is its »*insensitivity to boundaries*«. The loss function cannot accurately account for the precise contours of segmented regions and fails to reflect how the overlap occurs if the number of true positives stays the same. See Fig. 3.3 for a detailed illustration of the issue.

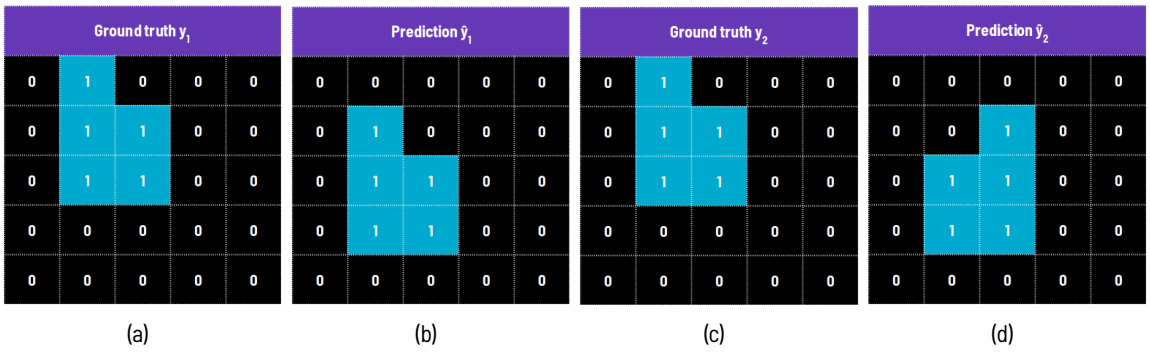


Fig. 3.3. The images demonstrate the boundary insensitivity of the DL. Image (a) and (c) depict the ground truths y_1, y_2, and image (b) and (d) the corresponding predictions $\hat{y}_1, \hat{y}_2$. Despite their visual differences, both overlaps yield an identical loss with $DL_1 = DL_2 = 1 - \frac{2 \cdot 3}{2 \cdot 3 + 2 + 3} = 0.5$.

Another notable limitation of the DL is its »*sensitivity to region size*«. Consider a scenario where a large

region needs to be predicted. If a single pixel is misclassified, the resulting loss is much lower than if a pixel is misclassified when predicting smaller regions. See Fig. 3.4 for further illustration.

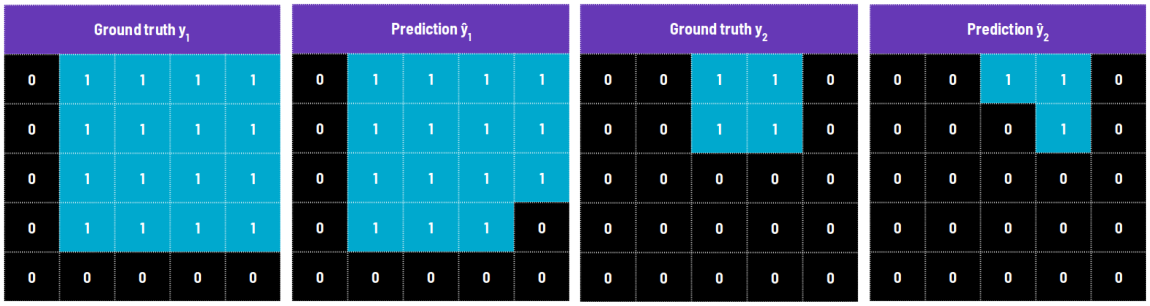


Fig. 3.4. The images demonstrate the sensitivity towards small regions. In images (a) and (b), the ground truth y and prediction $\hat{y}$ have a misclassification of one pixel, resulting in a DL of $DL_1 = 1 - \frac{2 \cdot 15}{2 \cdot 15 + 0 + 1} \approx 0.03$. On the other hand, in the image (c) and (d), the misclassification occurs on a smaller region resulting in a higher DL of $DL_2 = 1 - \frac{2 \cdot 3}{2 \cdot 3 + 0 + 1} \approx 0.142$

A further limitation of the DL, described in [308], is its »*insensitivity to FP / FN weight*«, or which can be a disadvantage since, depending on the task, researchers want to weigh false positives or false negatives differently. To illustrate this limitation, consider the example shown in Fig. 3.5.

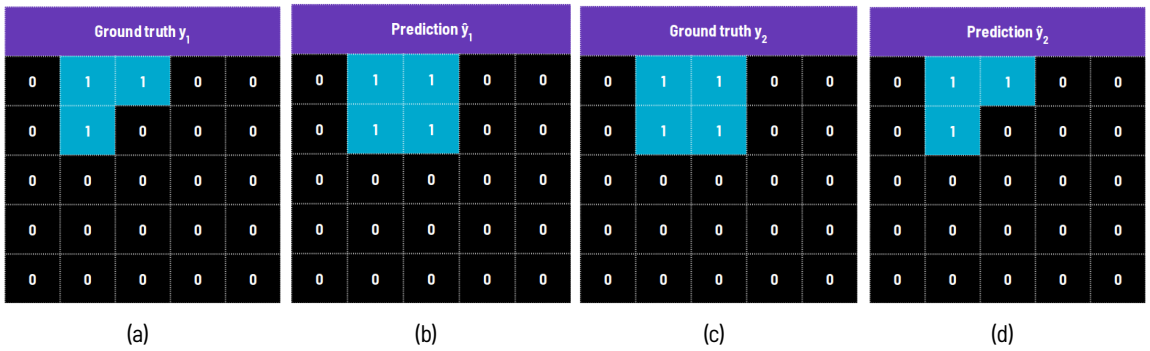


Fig. 3.5. The concept of insensitivity to the weight put on false negative and false positive predictions is demonstrated in this figure. The loss from (a) and (b) $DL_1 = 1 - \frac{2 * 3}{2 * 3 + 1 + 0} \approx 0.142$ is the same as the loss from (c) and (d) $DL_2 = 1 - \frac{2 * 3}{2 * 3 + 0 + 1} \approx 0.142$.

4. Methodology

This chapter outlines the methodology that offers a unique framework for users to experiment with multiple loss functions, combining them and applying several merging strategies to output a final loss. The goal is to enable users to gain insights into the most suitable combination for their specific data. The chapter is organized into two sections. The Limiting Factors section expands upon the constraints of the Dice Loss (DL) as mentioned earlier, incorporating additional limitations and summarizing them to lay the groundwork for the subsequent section. In section Loss Merging, a framework is introduced that facilitates combining and merging multiple loss functions to address the discussed limitations and enhance the performance compared to models trained with individual loss functions.

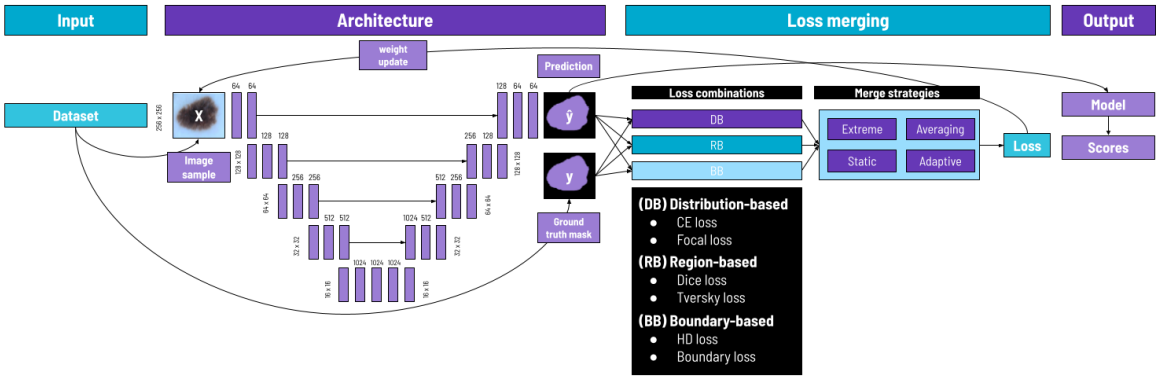


Fig. 4.1. Illustration of the proposed framework where individual losses originating from distribution, region, or boundary-based types are generated in each training iteration. These losses are combined and merged using various proposed strategies, resulting in a single, unified loss. This approach aims to enhance performance beyond that of baseline models, which are trained with just a single loss function.

4.1. Limiting Factors

This section extends the taxonomy introduced by [308] for the DL, addressing the following limitations for the six loss functions introduced in Section 2.3.2.

1. **Distance insensitivity:** As previously discussed in the context of the Dice Loss (DL), insensitivity to distances is a notable limitation for a range of loss functions. Among the functions covered in Section 2.3.2, all distribution and region-based functions exhibit this limitation. For distribution-based functions, this limitation can be demonstrated analogously to what is shown for DL in Fig. 3.2. How-

ever, this insensitivity does not constrain boundary-based loss functions that incorporate a distance transform in their calculations. See the corresponding row of table 4.1 for quantitative analysis.

2. **Boundary insensitivity:** This limitation, initially introduced in Chapter 3 for the DL, is also pertinent to distribution and region-based loss functions. The two boundary-based loss functions, namely HL and BL, appear to be somewhat resilient to this limitation, although they are not completely aware of the entire shape. In instances where two predictions are compared - with the first matching the shape of the ground truth but located farther away than a second prediction that exhibits a different shape - the latter might yield a smaller loss, thereby providing possible misleading feedback. As demonstrated in table 4.1, distribution and region-based losses yield the same result.

3. **Region-size sensitivity:** This sensitivity is associated with the issue of class imbalance and predominantly affects distribution and region-based loss functions. While initially introduced in the context of DL in Chapter 3, distribution-based loss functions are known to work best on data with equal distribution and thus are also affected by sensitivity to region size [120] [292] while boundary-based loss functions are not affected by this type of sensitivity. Table 4.1 presents a quantitative analysis where distribution and region-based losses change substantially comparing l_1 with l_2 where the former represents one misclassified pixel on large regions while the latter stems from one misclassified pixel on small regions.

4. **FP / FN weight insensitivity:** All loss functions, with the exception of the Tversky Loss (TL), which is explicitly designed to address this issue, exhibit insensitivity to the weighting of false positives and false negatives. As highlighted in Section 2.3.3, researchers may desire to manipulate high precision, low recall scenarios, or vice versa, depending on specific objectives. This manipulation can be achieved by manually adjusting the weights attributed to false positives or false negatives. On table 4.1 we can see that the TL is the only function sensitive to FP / FN weight.

5. **Outlier sensitivity:** Outliers can be described as data points unusually distant from the overall pattern and deviate significantly from other observations. Outliers may arise for various reasons, such as errors in labeling or the presence of noise. In the context of loss functions discussed in Section 2.3.2, it is essential to note that all loss functions can be sensitive to outliers. This sensitivity can disproportionately affect the model's total loss value during training. While it is generally difficult to ascertain which loss function is most affected by outliers, it is observed that boundary-based loss functions are more impacted when the mislabeled pixels are further from the region of interest. Conversely, region-based loss functions are influenced equally by pixels closer to or further from the region of interest. While many techniques exist, such as data augmentation and regularization, which can mitigate the impact of outliers and enhance the model's performance. For a more comprehensive understanding of this subject, please refer to [55], [89], [23].

6. **Computational complexity:** Boundary-based loss functions are known to be computationally expensive since they involve distance calculations between sets of points, which are much more complex and time-consuming compared to other loss functions [133]. To reduce the complexity of the HL function, the one-sided version HL_{os} can be used instead, as defined in 2.31, to reduce the effort of the distance transform $D_y(i,j)$, as defined in 2.26, for the ground truth only. Since $D_y(i,j)$ does not change during training, it can be pre-calculated for all output labels Y whenever the model is

initialized. Similarly, the BL function presented in Section 2.3.2 also requires the distance transformation to be pre-calculated, which can cause a bottleneck, especially for high-resolution images. To further reduce computational complexity, a subsampling method can use only a subset of points instead of all the pixels in the image.

7. **Manual parameter tuning:** Manual hyperparameter tuning can be a limiting factor for loss functions. Hyperparameter search can be time-consuming, especially when dealing with large datasets or complex architectures. It may also require domain expertise related to the dataset or the clinical operation, as discussed in Section 2.3.2. Additionally, hyperparameter search only sometimes guarantees to find the best solution since it relies on the intuition or experience of the user. To improve manual tuning and increase the likelihood of finding an optimal set of values, techniques such as grid search [7], random search [22], Bayesian optimization [288], or evolutionary algorithms [14] can be used. These techniques automate the hyperparameter search process and can save time and resources.

The following table 4.1 presents the quantitative results for limitations 1 through 4 confirming each statement from above. Each loss column displays two results, l_1 and l_2, which correspond to specific combinations of y and $\hat{y}$, as previously discussed in the context of Dice Loss (DL) in Chapter 3. The results that are underlined indicate scenarios that are either not affected or less affected by the corresponding limitation.

	DB				RB				BB				
	CE		FL		DL		TL		BL		HL		
	l_1	l_2	l_1	l_2	l_1	l_2	l_1	l_2	l_1	l_2	l_1	l_2	Reference
Distance Insensitivity	8.16	8.16	8.15	8.15	1.00	1.00	0.67	0.67	0.04	0.08	0.15	0.39	Fig. 3.2
Boundary insensitivity	6.36	6.36	6.35	6.35	0.40	0.40	0.33	0.33	0.04	0.04	0.15	0.18	Fig. 3.3
Region-size sensitivity	0.32	1.68	0.32	1.68	0.03	0.14	0.04	0.15	0.01	0.01	0.04	0.04	Fig. 3.4
FP / FN weight insensitivity	1.68	1.68	1.68	1.68	0.14	0.14	0.07	0.14	0.01	0.01	0.04	0.04	Fig. 3.5

Tab. 4.1. Results demonstrating limitation no. 1 to 4 for six loss functions quantitatively.

4.1.1. Summary

Table 4.2 summarizes the impact of each limiting factor on the six loss functions presented. It reveals that HL and BL are less affected by distance insensitivity, boundary insensitivity, and region-size sensitivity compared to the other loss functions. However, boundary-based loss functions have drawbacks, such as computational complexity, instability, and complex implementations.

Finding a single loss function that entirely circumvents all limitations is challenging. Nonetheless, the subsequent chapter explores a set of combinations of loss functions and merging techniques to leverage the individual strengths of each loss function to generate a more precise learning signal, thereby enhancing performance. This improvement can be important in various fields requiring a highly accurate segmentation performance, including distance measurements.

Although various implementations have attempted to address some of these limitations, such as the TL, proposed by [218], which tackles the FP/FN weight insensitivity or the GDL from [247], which is capable of

handling region-size insensitivity there is an absence of a unified framework that systematically combines and analyzes a set of loss functions. To the best of the author's knowledge, such a unified, comprehensive approach is yet to be developed.

		Limiting factors						
	Loss	Distance insensitivity	Boundary insensitivity	Region-size sensitivity	FP / FN weight insensitivity	Outlier sensitivity	Cumputational complexity	Manual parameter tuning
DB	CE	+++	+++	+++	+++	++	O(NCH*W)	-
	FL	+++	+++	+++	+++	++	O(NCH*W)	+++
RB	DL	+++	+++	++	+++	++	O(NCH*W)	-
	TL	+++	+++	++	-	++	O(NCH*W)	+++
BB	HL	-	+	-	+++	+++	O(2NCH*W	++
	BL	-	+	-	+++	+++	O(NCH*W)	++

Tab. 4.2. Summary of limiting factors. +++ strong, ++ moderate, + weak - no limitation

4.2. Loss Merging

Given the limitations of individual loss functions discussed in the previous section, this project aims to demonstrate that integrating multiple losses that capture different aspects of the segmentation problem can significantly enhance overall performance. The goal is to create a more robust and practical learning signal by leveraging each loss function's strengths and mitigating its weaknesses. Distribution-based losses, such as Focal Loss (FL), may handle challenging examples, while region-based losses can effectively address task-specific requirements or severe class imbalances. Additionally, boundary-based losses emphasize fine-grained boundaries and distances. The following subsection first describes which types and number of losses are combined and subsequently discusses how those proposed combinations are merged according to several merge strategies.

4.2.1. Loss Combinations

To facilitate a comprehensive analysis and maintain a well-organized setup, the number of loss functions is restricted to six as previously described in Section 2.3.2. This limitation allows for a systematic examination of various combinations while preserving a manageable scope for the research. The framework generally enables users to extend the number of losses to suit their specific needs. The table below displays all six functions employed to showcase the performance of the proposed method.

Types	Names
(DB)	Cross-Entropy Loss (CE), Focal Loss (FL)
(RB)	Dice Loss (DL), Tversky Loss (TL)
(BB)	Hausdorff Loss (HL), Boundary Loss (BL)

Tab. 4.3. Listing of six loss functions, grouped by type: (DB) Distribution-based, (RB) Region-based, and (BB) Boundary-based

Although combining multiple loss functions within each type (DB, RB, BB) is possible, this project proposes a combinatorial scheme that utilizes only one loss function of each type at a time. This limitation is justified by the similarity that loss functions of the same type exhibit while maintaining a manageable scope. Table 4.4 presents all twenty possible double and triple combinations, where the term 'double' refers to a strategy employing two functions. In contrast, 'triple' signifies using three functions in combination.

	(DB) Distribution-based		(RB) Region-based		(BB) Boundary-based		
No.	CE	FL	DL	TL	HL	BL	Combo type
1	✓		✓				double
2	✓			✓			double
3	✓				✓		double
4	✓					✓	double
5		✓	✓				double
6		✓		✓			double
7		✓			✓		double
8		✓				✓	double
9			✓		✓		double
10			✓			✓	double
11				✓	✓		double
12				✓		✓	double
13	✓		✓		✓		triple
14	✓		✓			✓	triple
15	✓			✓	✓		triple
16	✓			✓		✓	triple
17		✓	✓		✓		triple
18		✓	✓			✓	triple
19		✓		✓	✓		triple
20		✓		✓		✓	triple

Tab. 4.4. Table of all loss combinations without inter-type merging.

4.2.2. Merge Strategies

Following the systematic creation of a loss combination scheme with all combinations listed in table 4.4, this section further describes merging these combinations to form a single loss. Fig. 4.2 depicts the proposed strategies and their corresponding categories, such as *Extreme point*, *Averaging*, *Adaptive weighting*, and *Static weighting* strategies, which are discussed in greater detail below.

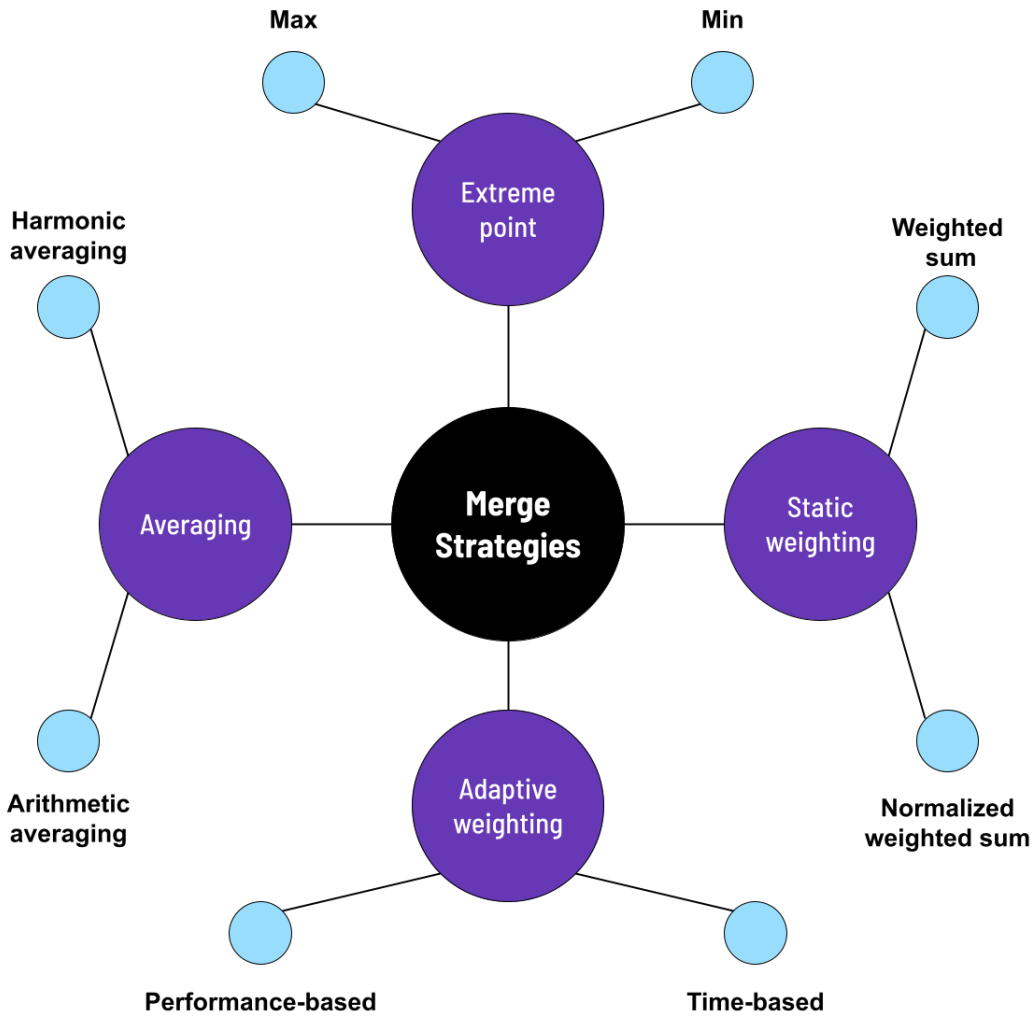


Fig. 4.2. Illustration of different loss merging strategies used to merge the different loss combinations listed in 4.4. The objective is to identify an optimal combination for a given dataset, ultimately enhancing the model's performance.

Extreme Point

This section introduces two extreme point strategies, the Max and the Min strategies, which aim to prioritize the largest or smallest value from different loss outputs.

The Max strategy (MAX) strategy calculates the loss as the maximum value among the distribution-based, region-based, and boundary-based losses. This approach aims to prioritize the most significant loss at each training step, focusing on the most challenging examples and improving overall performance. However, one disadvantage of this method is that it may focus exclusively on one loss if the range of losses is significantly different. This might lead to an unbalanced approach that fails to consider all losses during model training

appropriately. Formally the loss is defined as:

$$L_{MAX} = \max_{i=1,\dots,n} L_i \tag{4.1}$$

This equation shows that the Max strategy selects the maximum loss value, denoted as L_{MAX}, from a set of n losses, L_1 through L_n.

The Min strategy (MIN) takes the minimum value of the three loss functions at each training step and uses it for gradient updates. While this strategy can provide smooth convergence by focusing on the best-performing loss, it may overlook more significant errors in other losses and lead to suboptimal performance. It is defined as:

$$L_{MIN} = \min_{i=1,\dots,n} L_i \tag{4.2}$$

where the minimum value L_{MIN} is selected from a set of n losses, l_1 through L_n.

Averaging

Numerous mathematical techniques exist for determining the average of n data points, where, in this context, the data points represent n losses denoted as $L_1, L_2, \dots, L_n$. This study focuses on two prominent averaging strategies, which are discussed below.

Arithmetic averaging (AVG) is based on the arithmetic mean considering the average value of the individual loss functions as a final loss. This approach aims to balance the weight of each loss function and can provide a good trade-off between the best and worst-performing results. However, one disadvantage of this method is that critical information could be smoothed out, causing the model to miss essential properties of the data.

The averaging strategy is formally defined as:

$$L_{AVG} = \frac{1}{n} \sum_{i=1}^{n} L_i \tag{4.3}$$

This equation shows that the averaging strategy calculates the average arithmetic loss, denoted as L_{AA}, from a set of n losses, L_1 through L_n.

Harmonic averaging (HARMONIC) is a technique to form an average by dividing the number of data points by the sum of the reciprocals of the data points. The harmonic mean is typically used for rates and ratios such as speed, fuel efficiency, or price-to-earnings ratios. It is also applicable to merge losses specifically because it is less sensitive to outliers by down-weighting larger loss values.

The harmonic average can be formally defined as:

$$L_{HARMONIC} = \frac{n}{\sum_{i=1}^{n} \frac{1}{L_i}} \tag{4.4}$$

where L_{HA} is the final harmonic average calculated from a set of n losses, L_1 through L_n.

Static Weighting

Weighted sum (WS) merging is a technique that assigns different weights to individual loss functions and then computes the sum. While this strategy is quite flexible, allowing the user to control the importance of each loss individually, it can be challenging to find the right combination without in-depth domain knowledge.

The weighted sum strategy can be formally defined as:

$$L_{WS} = \sum_{i=1}^{n} w_i L_i \tag{4.5}$$

This equation shows that the weighted sum strategy calculates the weighted sum loss value, denoted as L_{WS}, from a set of n losses, L_1 through L_n, where w_i represents the weight assigned to each loss.

Normalized weighted sum (NWS) is a variation of the regular weighted sum strategy. The difference is that it outputs smaller values as the regular weighted sum as its weights get normalized, adding up to one. The proportional contribution of each weight is the same. One advantage of the normalized weighted sum is its simplicity, which lies in the weight contributions being limited to values between 0 and 1. The mathematical function that defines the normalized weighted sum is straightforward, making it a practical and accessible approach. The formula for the normalized weighted sum strategy is:

$$L_{NWS} = \sum_{i=1}^{n} \frac{w_i}{\sum_{j=1}^{n} w_j} L_i \tag{4.6}$$

where n is the number of loss functions, w_i is the weight of the i-th loss function, and L_i the value of the ith loss function.

Adaptive Weighting

This section describes two more strategies individually designed for this project. While the first strategy's weights are crafted around the loss performance, the second strategy calculates the weights based on temporal factors such as the current training epoch or the current number of steps. Both strategies include

several variations and hyperparameters, which can be set to define the weight's contribution gradually. The mathematical definition was inspired by the Focal Loss (FL) and the Bernoulli distribution.

Performance-based weighting (PBM) is a method for dynamically adapting weights depending on the performance of every single loss. The contributions of single losses to the optimization process for this method are formulated as follows:

$$L_{PBM} = \sum_{i=1}^{n} w_i \cdot L_i \tag{4.7}$$

Here, L_{PBM} is the performance-based loss, L_i is the i-th component of the loss, and w_i is the weight assigned to the i-th component.

The weight w_i is calculated based on the normalized loss component $\bar{L}_i$ and a hyperparameter α that controls the function's slope. The following equation gives the weight w_i:

$$w_i = (1 - I) \cdot (\bar{L}_i)^{\alpha} + I \cdot (1 - (\bar{L}_i))^{\alpha} \tag{4.8}$$

Here, I is a binary parameter determining whether an ascending or descending strategy is used. An ascending strategy assigns larger weights to larger losses, while a descending strategy assigns smaller weights to larger losses. The normalized loss component $\bar{L}_i$ is calculated as

$$\bar{L}i = \frac{L_i}{\sum j = 1^n L_j} \tag{4.9}$$

where the i-th component of the loss is divided by the sum of all loss components, and the resulting value represents the proportion of the total loss attributed to the i-th component.

The choice of α determines the slope of the weight function, as shown in Fig. 4.3. When $I = 1$, the weight function represents a descending strategy, where higher losses receive smaller weights. When $I = 0$, the weight function represents an ascending strategy, where higher losses receive larger weights. By selecting appropriate values of α, the weight function can be adjusted to achieve the desired balance between different loss components. Note that I can also be set to values between 0 and 1. If $I = 0.8$, we get a descending strategy for smaller losses and a lightly ascending strategy for larger losses.

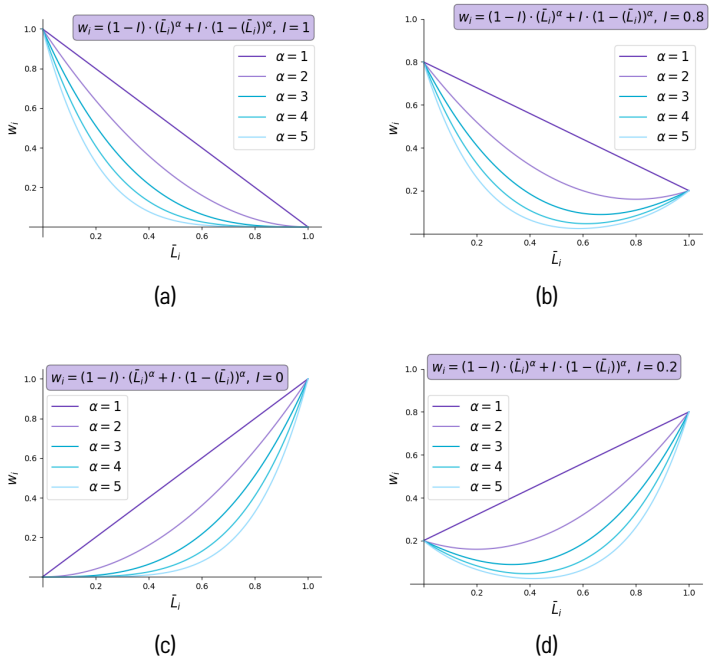


Fig. 4.3. (a) represents the descending strategy of performance-based weighting. To use this strategy, I must be set to 1. As α increases, the weight contribution for larger losses decreases, while the contribution for smaller losses increases. Image (b) combines descending strategy for smaller losses and a smooth ascending strategy for larger losses. (c) represents an ascending strategy, which can be used by setting I to 0 and using α to control w_i given a normalized loss L_i. In this strategy, higher losses receive higher weight proportions than smaller losses. Graph (d), opposite graph (b), represents a smooth descending strategy for smaller losses and an ascending strategy for larger losses.

Time-based weighting (TBM) is a strategy incorporating a time component to the loss calculation. The idea is to either gradually increase or decrease weights over time. The following method is implemented to be compatible with any other loss merging method discussed so far by simply adding a time-based weighting factor u_i to the general weighting scheme $L_i \cdot w_i$. The function can be formally described as follows:

$$L_{TBM}(t) = \sum_{i=1}^{n} w_i L_i u_i(t) \tag{4.10}$$

where w_i and L_i represent the weight and loss function, respectively, of any loss functions discussed previously, and $u_i(t)$ is defined as:

$$u_i(t) = \begin{cases} (1-I_i)\cdot\left(\frac{t}{T}\right)^\alpha + I_i\cdot\left(1-\frac{t}{T}\right)^\alpha, & i=1 \\ (1-I_i)\cdot\left(\frac{t}{T}\right)^\beta + I_i\cdot\left(1-\frac{t}{T}\right)^\beta, & i=2 \\ (1-I_i)\cdot\left(\frac{t}{T}\right)^\gamma + I_i\cdot\left(1-\frac{t}{T}\right)^\gamma, & i=3 \end{cases} \tag{4.11}$$

This formulation is very similar to the performance-based weighting scheme, which can be applied to every loss output individually. $0 \leq u_i(t) \leq 1$ represents the time-dependent weight of the i-th loss function

at step or epoch t. T denotes the total number of epochs or steps, and $\alpha \geq 1$, $\beta \geq 1$, and $\gamma \geq 1$ are parameters that control the impact on the corresponding loss functions. $I \in [0, 1]$ determines whether an increasing or decreasing time-based strategy is employed. If $I = 0$, the weight u_i will increase as training progresses, whereas if $I = 1$, u_i will gradually decrease. Similar to the performance-based weighting scheme, it is possible to define I as a continuous value between 0 and 1, changing the function's slope to the opposite direction using the same strategy. In other words, weights, during the beginning, could be gradually decreased and then automatically increased again when approaching the end of the training cycle.

Strategy type	Strategy	Hyperparameters	Definition
Extreme point	MAX	-	$\max_{i=1,\ldots,n} L_i$
	MIN	-	$\min_{i=1,\ldots,n} L_i$
Averaging	AVG	-	$\frac{1}{n}\sum_{i=1}^{n} L_i$
	HARMONIC	-	$n/\sum_{i=1}^{n} \frac{1}{L_i}$
Static weighting	WS	$w_1, w_2, \ldots w_n$	$\sum_{i=1}^{n} w_i L_i$
	NWS	$w_1, w_2, \ldots w_n$	$\sum_{i=1}^{n} w_i / \sum_{j=1}^{n} w_j L_i$
Adaptive weighting	PBM	α, I	$\sum_{i=1}^{n}((1-I)\cdot(\bar{L}_i)^{\alpha} + I \cdot (1-(\bar{L}_i))^{\alpha}) \cdot L_i$
	TBM	α, β, γ, I	$\sum_{i=1}^{n} w_i L_i u_i(t)$

Tab. 4.5. Summary of loss merging strategies including their hyperparameters

5. Experimental Setup

This chapter describes the experimental setup, introducing the datasets employed to evaluate the proposed method, followed by an outline of a distinct *Configuration* setup, which encompasses the selection and rationale behind all parameters. Subsequently, the *Preprocessing* steps are discussed, along with specific settings related to *Training*, *Validation*, and *Testing*. The *Monitoring* subsection enumerates the metrics utilized for this project and elaborates on additional techniques such as learning curves, Weights and Biases (wandb) (Weights and Biases), early stopping, and the ablation setup, which measures similarity across different subsets used for training.

The setup is designed to comprehensively assess the method's performance, scalability, and robustness. It is based on the model production cycle presented in Section 2.3.5, which systematically creates, trains, and evaluates new models using a specified configuration.

5.1. Datasets

Three datasets were used to evaluate the proposed methodology's performance. These include the Medaka Fish Dataset (MFD) [222], the Skin Lesion Dataset (SLD) from International Skin Imaging Collaboration (ISIC) [58] and the Indian Diabetic Retinopathy Image Dataset (IDRID) [199] where each of them presents unique challenges and properties which allows for a comprehensive analysis of the proposed approach.

The Medaka Fish Dataset consists of images of the Medakas fish cardiac system. The dataset contains four classes, including the background class. The images display varying complexity, as they contain multiple cellular structures and patterns. The Bulbus class, illustrated as the purple mask in Fig. 5.1 on the image (d), is the hardest class to predict as the cardiac outflow is not as clearly delineated as the Atrium and Ventricle [222].

The Skin Lesion Dataset contains dermoscopic images of skin lesions, including various types of skin cancer, such as melanoma, basal cell carcinoma, and squamous cell carcinoma. The dataset also contains benign skin lesions, such as nevi and seborrheic keratoses. This dataset aims to accurately delineate the borders of skin lesions, which is an essential part of the diagnosis and treatment planning for skin cancer [58].

The Indian Diabetic Retinopathy Image Dataset contains high-resolution retinal images from patients with diabetic retinopathy, a diabetes-related complication and leading cause of vision loss worldwide [243]. The dataset features images exhibiting varying degrees of diabetic retinopathy severity alongside standard retinal images for comparison. The primary objective of this dataset is to identify and classify different retinal

lesions, such as microaneurysms, hemorrhages, and exudates, which can offer valuable insights for the early detection and treatment of diabetic retinopathy. Due to its highly imbalanced properties and distribution of lesions across the entire retina, this dataset presents significant challenges for prediction tasks.

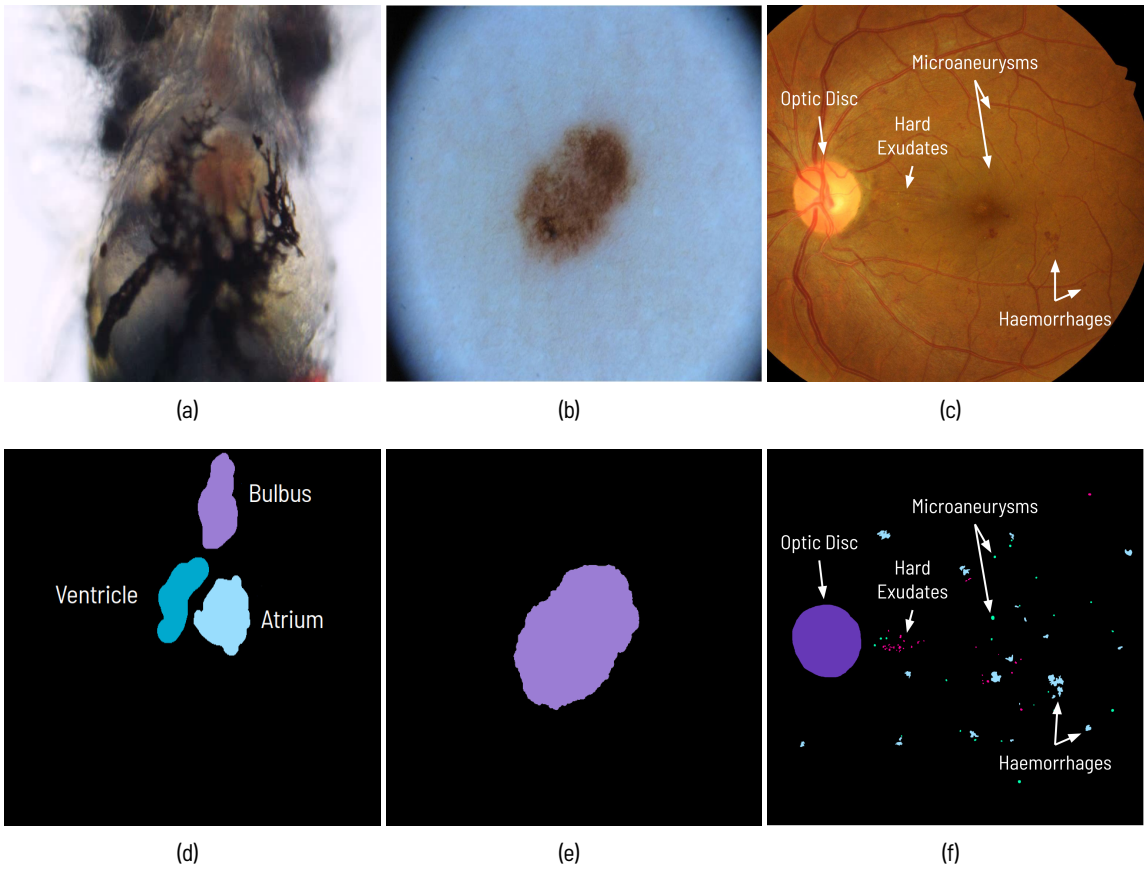


Fig. 5.1. Labels and masks of three different datasets used to evaluate this project. The image (a) is a label extracted from the Medaka Fish Dataset (MFD) covering the central vascular system (Bulbus, Atrium, and Ventricle) [222]. Image (b) displays a sample used along the International Skin Imaging Collaboration (ISIC) challenge 2017 [58], which displays a skin cancer lesion. Image (c) illustrates a sample from the Indian Diabetic Retinopathy Image Dataset (IDRID)[199] containing five classes, including the background. The second row (d-f) depicts the corresponding masks from the labels above.

		No. images				
Name	Reference	Training	Testing	No. classes	Distribution	Preprocessing
Medaka Fish	[222]	565	165	4	94.2%, 1.8%, 1.5%, 2.6%	resize,split,shuffle
Skin Lesion (ISIC)	[58]	2000	600	2	76.7%, 23.3%	resize,split,shuffle
IDRID	[199]	54	27	5	96.2%, 1.0%, 0.95%, 0.1%, 1.8%	resize,split,shuffle

Tab. 5.1. List of three datasets used to evaluate this project. The first entry of the *class distribution* column represents the background class. It can be seen that the datasets, especially Medaka and IDRID, are extremely unbalanced. The distribution for the MFD corresponds to the classes Background, **Bulbus**, **Atrium**, and **Ventricle**. For the SLD with two classes, it is the Background and the Foreground, and for the IDRID, it would be Background, **Haemorrhages**, **Hard Exudates**, **Microaneurysms**, and **Optic Disc**.

Table 5.1 offers additional details on the datasets utilized in this project, including their respective sizes, the number of classes, and the corresponding distribution of each class.

5.2. Configuration

This section briefly discusses the configuration as the first step of the model production cycle illustrated in Fig. 2.21. For a more detailed description and the rationale concerning the parameter selection, please refer to Chapter E in the appendices.

5.2.1. Parameter Definition

The configuration involves multiple parameters, categorized according to the framework component they affect. These parameters are divided into two classes: one that maintains a default value throughout the project and another with values that change during training. The changing parameters are further classified into discrete and continuous sets. It is crucial to clearly define these parameters to ensure a consistent and reproducible experimental setup, making it easier to understand the impact of each parameter on the model's performance. The appendix thoroughly analyzes these parameters, specifically in Section E.1. This comprehensive examination allows for a deeper understanding of the parameter space and its influence on the overall results. Based on the parameter analysis, a final predefined sweep configuration [1] is established, facilitating a proper and precise evaluation of the model's performance under various conditions.

5.2.2. Final Setup

The final setup for this project encompasses five distinct configurations for each dataset. The first configuration establishes a baseline by training models using the six loss functions described earlier. The second configuration employs a custom approach, systematically iterating through all possible combinations of parameter values. This is feasible because the entire set of parameters originates from a discrete space.

[1] Sweep configurations are a feature provided by wandb which enables different types of hyperparameter exploration including custom-designed training setups.

The third and fourth configurations utilize an adaptive resource allocation algorithm called Hyperband [151], specifically designed to efficiently identify optimal hyperparameters within a continuous space. It is important to note that the parameter `selection_percentage` is set to 1.0 for the IDRID dataset, as using a subset of an already tiny dataset did not seem plausible. The fifth configuration represents a particularly promising combination, which is determined based on the results of previous configurations. Its purpose is to reaffirm the outcomes through a series of repeated trainings. This approach is designed to enhance the stability and reliability of the proposed implementation.

The following table summarizes the presented configurations along their most important parameters for each dataset, resulting in 15 sweep configurations.

No.	Combo type	Selection %	Strategies	Dataset	Time based	Epoch	Model	Config type	Model count	Auto LR	Img size
1	single	0.32,0.64,1.00	-	Medaka	False	100	UNET	Baseline	18	False	256 x 256
2	double,triple	0.32,0.64,1.00	MAX,MIN,AVG,HARMONIC,WS,NWS	Medaka	False	100	UNET	Discrete	360	False	256 x 256
3	double,triple	0.32,0.64,1.00	PBM	Medaka	False	100	UNET	Continuous	-	False	256 x 256
4	double,triple	0.32,0.64,1.00	TBM	Medaka	True	100	UNET	Continuous	-	False	256 x 256
5	double, triple	0.32,0.64,1.00	Any	Medaka	False	100	UNET	Specific	18	False	256 x 256
6	single	0.32,0.64,1.00	-	Skin lesion	False	100	UNET	Baseline	18	False	256 x 256
7	double,triple	0.32,0.64,1.00	MAX,MIN,AVG,HARMONIC,WS,NWS	Skin lesion	False	100	UNET	Discrete	360	False	256 x 256
8	double,triple	0.32,0.64,1.00	PBM	Skin lesion	False	100	UNET	Continuous	-	False	256 x 256
9	double,triple	0.32,0.64,1.00	TBM	Skin lesion	True	100	UNET	Continuous	-	False	256 x 256
10	double, triple	0.32,0.64,1.00	Any	Skin lesion	False	100	UNET	Specific	18	False	256 x 256
11	single	1.00	-	IDRID	False	150	UNET	Baseline	6	True	512 x 512
12	double,triple	1.00	MAX,MIN,AVG,HARMONIC,WS,NWS	IDRID	False	150	UNET	Discrete	120	True	512 x 512
13	double,triple	1.00	PBM	IDRID	False	150	UNET	Continuous	-	True	512 x 512
14	double,triple	1.00	TBM	IDRID	True	150	UNET	Continuous	-	True	512 x 512
15	double, triple	1.00	Any	IDRID	False	150	UNET	Specific	6	True	512 x 512

Tab. 5.2. Five distinct configurations are utilized for each dataset. It is important to note that for the Indian Diabetic Retinopathy Image Dataset (IDRID), certain configuration parameters, such as `selection_percentage`, `epochs`, `auto_learning_rate`, and `image_size`, primarily differ due to its small size and the comparatively minimal additional computational effort needed for training. To ensure more dependable results, each configuration undergoes multiple training sessions, effectively preventing performance biases in extreme directions.

5.3. Preprocessing

Given that the primary objective of the research is to demonstrate the efficacy of combining loss functions, no data augmentation techniques were applied to the datasets. The preprocessing steps, on the other hand, included resizing images to a uniform resolution of (256, 256) and (512, 512) pixels, creating a validation set by splitting the training samples using an 80:20 ratio and employing random sampling during the training process from the remaining training set.

5.4. Training, Validation, and Testing

The training, validation, and testing step encompass setting up the training loop, data loading, and optimization. The first step involves initializing the model with the sweep configuration and the hyperparameters discussed in the previous sections, which ensures that the appropriate values are supplied when the training, validation, and testing loops are invoked. The Adam optimizer, which employs an adaptive learning

rate for each parameter during backpropagation, is the only additional optimization technique applied at the training stage.

5.5. Monitoring

Monitoring is a crucial part of the model production cycle as it helps track the progress of the training, validation, and testing processes, allowing for the detection of potential issues or improvements. This section discusses different monitoring techniques and tools for experimenting to ensure the creation of a fully functioning, computationally efficient setup to provide a system that can compare the trained models effectively.

5.5.1. Performance Metrics

To provide precise analysis and highlight various aspects of the results, a range of segmentation metrics has been selected to evaluate the performance of the models. It is crucial to choose the correct metric that can accurately perform, even in highly imbalanced situations, without yielding misleading outcomes. Chapter C in the appendix, offers a *Metric Comparison*, resulting in the selection of DSC and IoU as suitable choices for performance evaluation within the scope of this project. Considering the marginal differences between IoU and DSC, where DSC is typically proportionally higher, the results are exclusively presented employing IoU. Additionally, an IoU score is computed for each class to probe into the performance differences among different classes. As additional metrics, PPV and TPR are recorded and showcased, enabling the analysis of scenarios where high precision is contrasted with low recall or vice versa. Incorporating multiple metrics provides a more exhaustive comprehension of the model's performance and assists in determining areas requiring enhancement. Table 5.3 summarizes all the segmentation metrics used in this study.

Metric name	Averaging	Description
Intersection over union (IoU)	macro	Section 2.3.3
Intersection over union (IoU)	none	Separate calculation of each class
Positive predictive value (PPV)	macro	Section 2.3.3
True positive rate (TPR)	macro	Section 2.3.3

Tab. 5.3. Overview of segmentation metrics used for evaluating model performance in this project.

5.5.2. Learning Curves

By plotting learning curves for both training and validation, the progress of the models can be visualized over time. These curves and the metrics provide intuitive insight into potential overfitting or underfitting issues and the effectiveness of the loss function combinations.

5.5.3. Weights and Biases

As a first visualization tool, wandb is set up to monitor the training progress in real-time. Integrating wandb with the proposed framework allows different aspects to be observed and validated for plausibility. While this is very helpful during the prototyping stage, wandb can also help summarize the models by providing excellent features for grouping or filtering the results based on the given parameters.

5.5.4. Early Stopping

Early Stopping can prevent overfitting and reduce training time, which makes sense when dealing with larger datasets that take much longer to train. Early stopping monitors the validation loss and stops the training process when no significant improvement is observed for a predefined number of epochs, thereby ensuring the model to not continue training without making meaningful progress. This project employs Hyperband for continuous configurations setting which is an early stopping technique based on the Successive Halving algorithm [152]. For more information on this algorithm and the rationale of its selection, refer to Section E.2 in the appendix.

5.5.5. Ablation Setup

This subsection describes a method that can analyze the similarity of two result sets trained on different subsets of the data. As input, we are given the full results for a specific configuration from the table 5.2. The first step is to calculate the average IoU for every loss-merge combination and each subset of the data denoted in percent as 100, 64, and 32%. Formally, this can be defined as

$$IoU_{avg} = \frac{1}{n} \sum_{i=1}^{N} IoU_i \tag{5.1}$$

where N is the number of entries that match the combination of loss, strategy, and subset, and IoU_i is the IoU score of the i-th matching model entry. As a result, for configuration no.2 from 5.2, we would be given $T = 360$ distinct values of IoU_{avg}. The algorithm further calculates the ratios of IoU_{avg} between the subsets of 100% to 64% and between 64% to 32% of the data, denoted as $R_{1.0,0.64}$ and $R_{0.64,0.32}$ formally defined as

$$R_{1.0,0.64} = IoU_{avg}(1.0) \oslash IoU_{avg}(0.64) \tag{5.2}$$

and

$$R_{0.64,0.32} = IoU_{avg}(0.64) \oslash IoU_{avg}(0.32) \tag{5.3}$$

where $\oslash$ is the pointwise Hadamard division [285] resulting for configuration no. 2 of table 5.2 in $M = 120$[2] distinct values. The last step is to calculate the averages and standard deviation of these two ratio sets as

$$R_{avg} = \frac{1}{M} \sum_{j=1}^{M} R_j \tag{5.4}$$

[2] M and T are subdivided further in practice for distinct subsets of double and triple combinations.

and

$$R_{std} = \sqrt{\frac{1}{M}\sum_{j=1}^{M}(R_j - R_{avg})^2} \tag{5.5}$$

The mean and standard deviation provide insights into the direction of data change and the magnitude of its deviation around the mean ratio. Should the mean exceed one, it implies a decrease in performance. A low standard deviation suggests how proportional those increases or decreases are and how similar the results perform, which is advantageous as it allows us to get insights from just a subset of the data, providing information about suitable combinations.

6. Implementation

This chapter presents the implementation details of the proposed segmentation framework, as depicted in Fig. 4.1. The primary goal of this chapter is to provide a comprehensive understanding of the practical aspects of this project. The subsequent sections discuss the specific components of the proposed segmentation framework, including Programming Language, Deep Learning Framework, Experiment Management Tools, the Network Architecture and a concise overview of a unique Image Analysis Spreadsheet utilized for an in-depth analysis of loss functions at the pixel level. To guarantee reproducibility and ease of use, the entire set of libraries and dependencies can be conveniently accessed within the provided Docker image this project includes and available on Github. The repository's Readme file contains all the installation instructions for this work.

6.1. Programming Language

Python 3.9.13 was employed to implement the segmentation framework, taking advantage of its rich ecosystem ideal for machine learning applications.

For image processing, the OpenCV library [32] was used, handling tasks such as opening, resizing, and saving images. Matplotlib [118] was utilized for visualization and plotting, enabling the creation of various graphics to analyze the performance of the segmentation models.

6.2. Deep Learning Framework

This project uses PyTorch as a deep learning framework. Some advantages of PyTorch are its extensive ecosystem and strong community support, which facilitated the development and experimentation of the proposed work. PyTorch Lightning, a lightweight wrapper built on top of PyTorch, was employed to simplify the process further. The main advantage is its simplified usage, allowing researchers to focus on core aspects of their projects by automating boilerplate code with deep learning experiments [156]. The project can be launched from within a docker container to enable a fast and reliable setup.

The implementation structure follows the model production cycle presented in Section 2.3.5, which consists of a configuration, some preprocessing steps, training, validation, testing, and monitoring.

6.3. Experiment Management Tools

It is crucial for a ML pipeline to effectively manage the experiments and logging to monitor the performance of the models. This project employs wandb, a convenient experimental tracking, visualization, and management tool. wandb allows for keeping track of hyperparameters, model configurations, and results and visualizing metrics and loss curves even in real time. Such a tool ensures that the best model configurations can be easily identified and different loss combinations effectively compared. Fig. 6.1 illustrates a screenshot of the wandb graphical user interface with a mask viewing section suitable for semantic segmentation.

PyTorch Lightning's built-in loggers, CSVLogger and WandbLogger, were used to enhance logging capabilities further. The CSVLogger provides an easy way to save training, validation, and testing metrics in a CSV file to ensure a local record of the experiments. WandbLogger integrates with wandb by automatically logging the metrics, model checkpoints, and other information.

Combining a local and web-based logging capability seems convenient for managing experiments, logging essential information, and ensuring reproducibility and transparency.

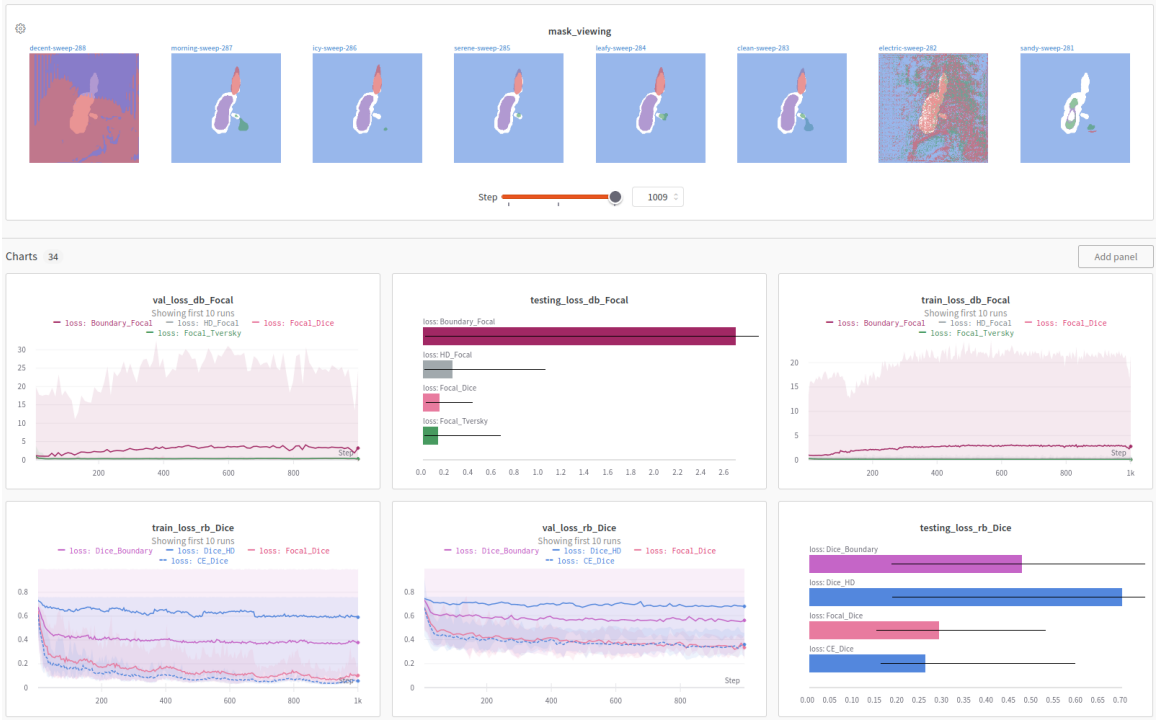


Fig. 6.1. Graphical user interface of Weights and Biases (wandb). The mask viewing section displays predicted masks against the inverted ground truth label. The charts section provides training, validation, and testing results, which can be monitored in real-time.

6.4. Network Architecture

6.4.1. U-Net Implementation

The implemented network architecture is based on the U-Net, introduced in Section 2.3.4 as part of an encoder-decoder network. The specific implementation used for this project was obtained from the GitHub repository of Boris Dayma, available at https://github.com/borisdayma/lightning-kitti and is described in the appendix in Section F.1 in more detail.

6.4.2. Integration of Loss Functions

Each loss function is structured as a separate class and imported during the model initialization. During the training process, the forward function of the corresponding loss function class is utilized to compute the loss.

For instance, the deep learning class is implemented as follows:

```
import torch
import torch.nn as nn

class DiceLoss(nn.Module):
    def __init__(self, eps=1e-6, do_bg=True) -> None:
        super(DiceLoss, self).__init__()
        self.eps = eps
        self.do_bg = do_bg

    def forward(self, y_hat: torch.Tensor, y: torch.Tensor):
        bs, cl = y.size(0), y.size(1)
        y, y_hat = y.view(bs, cl, -1), y_hat.view(bs, cl, -1)
        intersec = torch.sum(y_hat * y, dim=(0, 2))
        card = torch.sum(y_hat + y, dim=(0, 2))
        diceScore = (2 * intersec + self.eps) / (card + self.eps)
        if(self.do_bg==False):
          dice_score=dice_score[1:, ...]
        return (1. - dice_score).mean()

```

Listing 6.1 Implementation of the Dice Loss function for semantic segmentation tasks, inheriting from PyTorch's `nn.Module` class. The function calculates the loss based on predicted and ground truth tensors. The tensors are reshaped, and the intersection and cardinality of the predictions and ground truth are calculated. The dice score is computed, and the loss is returned as the mean of 1-DSC. The implementation allows including or excluding the background class in the loss calculation, providing flexibility for various segmentation scenarios. The required dependencies are `torch` and `torch.nn`.

Other loss functions in this work follow a similar structure to the deep learning class, ensuring a consistent and modular approach to handling various loss functions during model training.

6.5. Image Analysis Spreadsheet

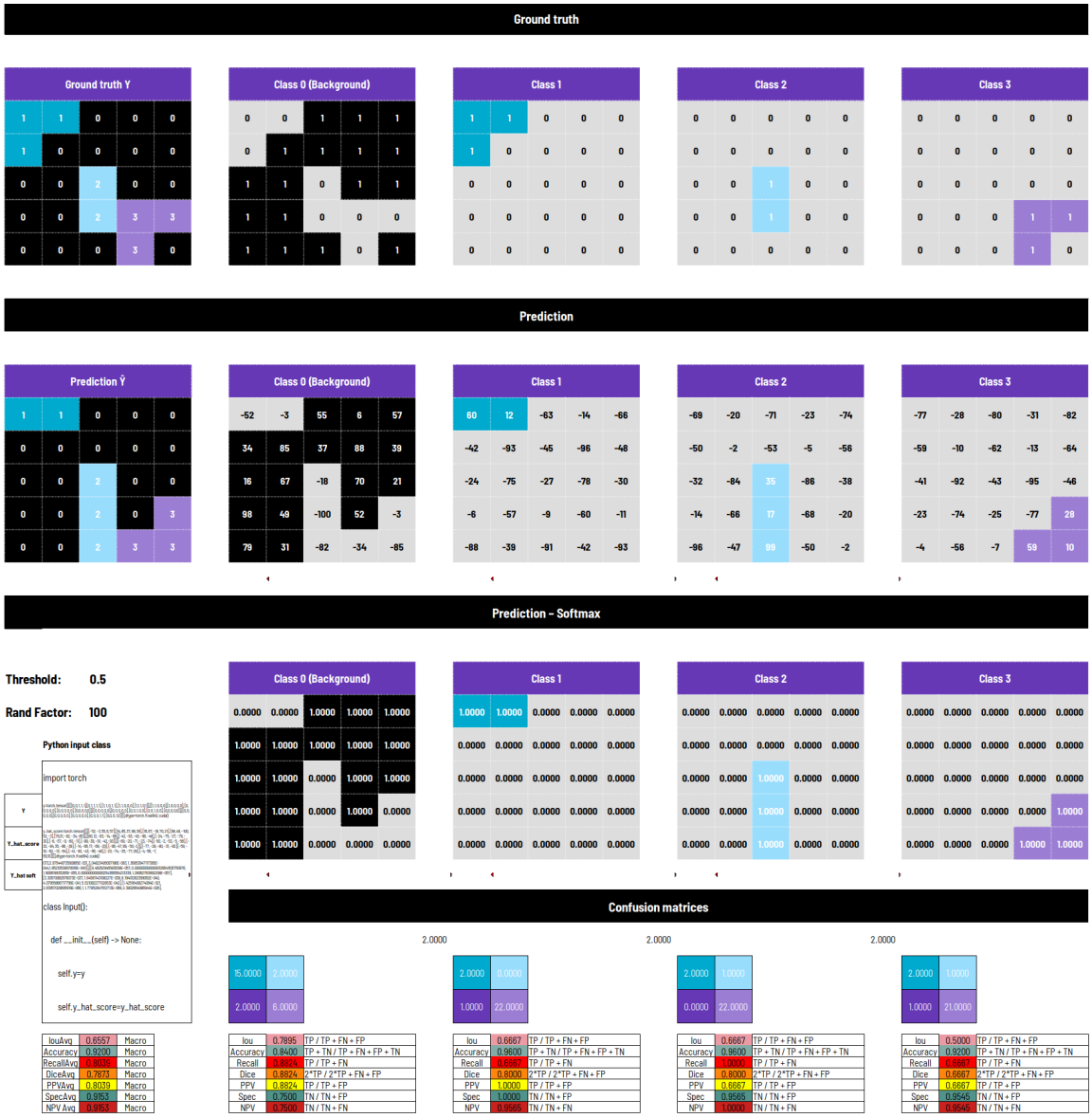


Fig. 6.2. The screenshot showcases a set of (5,5) sized images. The top row displays the ground truth, the second row presents the corresponding prediction scores, and the third row illustrates the application of the softmax function on the predictions as defined in Section B.1. Beneath these toy images, the corresponding metrics are calculated based on the confusion matrices.

In addition to the standard libraries and frameworks, a unique working environment using LibreOffice Calc was set up. This allowed for a more in-depth analysis of loss functions at the pixel level. An environment for small images of size (5,5) and (10,10) with up to four classes were created for ground truth labels, prediction scores, and softmax probabilities. The spreadsheet further calculates the confusion matrix of each class

and the seven metrics introduced in Section 2.3.3. The Calc spreadsheet organized the images to visualize and manipulate the data quickly. The spreadsheet was further connected to a Python script, which interfaced with the deep learning framework to compute and analyze the various losses. This custom setup allowed to gain deeper insights into the behavior and effectiveness of different loss functions and metrics on a small scale. It helped to understand their impact on semantic segmentation performance better.

7. Results and Analysis

This chapter presents the results and analyses of experiments designed to evaluate the effectiveness of the proposed methodology, which combines different loss functions for semantic segmentation tasks. The objective is to gain insights into this approach's performance and potential advantages compared to baseline models.

The chapter is organized into several sections, presenting the results from distinct perspectives. It begins with *quantitative* and *qualitative results*, offering an in-depth analysis of the proposed method's performance across multiple datasets and evaluation metrics. Subsequent sections analyze the *computation time* and evaluate the feasibility of using a data subset to conserve computational resources and time. The findings of the *ablation study* are also presented to shed light on this aspect further.

7.1. Quantitative Results

The section aims to provide a comprehensive understanding of the relationships and patterns observed within the data, shedding light on crucial findings that contribute to answering the research questions. The results are showcased using suitable statistical methods and visual representations, including tables, graphs, and charts.

The section centers on presenting and analyzing quantitative findings from the final configurations, as outlined in table 5.2, excluding the time-based merging strategy. This particular strategy is omitted due to its high computational demand and can be explored in future research.

The results are presented individually for each dataset to promote clarity and coherence. Each dataset section is systematically organized into four dataset subsections

- Baseline Performance
- Overall Performance
- Discrete Performance
- Continuous Performance

each, excluding »*Overall Performance*«, corresponds to a specific configuration mentioned above, resulting in nine configurations utilized for model training.

Several configurations undergo multiple training iterations when feasible to provide reliable and robust performance. This practice is specific to the dataset and will be discussed in the relevant chapters. For further

analysis, the top three performing configuration runs for each dataset subsection are documented in the appendix under Chapter G.

7.1.1. Medaka Fish

As outlined in Section 5.1, the MFD comprises images of the Medaka Fish cardiac system, including four classes, one of which is the background class. This subsection elaborates on the results of the four configurations detailed in Section 5.2, specifically configurations 1 through 4, as shown in table 5.2. These configurations consist of the baseline, discrete, and continuous merge strategy configurations.

Baseline Performance

Table 7.1 summarizes the baseline results for the MFD. Each row represents the average performance by the loss function. This baseline average was created using configuration no. 1 defined in Section 5.2. This configuration underwent six training rounds for a reliable performance assessment, leading to the development of 108 models. The final column of the table, »*PPV vs. TPR*«, determines whether a scenario presents high precision with low recall or low precision with high recall.

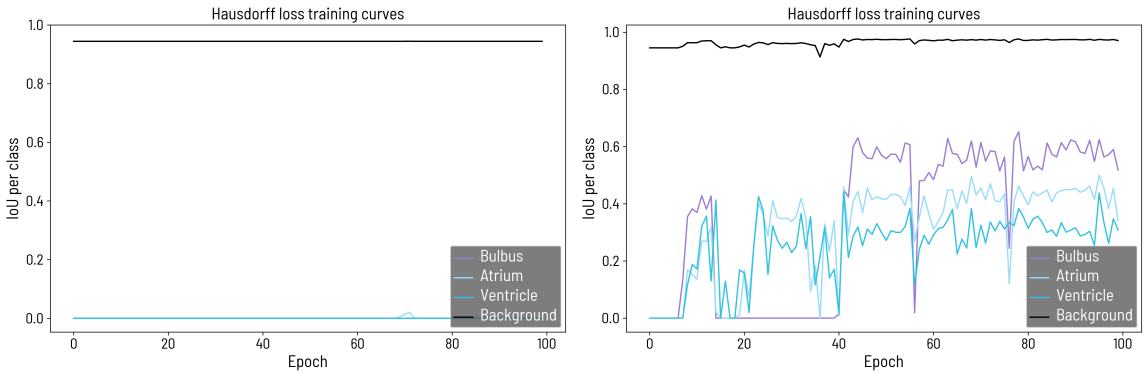
Distribution-based Results
The most notable results were achieved by the CE and the FL, which performed comparably well on average. The most significant difference, approximately 6%, was observed in the Bulbus class, which was better predicted by the FL. This observation aligns with the findings of [222], which reported that the Bulbus class is the most challenging to predict. This outcome is reasonable, as the FL is known to down-weight more straightforward examples and focus on more difficult ones.

Region-based Results
DL and TL performed relatively well, with scores 3.9 and 4.2% lower than the CE and FL, respectively. Interestingly, even though the TL exhibited just an IoU that was 0.3% lower than the IoU of the CE, its TPR was 2.1% higher. Among all the loss functions in the distribution and region-based domain, the TL demonstrated a higher TPR against a typically higher PPV, indicating successful implementation, as discussed in Section 2.3.2.

Boundary-based Results
Although the Boundary Loss (BL) converged several times during testing, it appeared incapable of converging for this final baseline configuration which can be typical for this type of loss, as it was introduced as an addition rather than intended to serve as a standalone loss function [133]. In contrast, the Hausdorff Loss (HL) was much more stable. Although some runs did not converge, the average performed reasonably well.

While initial tests with the original loss defined in 2.29 denoted as HL, proposed by [129] had a shallow convergence rate for baseline training, this work uses the proposed variant HL_1 defined in 2.30 which, after some extensive testing, demonstrated a significantly more robust performance.

Fig. 7.1 illustrates two models trained with the original loss described in [129] on image (a) and the improved variant illustrated on image (b) indicating a meaningful implementation.



(a) HL loss training curves

(b) HL_1 loss training curves

Fig. 7.1. Learning curves for a model trained with the regular (a) HL and enhanced (b) HL function. It can be seen that even the difficult Bulbus class starts to improve around epoch 40, showing a drastic performance increase.

No.	Loss type	Loss	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
1	***DB***	***CE***	***0.672***	***0.976***	***0.697***	***0.530***	***0.485***	***0.808***	***0.766***	***PPV***
2	***DB***	***FL***	***0.672***	***0.976***	***0.750***	***0.515***	***0.445***	***0.814***	***0.762***	***PPV***
3	RB	DL	0.633	0.973	0.673	0.465	0.421	0.782	0.725	PPV
4	RB	TL	0.630	0.971	0.656	0.408	0.487	0.736	0.749	TPR
5	BB	HL_1	0.542	0.968	0.343	0.462	0.394	0.678	0.595	PPV
6	BB	BL	0.096	0.329	0.016	0.011	0.027	0.252	0.259	TPR
		Grand Average	***0.528***	***0.850***	***0.502***	***0.392***	***0.366***	***0.667***	***0.629***	***PPV***

Tab. 7.1. The table presents a summary of the average values of various metrics for six baseline models trained for each loss function and each selection percentage. The total number of models trained for this baseline result is 108. The columns $IoU_0, \ldots, IoU_3$ represent the four classes, Background, Bublus, Atriums and Ventricle. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

Overall Performance

The subsequent pair of heatmaps exhibit the average performance for every combination of loss and merge strategy, based on configurations no. 2 and 3 detailed in table 5.2. Three hundred sixty models, derived from the product of 6 merge strategies, 20 loss combinations, and 3 dataset subsets, were trained for configuration 2. Configuration no. 3 provides an additional model count 217, resulting in an overall 577 for the MFD.

(a) Overall performance of double combinations

(b) Overall performance of triple combinations

Fig. 7.2. Average IoU for all double and triple loss-merge combinations.

These heatmaps, showcasing average scores, provide quick and intuitive insights into the framework's performance. For the double loss combinations, it is immediately apparent that the AVG, HARMONIC, NWS, and WS strategies excel, while the PBM, MIN, and MAX strategies underperform. Columns demonstrating the averages for each loss combination enable rapid identification of high-performing options. The same advantages hold for the heatmap depicting the triple loss combinations.

Although a direct comparison between the double and triple loss combinations is not feasible, a noteworthy characteristic observed in both categories is the similar performance trend across different merge strategies. Specifically, the MIN and MAX strategies consistently demonstrated suboptimal performance for both loss combinations. On the other hand, the PBM strategy has performed better on average for triple loss combinations.

Discrete Performance

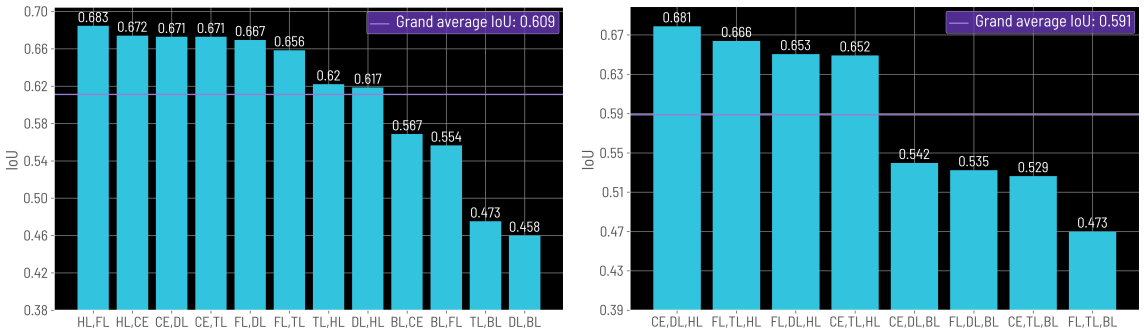
The discrete performance results illustrate additional graphs from the loss combination and merge strategy perspective. The bars of each graph are displayed in descending order from left to right by IoU. The discrete performance results display models from configuration no. 2, which consists of 360 models.

Loss combination results

Fig. 7.3 displays all average double and triple loss combination results in descending order. We can see that the top six double loss combinations show similar performance and even topping the top-performing baseline results. Interestingly all losses, including the BL, are performing rather low for the double and triple loss combinations.

Looking at table G.2, which displays the averages of the top three performing models for each double loss combination and selection percentage, there are three combinations HL, FL, CE, DL, and HL, CE which exceed

the top three baseline averages.



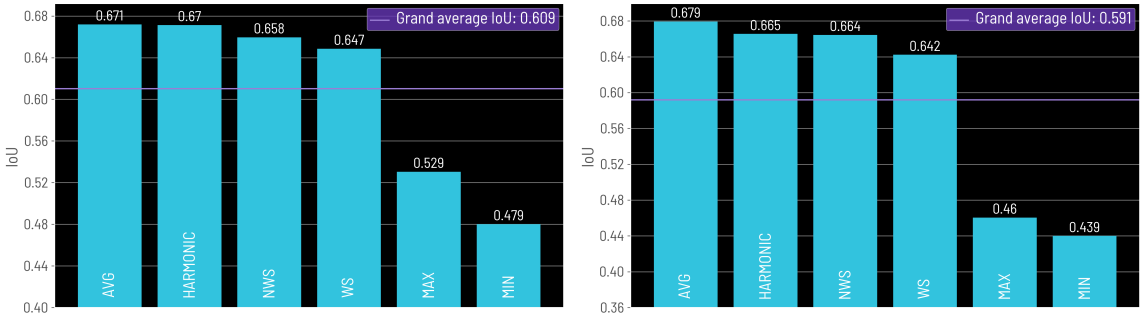
(a) Average IoU for Each Double Loss Combination

(b) Average IoU for Each Triple Loss Combination

Fig. 7.3. The figures showcase the average IoU for loss combination, as well as their corresponding selection percentages, derived from an extensive set of 360 models trained on the MFD.

Merge strategy results

Fig. 7.4 illustrates the average scores for all trained models by merge strategy. Both graphs indicate a rather stable performance except for the MAX and MIN strategies.



(a) Average IoU for Each Double Merge Strategy

(b) Average IoU for Each Triple Merge Strategy

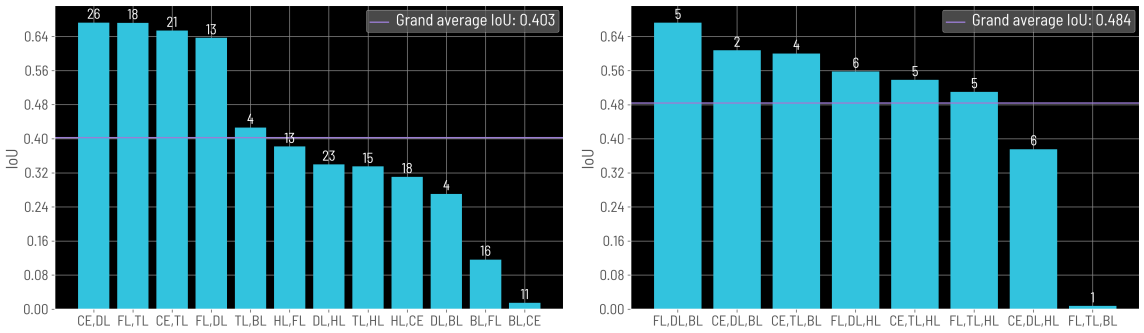
Fig. 7.4. The figures showcase the average IoU for each merge strategy, as well as their corresponding selection percentages, derived from an extensive set of 360 models trained on the MFD.

Continuous Performance

The models examined in this section were trained to utilize configuration no. 3 as outlined in table 5.2. Generally, comparing models across different configurations may present challenges due to differing numbers. While the model count in discrete configurations is predefined, continuous configurations employ a Hyperband search algorithm to identify suitable hyperparameters for the performance-based merge strategy. Fig. 7.5 presents the average result for each loss combination implementing this strategy. The bars also depict the model count, which can vary significantly due to the Hyperband algorithm's specific selection

mechanism. Notably, top-performing combinations were chosen more frequently, particularly in the case of high-performing double-loss combinations.

Interestingly, while some double loss combinations with the HL performed commendably in discrete configurations, combinations involving boundary-based elements demonstrated poor performance when using the performance-based strategy for double losses. Conversely, at least among the top performers, the BL combinations surpassed the HL combinations for triple losses.



(a) Average IoU for Each Double Loss Combination

(b) Average IoU for Each Triple Loss Combination

Fig. 7.5. The figures showcase the average IoU for loss combination, derived from an extensive set of 217 models trained on the MFD using the performance-based merge strategy.

Fig. 7.6 plots the hyperparameter ranges on the x-axis against the average IoU on the y-axis. Notably, the `pbm_I` parameter exhibits a clear trend, wherein a higher value tends to improve performance. By analyzing table G.6 and table G.7, which present the top-performing models generated by the continuous configuration no. 3, we observe that the average `pbm_I` value for double loss combinations is 0.755, while for triple loss combinations, it is 0.504. Additionally, the `pbm_alpha` value for these high-performing models was, on average, approximately 6.

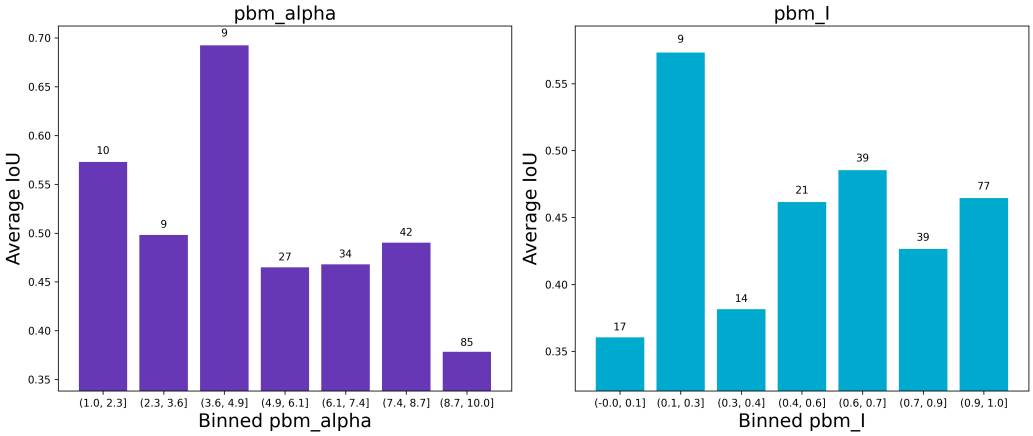


Fig. 7.6. Hyperparameter analysis for performance-based merging. The x-axis corresponds to the hyperparameter `pbm_alpha` and `pbm_I` defined in table E.1 and Section 5.2. Both values are float with `pbm_alpha` $\in (1, 10]$ and `pbm_I` $\in [0, 1]$. The bars are labeled with the model count and represent the average IoU for the indicated range.

Specific Performance

As a specific loss-merge combination for the specific configuration no.5 defined in table 5.2, the PBM strategy along the combinations of FL,DL and BL was used, and resulted in a grand average of 0.695 for 18 models trained, surpassing the continuous average by 2.5% as well as the best average baseline implementation by 2.3%. The detailed results for each model can be viewed in Section G.1.4.

7.1.2. Skin Lesion

This second dataset comprises dermoscopic images of skin lesions, as described in more detail in Section 5.1. The dataset consists of two classes: foreground and background. Configurations 6 to 9 from table 5.2 were used for this dataset, including the baseline, one discrete, one continuous and the specific configuration, similar to the MFD.

Baseline Performance

Table 7.2 summarizes the baseline results for the SLD. Each row represents the average performance by the loss function. This baseline average was created using configuration no. 6 defined in Section 5.2. This configuration underwent six training rounds for a reliable performance assessment, leading to the development of 108 models. The final column of the table, »*PPV vs. TPR*«, determines whether a scenario presents high precision with low recall or low precision with high recall.

Distribution-based results
Although a model trained with the CE achieved the highest score among all models, see G.9 (No. 1), the

average result over nine models was 3.7% lower than the average trained with the FL. Interestingly, none of the top-performing CE models used the entire dataset but subsets of 64 and 32%. All models that used the entire dataset for training resulted in an average of 14.7% lower than the top-performing model, which only used 64% of the data. See table G.9 for further information on the top-performing models.

The FL achieved an average IoU of 0.594 across the 18 baseline models. The best-performing model trained with the FL scored 0.666, nearly as good as the overall best-performing model. As expected, most top-performing models trained with the FL used the entire dataset.

Region-based results

The DL achieved an average IoU of 0.560, resulting in a performance decrease of 3.4% compared to the FL average. Among the region-based losses, the TL achieved slightly better performance with an IoU of 0.574. Even if the average TPR of the TL did not top the average results of FL and DL, it still represents the top-performing model in terms of TPR, demonstrating its intended function as described in [218]. See result table G.9 (No. 28).

Boundary-based results

Among the boundary-based loss functions, the HL achieved an average IoU of 0.542 and the BL an average IoU of 0.287 which seemed to have been converging for this dataset but still lacks meaningful results compared to other loss functions used.

No.	Loss type	Loss	IoU	PPV	TPR	PPV vs. TPR
1	DB	CE	0.557	0.753	0.682	PPV
2	***DB***	***FL***	***0.594***	***0.792***	***0.723***	***PPV***
3	RB	DL	0.560	0.739	0.714	PPV
4	RB	TL	0.574	0.751	0.713	PPV
5	BB	HL_1	0.542	0.750	0.677	PPV
6	BB	BL	0.287	0.519	0.520	TPR
		Grand Average	***0.512***	***0.712***	***0.666***	***PPV***

Tab. 7.2. The table presents a summary of the average values of various metrics for six baseline models trained for each loss function and each selection percentage. The total number of models trained for this baseline result is 108. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

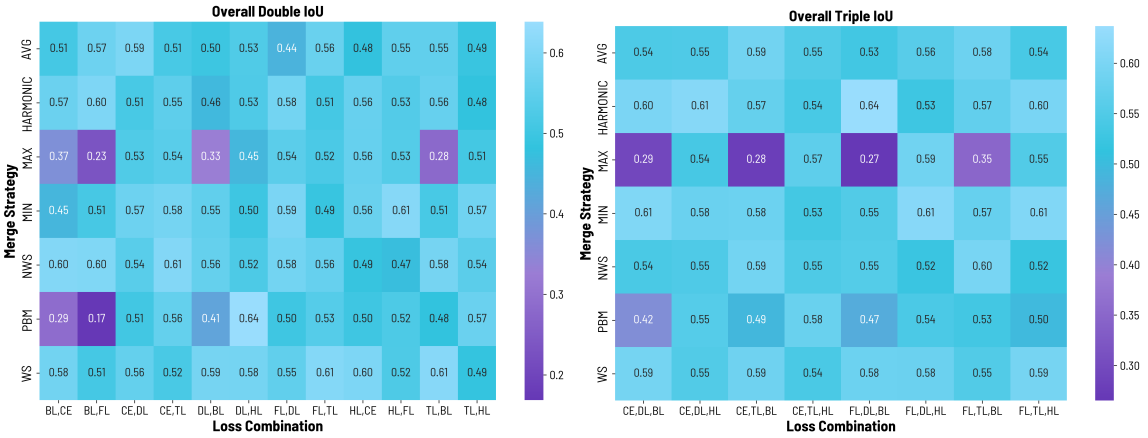
Overall Performance

The pair of heatmaps illustrated below portray the cumulative results obtained from the SLD, which was generated via the execution of configurations 7 and 8 as defined in 5.2. Specifically, discrete configuration no. 7 led to the training of 360 models, while continuous configuration no. 8 contributed an additional 146 models, thereby yielding a total of 506 models.

It is important to note that while the model count for the discrete configuration is predetermined, the continuous configuration necessitates an exploration of an array of hyperparameters. This exploration is facil-

itated by the Hyperband algorithm, which strategically favors promising combinations while simultaneously discarding those that demonstrate inferior performance. In this context, »*combination*« refers to integrating loss combinations, merge strategies, and their corresponding hyperparameters.

The relatively fewer models used for the continuous configuration within this dataset can be attributed to its substantial size. Specifically, the training set of the SLD is approximately 3.5 times larger than that of the MFD. Consequently, it demands significantly greater computational resources and time, hence the more limited model use in the continuous configuration.



(a) Overall performance of double combinations

(b) Overall performance of triple combinations

Fig. 7.7. Average IoU for all double and triple loss-merge combinations.

Analyzing the two heatmaps depicted in Fig. 7.7, it becomes apparent that all merging strategies, except for MAX and PBM, display good results for this specific dataset. Two combinations stand out above the rest: firstly, the DL, HL loss combined with the PBM strategy for double loss combinations, and secondly, the FL, DL, BL combination with the HARMONIC strategy for triple combinations. Given PBM's generally low average performance, the former combination's high performance might be an outlier, requiring further examination. In contrast, the success of the latter combination appears more consistent, mainly due to the overall decent performance of the HARMONIC strategy.

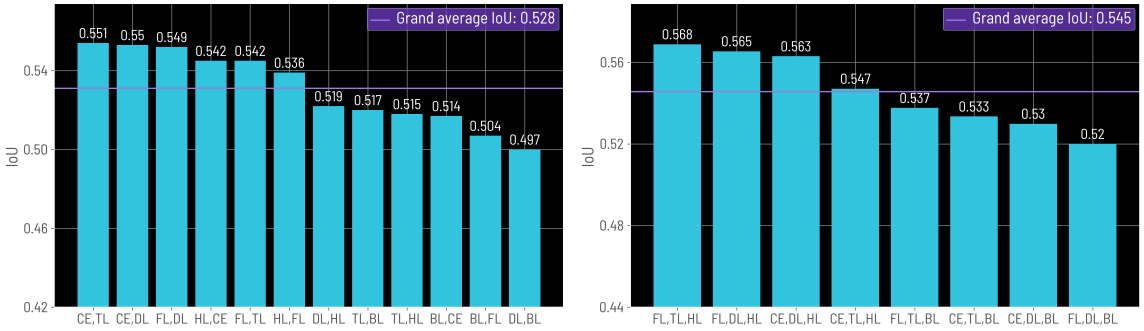
Discrete Performance

This subsection analyzes the discrete configuration no. 7 exclusively from a loss combination and merge strategy perspective. The average results are generally the average of the respective columns and rows of the two heatmaps presented above.

Loss combination results

Fig. 7.8 displays the average result of 360 models, each trained using a different loss combination. Image (a) showcases all dual loss combinations, while image (b) exhibits all triple loss combinations arranged in

descending order by IoU. As observed with the MFD, combinations involving the BL performed below the grand average for both dual and triple combinations. For dual loss combinations, distribution and region-based losses outperformed others, although some combinations with the HL also yielded decent results.



(a) Average IoU for Each Double Loss Combination

(b) Average IoU for Each Triple Loss Combination

Fig. 7.8. The figures showcase the average IoU for loss combination, as well as their corresponding selection percentages, derived from an extensive set of 360 models trained on the SLD.

Merge strategy results

Fig. 7.9 displays the average performance categorized by merge strategy. In the case of the Medaka Fish, the AVG strategy ranked among the top strategies for all loss combinations. However, for the SLD, AVG only reached the fifth position for dual loss combinations and the fourth for triple loss combinations. Meanwhile, the static weighting strategies, WS and NWS, showed stable performance on average for both dual and triple loss combinations.

(a) Average IoU for Each Double Merge Strategy

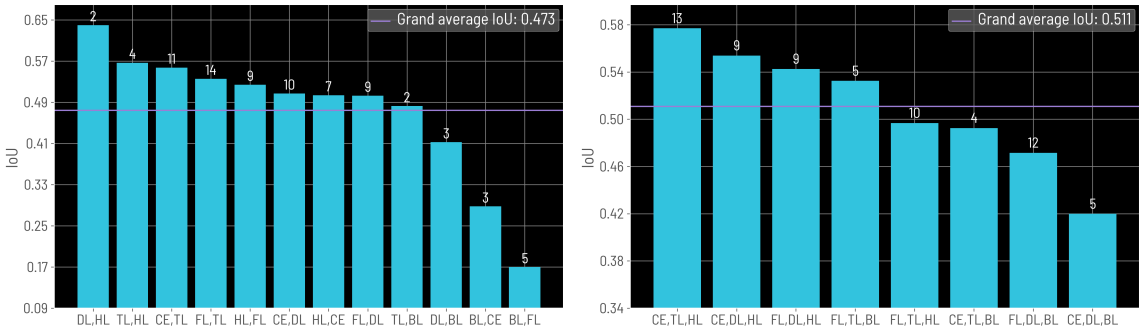
(b) Average IoU for Each Triple Merge Strategy

Fig. 7.9. The figures showcase the average IoU for each merge strategy, as well as their corresponding selection percentages, derived from an extensive set of 360 models trained on the SLD.

Upon examining the top-performing results in the appendix, specifically in tables G.12 and G.13, we find that no average loss combination outperforms the top baseline models summarized in table G.9. However, there are individual exceptions. Specifically, model no. 37 and 91 from table G.12, as well as models no. 28 and 37 from table G.12, outperform the top-performing baseline model no. 1 from table G.9.

Continuous Performance

Fig. 7.10 displays the performance-based results obtained from continuous configuration no. 8. defined in table 5.2. Interestingly, the two best results from the double loss combinations (a) include the HL, which generally seemed to have been performing rather decent for this dataset. The BL combinations performed rather low again. The triple-based loss combinations of the CE, TL, and HL displayed a decent result consisting of the average of 13 trained models with a performance close to the best baseline model average.



(a) Average IoU for Each Double Loss Combination

(b) Average IoU for Each Triple Loss Combination

Fig. 7.10. The figures showcase the average IoU for loss combination, derived from an extensive set of 217 models trained on the SLD using the performance-based merge strategy.

Fig. 7.11 examines the two hyperparameters `pbm_alpha` and `pbm_I` where the graph for the former does not show a clear trend while the latter indicates that models with a higher `pbm_I` are performing better.

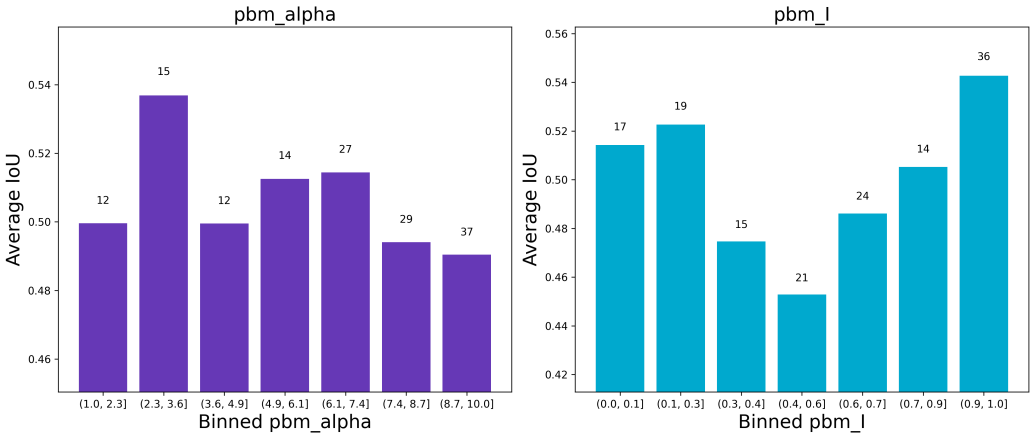


Fig. 7.11. Hyperparameter analysis for performance-based merging. The x-axis corresponds to the hyperparameter `pbm_alpha` and `pbm_I` defined in table E.1 and Section 5.2. Both values are float with `pbm_alpha` $\in (1, 10]$ and `pbm_I` $\in [0, 1]$. The bars are labeled with the model count and represent the average IoU for the indicated range.

Specific Performance

As a specific loss-merge combination for the specific configuration no.10 defined in table 5.2, the NWS strategy along the combinations of CE and TL was used and resulted in a grand average of 0.645 for 18 models trained, surpassing the discrete average by 3.5% as well as the best average baseline implementation by 5.1%. The detailed results for each model can be viewed in Section G.2.4.

7.1.3. IDRID

The third dataset comprises retinal images from patients with diabetic retinopathy and contains five classes, including the background. The IDRID dataset is highly challenging due to its small size, with only 81 images, and significant class imbalance, where the background region averages 92.2%. Furthermore, the classes are dispersed across the entire retina, adding an extra layer of difficulty for proper segmentation. As with the other datasets, the results are presented for the four configurations in table 5.2 (rows 11 to 14), which include the baseline, one discrete, and two continuous merge strategy configurations.

Baseline Performance

This subsection presents the baseline results for the IDRID. The models for this baseline training were obtained using configuration no. 11 from the configuration table 5.2.

Configuration no. 11 was trained six times to provide a robust performance estimate. Table 7.3 displays the average results of the six models for each loss function introduced earlier. The CE achieves the best performance, with an average IoU of 0.341. The grand average across all losses is 9.2% lower at 0.249. Note that due to the small dataset size, it was not further subdivided into smaller subsets.

Distribution-based results
The distribution-based results with the CE and the FL are the top performers for this baseline training, where the CE is even 3% higher than the average models trained with the FL. For both losses, it can be observed that the IoU for the Haemorrhages and Microaneurysm classes were nearly zero. As expected, the background class displayed decent results, which can generally be neglected due to the high-class imbalance for this dataset, as illustrated in table 5.1. The best meaningful results were achieved for the Optic Disc, representing the most straightforward class, followed by the Hard Exudates.

Region-based results
The region-based loss functions performed worse on average, with up to 8.9% less than the top model from the distribution-based results. The TL, as expected, emphasized a higher TPR, even if its overall performance was lower than models trained with the DL.

Boundary-based results
While the HL appeared to converge, providing similar results to the region-based performance, the BL seemed to fail, even for the background class.

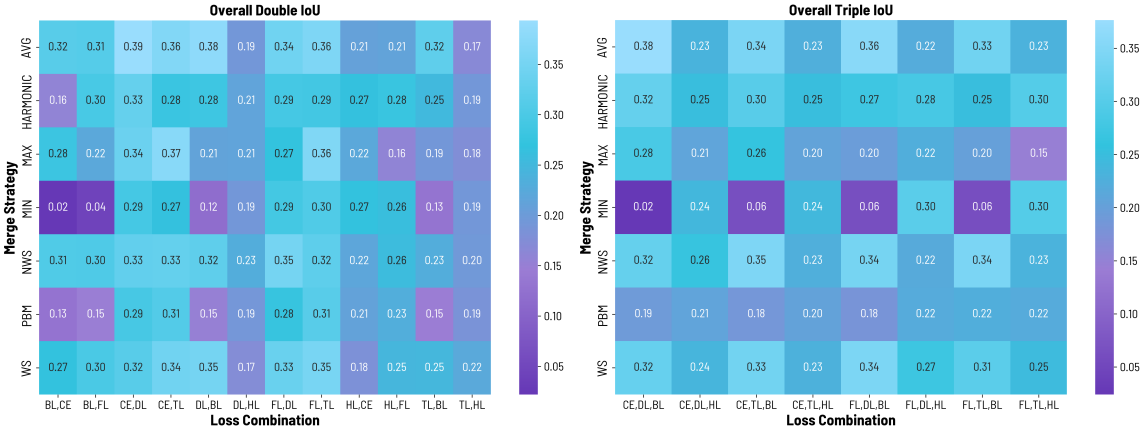
No.	Loss type	Loss	Iou	IoU 0	IoU 1	IoU 2	IoU 3	IoU 4	PPV	TPR	PPV vs. TPR
1	***DB***	***CE***	***0.341***	***0.955***	***0.008***	***0.262***	***0.002***	***0.477***	***0.503***	***0.410***	***PPV***
2	DB	FL	0.311	0.944	0.002	0.253	0.020	0.337	0.517	0.388	PPV
3	RB	DL	0.275	0.931	0.025	0.183	0.033	0.202	0.431	0.360	PPV
4	RB	TL	0.252	0.956	0.000	0.000	0.000	0.301	0.287	0.292	TPR
5	BB	HL_1	0.260	0.944	0.000	0.000	0.000	0.354	0.322	0.309	PPV
6	BB	BL	0.031	0.131	0.011	0.006	0.001	0.006	0.198	0.166	PPV
		Grand Average	***0.249***	***0.814***	***0.009***	***0.128***	***0.010***	***0.284***	***0.386***	***0.326***	***PPV***

Tab. 7.3. The table presents a summary of the average values of various metrics for six baseline models corresponding trained for each loss function. The total number of models trained for this baseline result is 36. The columns $IoU_0, \ldots, IoU_4$ represent the five classes, Background, Haemorrhages, Hard Exudates, Microaneurys and Optic Disc. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

Overall Performance

The following two heatmaps represent the average performance for every loss and merge strategy combination of the models obtained from configuration no. 12 and configuration no. 13 presented in 5.2. Configuration no. 12 consists of a model count of 120 models $= 6$ merge strategies $\times$ 20 loss combinations. Even if the best practice could not be observed by training this configuration six times equally as the baseline, the configuration was trained two times to provide a more robust average with just a single run resulting in a total of 240 models from configuration no. 12. The heatmap additionally consists of the continuous configuration no. 13 with an additional model count of 430. Configuration no. 13 requires a search through a continuous hyperparameter space, so the model count is not set beforehand.

While the exact average for each loss combination and merge strategy is discussed in subsequent chapters, the heatmap already provides some valuable intuition about suitable or unsuitable combinations for this dataset. The average top-performing loss/merge combination for the double combo achieves a substantial increase of 4.9 % in terms of IoU compared to the top-performing baseline models. It can also be seen that some loss combinations or merge strategies fail to provide meaningful results, such as the MIN strategy or the TL, HL loss combination.



(a) Overall performance of double combinations

(b) Overall performance of triple combinations

Fig. 7.12. Average IoU for all double and triple loss-merge combinations.

A noticeable similarity comparing heatmap (a) from 7.12 with heatmap (b) from 7.12 is that the MIN and PBM strategies performed very low in both results. Fig. 7.13 illustrates an example where a generally low-performing loss with small values gets selected exclusively as the high-performing loss initiates with larger values.

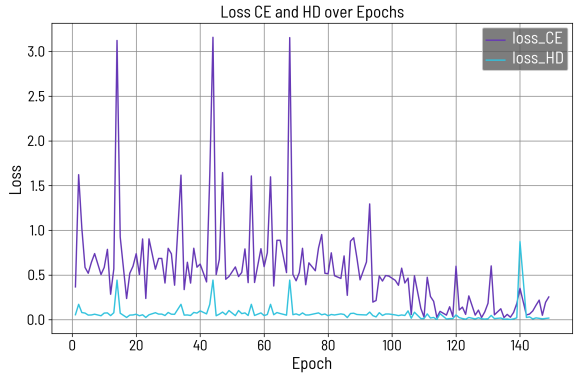


Fig. 7.13. Learning curves for a model trained with a HLCE combination using the MIN strategy. It can be seen that the strategy always selects the HL as feedback for gradient descent, completely ignoring the CE due to its higher initial values.

The following section provides a detailed result for every loss combination and merge strategy.

Discrete Performance

This section exclusively presents the models obtained from configuration no. 12 presented in table 5.2. The configuration is named discrete as it does not require an extensive search of hyperparameters and

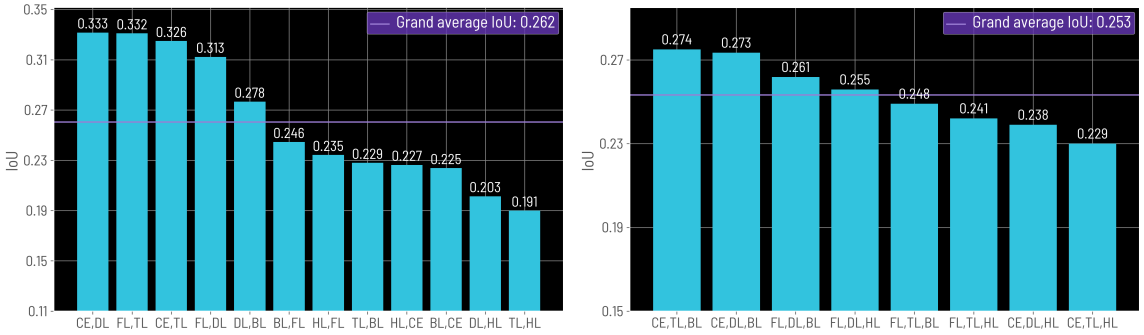
consists of a fixed number of models for every configuration run. The following results are presented from the loss combination and merge strategy perspective, illustrated as bar plots sorted by the best-performing averages.

Loss combination results

Fig. 7.14 illustrates the average performance for all 240 models created from training configuration no. 12, by double and triple loss combinations. The height of each bar represents the average results in terms of IoU corresponding to the average of the respective column of the double or triple heatmap presented earlier.

For the double loss combinations, it can be immediately seen that the average results obtained from using distribution and region-based loss combinations outperformed any combination containing a boundary-based loss function. The top four loss combination averages lay above an IoU of 0.3, close to the best-performing baseline averages. Looking at the G.18, which lists the top performing results from double loss combinations for the IDRID, we see that the CE, TL average, is 3.1% higher than the average of the top three baseline models represented by model no. 1 to 3 from table G.17.

The triple loss combinations resulted in a lower performance indicating that boundary-based loss functions could not improve the performance compared to double loss combinations. Comparing the top three performing models (No. 13,14,15) from table G.19 to the baseline top three results from the baseline, we can observe an increase of 2.2 %.



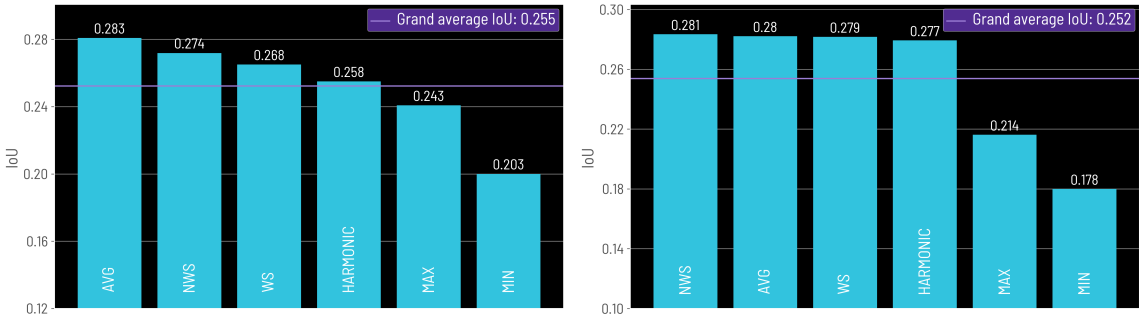
(a) Average IoU for Each Double Loss Combination

(b) Average IoU for Each Triple Loss Combination

Fig. 7.14. The figures showcase the average IoU for loss combination, derived from an extensive set of 240 models trained on the Indian Diabetic Retinopathy Image Dataset (IDRID).

Merge strategy results

Fig. 7.15 showcases the average IoU for each merge strategy. The averaging and static weighting strategies outperform others, while the extreme point strategies lag. Interestingly, despite the overall low performance of the MAX strategy, table G.20 reveals that it was used by the second-highest performing models (No. 7,8,9). A closer examination of 7.12, displaying the overall results heatmap, reveals a mixed performance for the MAX strategy, with some decent outcomes counterbalanced by several poor ones.



(a) Average IoU for Each Double Merge Strategy

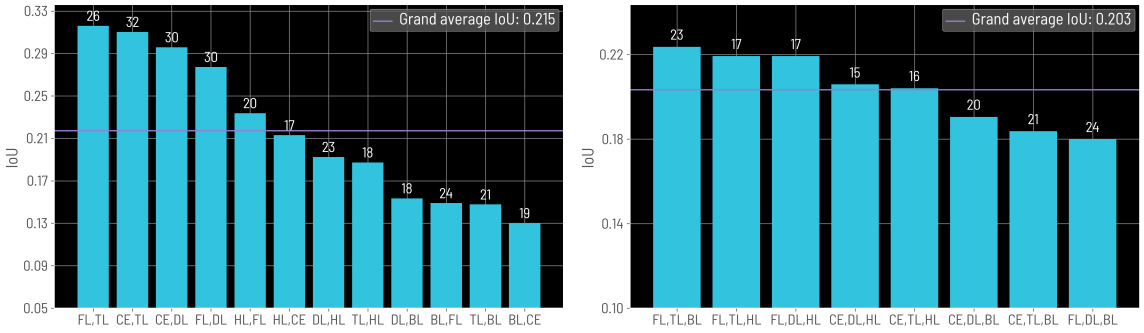
(b) Average IoU for Each Triple Merge Strategy

Fig. 7.15. The figures showcase the average IoU for each merge strategy, derived from an extensive set of 240 models trained on the Indian Diabetic Retinopathy Image Dataset (IDRID).

Continuous Performance

Fig. 7.17 presents the average performance for each loss function, employing the performance-based merge strategy from configuration no. 13 as outlined in 5.2. The graph also highlights the model count for each loss combination, which is significantly larger for the IDRID dataset, owing to its reduced computational resource requirements. The Hyperband search algorithm preferentially selected combinations that showed promise, as evidenced by the higher model count associated with combinations that achieved a greater average IoU. However, the overall performance of the performance-based strategy was noticeably lower than that of the top-performing discrete strategies, partly due to a few underperforming combinations such as BL, CE, or TL, BL in the double loss combination results.

Despite the different model counts and the need to traverse a continuous hyperparameter space, making a direct comparison to discrete strategies challenging, some insights can still be gained. Although the average performance was somewhat low, table G.22 reveals that several high-performing models could match those from the discrete configurations, demonstrating the efficacy of the hyperparameter selection in the performance-based strategy.



(a) Average IoU for Each Double Loss Combination

(b) Average IoU for Each Triple Loss Combination

Fig. 7.16. The figures showcase the average IoU for loss combination, derived from an extensive set of 432 models trained on the Indian Diabetic Retinopathy Image Dataset (IDRID) using the performance-based merge strategy.

Fig. 7.17 shows the average IoU corresponding to different model counts, illustrating a notable trend that a `pbm_I` value approximating one yields better performance. The results concerning `pbm_alpha` indicate optimal performance around 6.5. Upon reviewing the top-performing results enumerated in table G.22 and table G.23, it is observed that the average values for `pbm_alpha` and `pbm_I` were 3.57 and 0.795 for the former, and 4.448 and 0.802 for the latter, respectively.

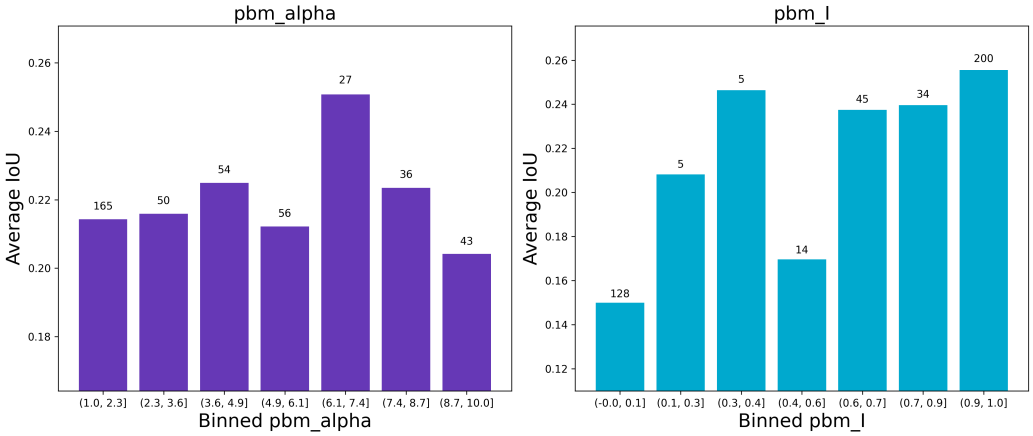


Fig. 7.17. Hyperparameter analysis for performance-based merging. The x-axis corresponds to the hyperparameter `pbm_alpha` and `pbm_I` defined in table E.1 and Section 5.2. Both values are float with `pbm_alpha` $\in (1, 10]$ and `pbm_I` $\in [0, 1]$. The bars are labeled with the model count and represent the average IoU for the indicated range.

Specific Performance

As a specific loss-merge combination for the specific configuration no.15 defined in table 5.2, the AVG strategy along the combinations of CE and DL was used, and resulted in a grand average of 0.38 for 6 models trained, failing to reach the discrete by 1% but outperforming the best baseline average by 3.9%. The detailed results for each model can be viewed in Section G.3.4.

7.2. Qualitative Results

This section offers a comprehensive overview and qualitative analysis of the results derived from the proposed framework across the suggested datasets. The visual examination of these outcomes can be critical, as it enables a quick and intuitive understanding of the model's performance, identifies its shortcomings, and potentially signals system bugs. A qualitative review can also reveal issues that numerical metrics overlook, enabling the user to enrich the understanding of the model's capabilities.

While the principal aim was to show if the proposed implementation outperforms the standard baseline methods, detecting a visual performance increase may be challenging, mainly if improvements are minimal. Nevertheless, the subsequent sections will analyze five inputs for each dataset predicted by a baseline model and a model using the proposed loss merging framework.

7.2.1. Medaka Fish

No.	R-time(m)	Loss	Selection %	Strategy	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
1	73	CE	1.00	-	0.697	0.974	0.756	0.471	0.586	0.807	0.832	TPR
2	138	HD,FL	1.00	AVG	0.762	0.984	0.751	0.607	0.705	0.874	0.834	PPV

Tab. 7.4. The table presents two distinct models that will be used in this section for a qualitative comparison. Model no.1 is constructed using a baseline setup, while model no.2 has been trained with the proposed loss merging framework. The columns $IoU_0, \ldots, IoU_3$ represent the four classes, Background, Bulbus, Atrium and Ventricle. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

Fig. 7.18 displays the qualitative results of a baseline model trained on the Medaka Fish Dataset (MFD). From Table 7.4, we observe that model no. 1 achieved an IoU score of 0.697, which is approximately 2% higher than the top-performing baseline averages discussed in Section 7.1.1. The Bulbus class exhibited the best prediction, with an IoU score of 0.756. The images reveal minimal false positives for this class, indicating the model's superior recall performance compared to its precision.

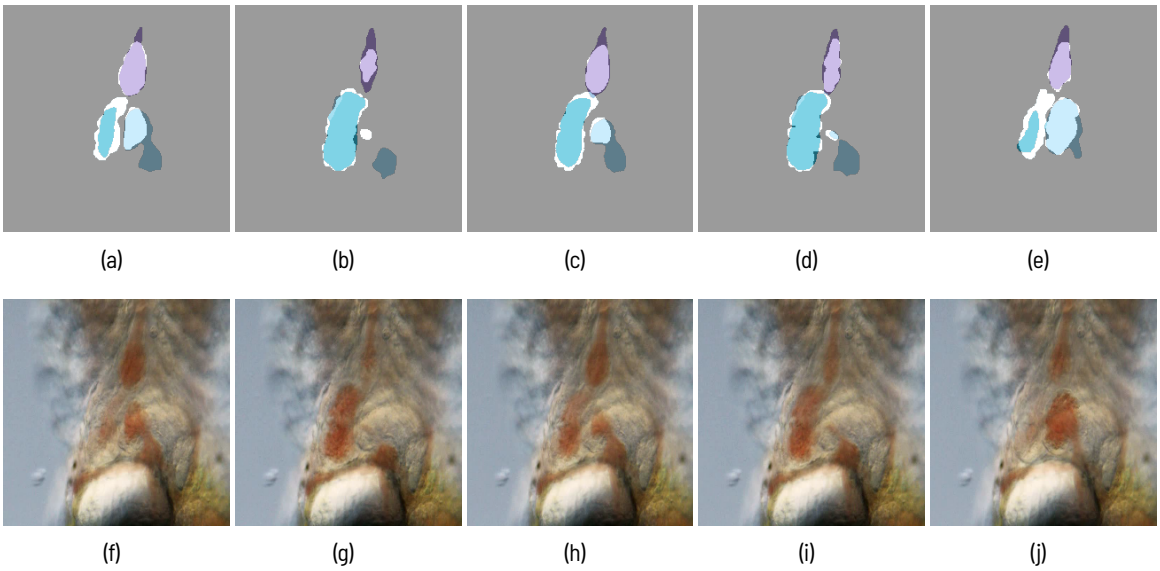


Fig. 7.18. Image (a) to (e) illustrate baseline predictions as colored masks against the inverse of the ground truth mask colored in grey. We can see three classes to predict named **Bulbus**, **Atrium** and **Ventricle**. The corresponding quantitative results are listed in table 7.4 as model no.1. Image (f) to (j) depict the input features for each prediction above.

Fig. 7.19 showcases the qualitative results of a model that has been trained using a combination of HL and FL, employing the arithmetic averaging strategy. The model attains an IoU score of 0.762. Compared to the baseline model, it achieves considerably higher precision, particularly noticeable for the Atrium class in images (a) through (e). Additionally, the Ventricle class improved significantly by 11.9%.

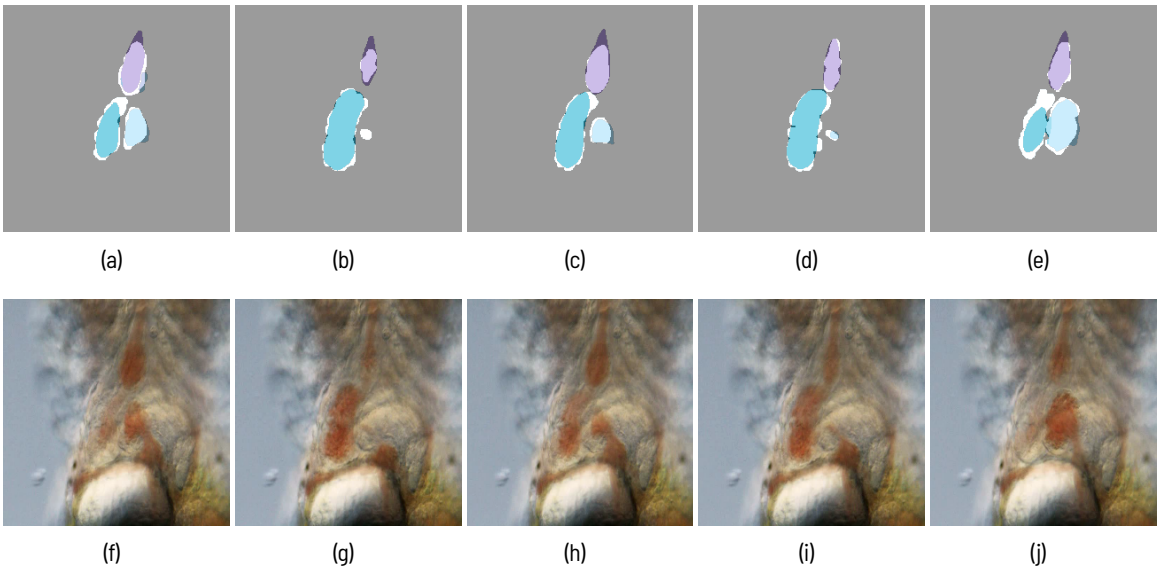


Fig. 7.19. Image (a) to (e) illustrate loss merging predictions as colored masks against the inverse of the ground truth mask colored in grey. We can see three classes to predict named **Bulbus**, **Atrium** and **Ventricle**. The corresponding quantitative results are listed in table 7.4 as model no.2. Image (f) to (j) depict the input features for each prediction above.

7.2.2. Skin Lesion

No.	R-time(m)	Loss	Selection %	Strategy	IoU	IoU 0	IoU 1	PPV	TPR	PPV vs. TPR
1	158	Focal	0.64		0.659	0.821	0.497	0.832	0.793	PPV
2	185	CE,TL	0.64	NWS	0.671	0.812	0.529	0.822	0.814	PPV

Tab. 7.5. The table presents two distinct models that will be used in this section for a qualitative comparison. Model no.1 is constructed using a baseline setup, while model no.2 has been trained with the proposed loss merging framework. The IoU column reflects the average outcomes contrasting the foreground class against the background class where the columns IoU_0, IoU_1 represent the Background and Foreground class individually. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

Fig. 7.20 portrays the qualitative results of a baseline model (no. 1) that was trained using the FL. The model obtained an IoU score of 0.659, which is 6.5% higher than the average of the top-performing baselines discussed in Section 7.1.2. The model also displayed a higher precision than recall, as evidenced by the PPV and TPR columns in Table 7.4. This attribute is reflected in the images, particularly in images (a) and (b), where a larger number of false negatives contribute to a lower true positive rate (TPR).

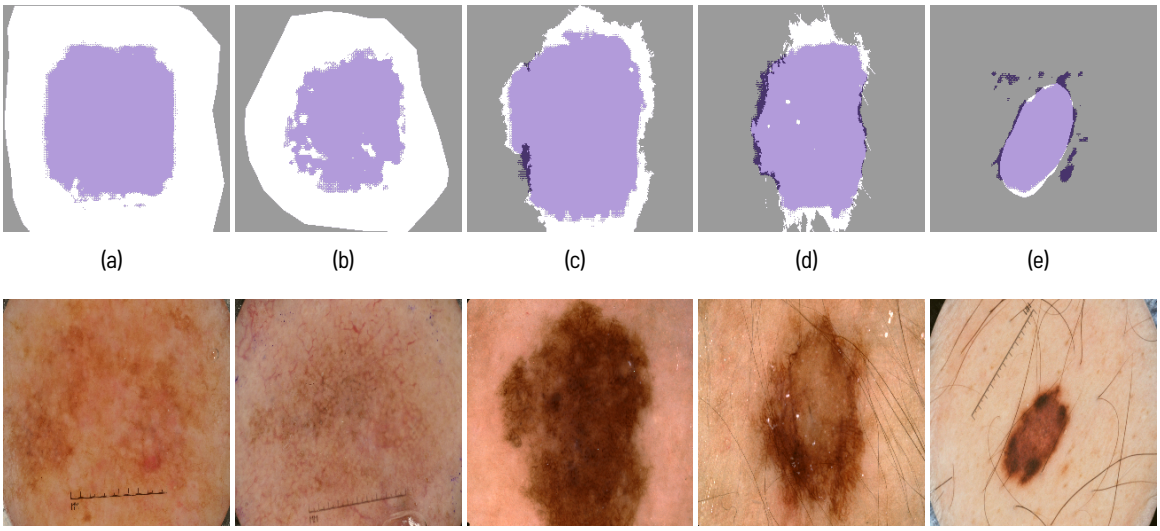


Fig. 7.20. Image (a) to (e) illustrate baseline predictions as colored masks against the inverse of the ground truth mask colored in grey. We can see one class to predict. The corresponding quantitative results are listed in table 7.5 as model no.1. Image (f) to (j) depict the input features for each prediction above.

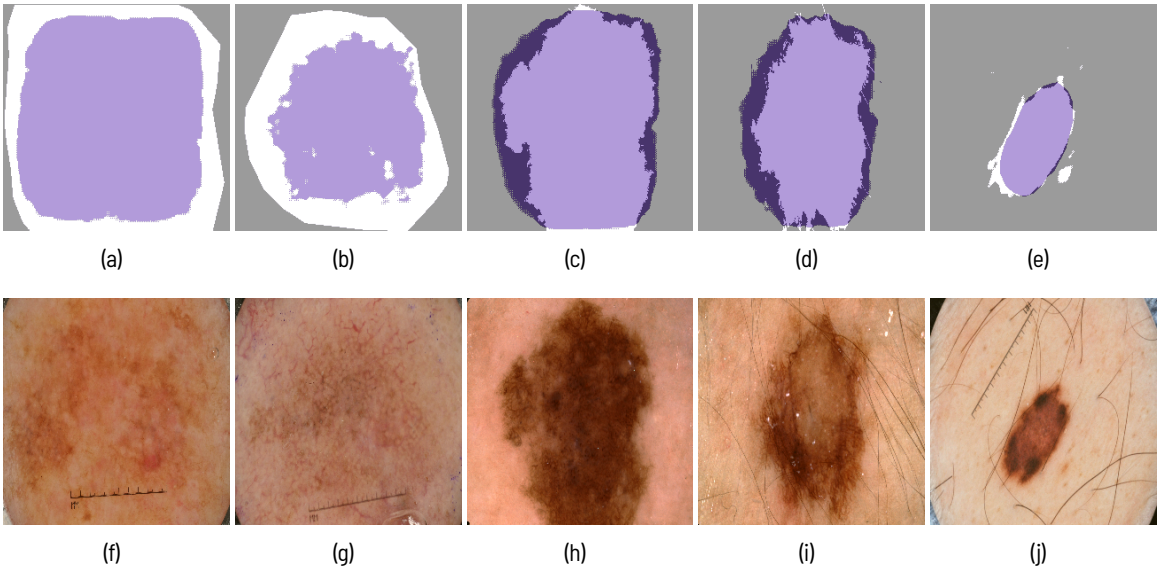


Fig. 7.21. Image (a) to (e) illustrate baseline predictions as colored masks against the inverse of the ground truth mask colored in grey. We can see one class to predict. The corresponding quantitative results are listed in table 7.5 as model no. 2. Image (f) to (j) depict the input features for each prediction above.

Fig. 7.21 demonstrates the performance of a model trained using a combination of CE and TL and merged using the normalized weighted sum (NWS) strategy. The IoU score is marginally higher, by just 1.2%, compared

to the baseline model no. 1 in Table 7.5. Although precision slightly declined, recall increased by 2.1%. This is particularly evident in images (a) through (d), where a notable reduction in false negatives contributes to the higher true positive rate.

7.2.3. Idrid

No.	R-time(m)	Loss	Strategy	IoU	IoU 0	IoU 1	IoU 2	IoU 3	IoU 4	PPV	TPR	PPV vs. TPR
1	42	CE		0.302	0.961	0.000	0.000	0.000	0.551	0.348	0.349	TPR
2	51	CE,DL	AVG	0.398	0.964	0.083	0.323	0.097	0.521	0.646	0.462	PPV

Tab. 7.6. The table presents two distinct models that will be used in this section for a qualitative comparison. Model no.1 is constructed using a baseline setup, while model no.2 has been trained with the proposed loss merging framework. The columns $IoU_0, \ldots, IoU_4$ represent the five classes, Background, Haemorrhages, Hard Exudates, Microaneurys and Optic Disc. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

The following section presents models trained on the IDRID, the most challenging dataset examined in this work. Fig. 7.22 demonstrates five predictions of a model trained using the CE. It is clear that almost no other class, except the Optic Disc, yields satisfactory results for this baseline model. This is reflected by the quantitative results presented in table 7.6 where the Optic Disc additional except the background class exclusively yields any meaningful results.

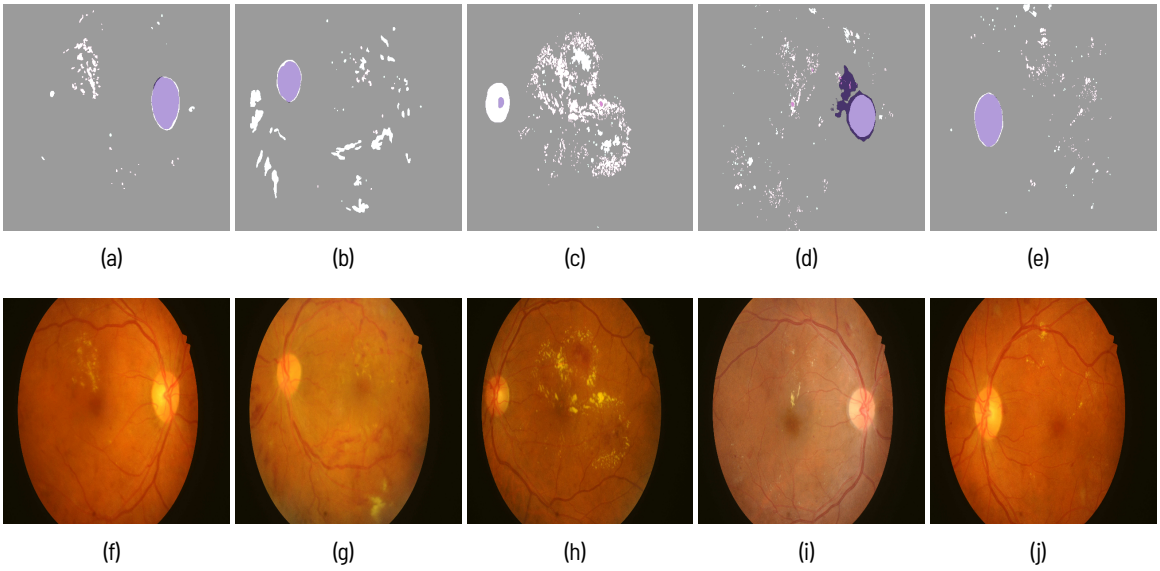


Fig. 7.22. Image (a) to (e) illustrate baseline predictions as colored masks against the inverse of the ground truth mask colored in grey. We can see four classes to predict named **Haemorrhages**, **Hard Exudates**, **Microaneurysms** and the **Optic Disc**. The corresponding quantitative results are listed in table 7.6 as model no. 1. Image (f) to (j) depict the input features for each prediction above.

Fig. 7.23 presents five predictions of a model trained with the combination of CE, DL using the arithmetic averaging strategy (AVG). The average IoU for this model is 9.6% higher, where the Hard Exudates class achieved the most significant change, which increased by 32.3%

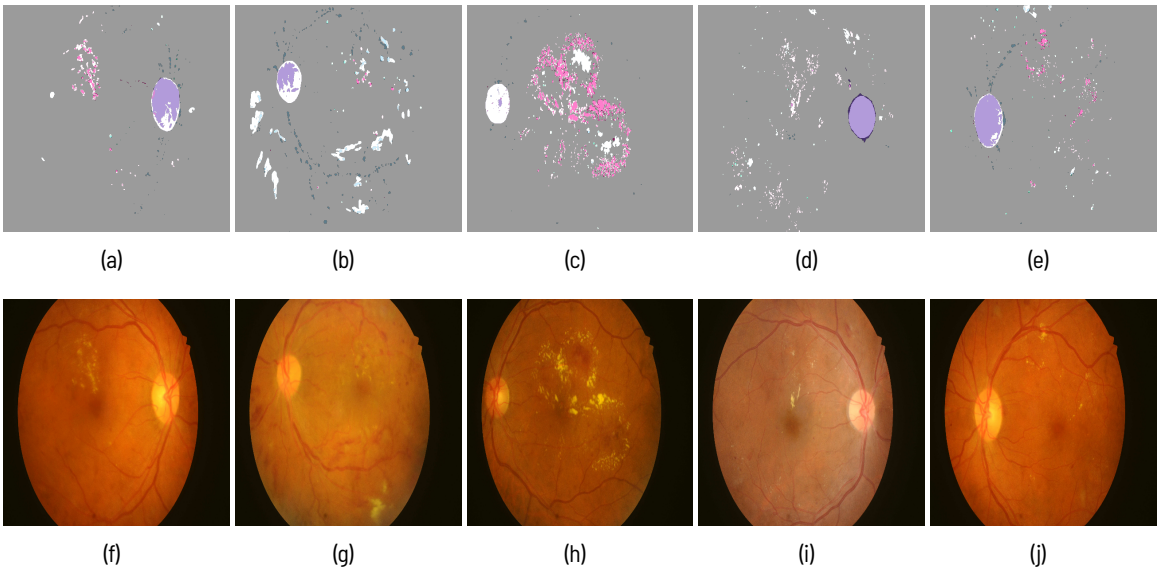


Fig. 7.23. Image (a) to (e) illustrate loss merging predictions as colored masks against the inverse of the ground truth mask colored in grey. We can see four classes to predict named **Haemorrhages**, **Hard Exudates**, **Microaneurysms** and the **Optic Disc**. The corresponding quantitative results are listed in table 7.6 as model no. 2. Image (f) to (j) depict the input features for each prediction above.

7.3. Computation Time

				Dataset size				
Config. No.	Dataset	Config type	Model count	100% (h)	64% (h)	32% (h)	Total (h)	Time per model (h)
1	Medaka	Baseline	18	4.1	2.5	1.8	8.4	0.46
2	Medaka	Discrete	360	119.0	80.0	46.0	245.0	0.68
3	Medaka	Continous	217	94.0	42.0	23.0	159.0	0.73
5	Skin lesion	Baseline	18	8.6	6.6	3.1	18.3	1.02
6	Skin lesion	Discrete	360	367.0	240.0	127.0	734.0	2.00
7	Skin lesion	Continous	146	121.0	65.8	37.8	224.6	1.50
9	IDRID	Baseline	6	2.5	-	-	2.5	0.40
10	IDRID	Discrete	120	150.0	-	-	150.0	1.30
11	IDRID	Continous	430	302.0	-	-	302.0	0.70
		Final Result	***1675***	***1168.2***	***436.9***	***238.7***	***1843.8***	***0.97***

Tab. 7.7. Table displays the final computation time for the configurations defined in table 5.2. The time-based merging configurations are skipped at this point as they were left out for the experiment.

Table 7.7 details the average computation time for each configuration, as outlined in table 5.2. The table includes the computation time for each dataset subset, the associated model count, and the average time per model. It is essential to highlight that some configurations were trained on different GPUs, complicating a straightforward comparison. Moreover, specific configurations, notably those related to performance-based continuous setups, might have been trained on multiple GPUs. Despite these factors, the data should give a good account of the final computation time for this experiment.

7.4. Ablation Results

This section explores the impact on performance when only a subset of the data is used to train the models. Given the high cost of computational resources for training ML models, it can be beneficial to get insights about the optimal loss combination from a smaller training set, thereby reducing the computational effort. However, this approach is only viable if the loss combinations remain consistent. The subsequent sections will present the varying results when the data is reduced from 100% to 64% and from 64% to 32% based on the formulation described in Section 5.5.5. The goal is to assess whether selecting suitable loss combinations undergoes significant changes when fewer data are utilized.

The results are presented for the Medaka Fish and Skin Lesion Dataset (SLD) exclusively, as for the IDRID, no training of subsets were designated due to its already small size. The results are presented qualitatively and quantitatively, where the former consists of the average of all loss-merge combination results for each dataset subsection presented as heatmaps in the appendix. The qualitative analysis aims to provide a more intuitive interpretation of how results change when just a subset of data is used, while the quantitative results consist of the ratio of IoU scores defined in equation 5.4, at different dataset subsets for all loss-merge combinations aiming to confirm the qualitative analysis results presented as heatmaps.

7.4.1. Medaka Fish

Fig. 7.24 depicts the outcomes derived from the algorithm described above for the MFD. The bar heights represent the average ratio for each combination type and predefined dataset range. If the averages exceed one, it implies a slight decrease in performance when reduced data is utilized for training. The averages are indicated above the bars, while the error bars' height represents the standard deviation.

Though the standard deviation for the 100 to 64% data range is relatively low, it substantially increases when considering ratios from 64 to 32% of the data, particularly for triple combination results which suggest a high data variance or potential outliers. We can refer to the six corresponding heatmaps displayed in the appendix as Fig. G.2 to gain a more intuitive understanding which additionally allows for a comprehension of the similarity between training with smaller dataset sizes and results using the entire dataset for training.

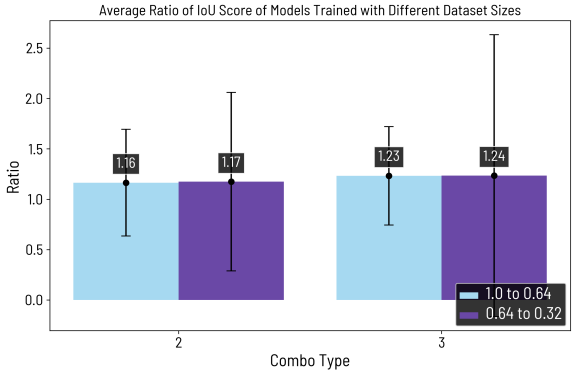


Fig. 7.24. Graph depicting the average ratio from models trained of different dataset sizes. The error bar indicates the standard deviation from the average ratio. If the ratio equals one, there would be no change when using a subsection of the data. If the ratio is above one, the performance decreases. The standard deviation indicates how dispersed the data is related to the mean.

7.4.2. Skin Lesion

Fig. 7.25 showcases the results for the SLD. Similar to the MFD, the averages here are slightly above one. However, the standard deviations for this dataset are considerably lower, pointing to a high level of proportionality.

Given the size of the SLD, even smaller sizes can yield valuable insights into optimal loss combinations. For additional analysis, the six corresponding heatmaps for this dataset are also displayed in the appendix as Fig. G.4.

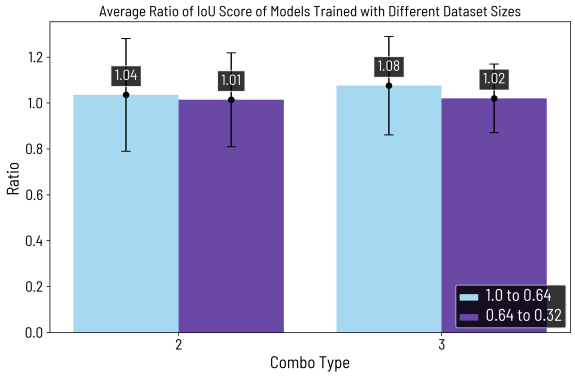


Fig. 7.25. Graph depicting the average ratio from models trained of different dataset sizes. The error bar indicates the standard deviation from the average ratio. If the ratio equals one, there would be no change when using a subsection of the data. If the ratio is above one, the performance decreases. The standard deviation indicates how dispersed the data is related to the mean.

8. Discussion

8.1. Restatement of the Research Problem

The research problem addressed in this thesis revolved around the challenges of semantic segmentation in CV, particularly in the context of highly unbalanced datasets of varying difficulty. The primary issue lies in optimizing loss functions, which is crucial in training effective segmentation models. Traditional approaches often rely on a single loss function, which might be optimal for all types of data and tasks and can lead to suboptimal performance and limit the potential of semantic segmentation models.

The research addressed the limitations of using a single loss function by developing and implementing a methodology that merges multiple loss functions, thereby enhancing segmentation performance. This process entailed several stages. First, a segmentation architecture based on the U-Net model was developed. Following this, baseline models were trained using a variety of distinct loss functions. An in-depth analysis of these loss functions was conducted to identify combinations that yielded promising results, leading to a series of merge strategies that were subsequently integrated into a fully automatic loss merging experimentation framework as a key output of this research.

Furthermore, the research extended to conducting extensive tests to gain insights into suitable loss combinations and merging strategies for optimal performance.

8.2. Key Findings and Interpretation

Chapter 7 presented a comprehensive range of results derived from three distinct datasets, each characterized by varying class counts and degrees of complexity. The Medaka Fish Dataset (MFD) may be considered the simplest, followed by the Skin Lesion Dataset (SLD), and ultimately, the Indian Diabetic Retinopathy Image Dataset (IDRID), which posed the most significant challenge. Several factors can influence the difficulty level of predictions for specific datasets, such as dataset size, class imbalance, or characteristics like the location, shape, and similarity of regions of interest. Related to these characteristics, predictions become more challenging as the variation for these properties increases. Among the datasets evaluated in this study, the IDRID shows significant variation in these attributes, in addition to its small size and high-class imbalance. The detailed properties of each dataset were introduced in the Datasets section.

The presented findings resulted by following a model production cycle defined in Section 2.3.5 and further described in Chapter 5, which includes dataset properties, loss combination, and merge strategy-specific

configurations, preprocessing, training, validation, and testing setups, monitoring functionalities and an ablation setup. Upon this setup described, the results were generated and analyzed.

		Baseline-Exceeding Count		Top Average (Specific)					Top Model				
				Merging			Baseline		Merging			Baseline	
	Difficulty	Double	Triple	IoU	Loss	Strategy	Iou	Loss	IoU	Loss	Strategy	Iou	Loss
Medaka Fish	+	23	18	0.695	FL,DL,BL	PBM	0.672	CE	0.772	CE,TL,BL	PBM	0.752	FL,HL
Skin Lesion	++	8	8	0.645	CE,TL	NWS	0.594	FL	0.672	CE,TL,HL	HARMONIC	0.667	CE
IDRID	+++	9	3	0.380	CE,DL	AVG	0.341	CE	0.414	CE,TL	MAX	0.367	CE

Tab. 8.1. Summary of quantitative results

Table 8.1 provides a summary of the quantitative results obtained for the three datasets that were evaluated. The column labeled »*Difficulty*« reflects the complexity, in ascending order, involved in training the dataset. The »*Baseline-Exceeding Count*« column indicates the number of times a model, using a specific combination of loss functions and merging techniques, surpasses the highest average baseline result in terms of the IoU score. The »*Top Average (Specific)*« column showcases the aggregate average of some promising loss-merge combinations and compares them with the top baseline results. The reasoning for these trainings was to reaffirm the validity by providing an equal iteration count as the baseline, which was not feasible for the entire range of combinations due to a lack of computational resources. These combinations are presented in the appendix in Section G.1.4, Section G.2.4 and Section G.3.4 and correspond to the fifth configurations for each dataset as defined in table 5.2. Lastly, the »*Top Model*« column highlights the highest individual scores achieved using merging techniques and baseline trainings.

8.2.1. Quantitative findings

The Quantitative Results section confirmed the correlation of dataset difficulty with the measured performance quantitatively. The MFD achieved the best overall results, followed by the SLD, with the poorest results attributed to the IDRID.

Loss combinations

With the primary goal to demonstrate whether combining losses improves performance over baseline models trained with a single loss, the analysis revealed that this was possible for a range of combinations, as presented in the corresponding »*Overall Performance*« subsection of each dataset section. The evidence suggests that double and triple loss combinations can generate top-performing results unattainable by the single loss baseline models.

The Medaka Fish Dataset (MFD) displayed the highest degree of performance improvement. The proposed framework introduced 23 distinct double and 18 triple combinations, each averaging higher performance than the top two baseline models in table 7.1. Table 8.1 lists this number in the »*Baseline-Exceeding Count*« section. Additionally, we can see a superior performance of the »*Top Average (Specific)*« column and »*Top Model*« column over the top baseline models displayed.

The Skin Lesion Dataset (SLD) performance was significantly lower in numbers of models that outperformed the baseline. Both double and triple combinations saw eight models on average showing superior performance than the top-performing baseline model no. 2, trained with the FL and displayed in 7.2. The »*Top Average (Specific)*« displays a pretty substantial increase using the combination of CE and DL along the arithmetic averaging strategy. On the other hand, the »*Top Model*« columns example presents a result which increased just lightly compared to the baseline.

The Indian Diabetic Retinopathy Image Dataset (IDRID) performance enhancement over the top-performing baseline model on the IDRID was achieved for nine models using a double loss and just three models using a triple loss combination even if individual models as illustrated in »*Top Average (Specific)*« and »*Top Model*« from the table 8.1 displayed quite superior performance.

These findings coincide with the dataset difficulty, implying that the more accessible a dataset is in size, distribution, shape, and location, the higher the number of loss combinations that can surpass the top-performing baseline models. It was observed further that double loss combinations outperformed triple loss combinations across all datasets, with combinations of CE, DL, and CE, TL showing the highest frequency of models which were capable of surpassing baseline models. For triple losses, the combination of FL, DL, HL exhibited the highest number of outperforming baseline models on average, followed by the combination of FL,TL, HL, and CE,DL, HL.

Merge strategies

Regarding merge strategies, the Normalized Weighted Sum (NWS) yielded the best results in outperforming top baseline models across all datasets, followed by arithmetic averaging (AVG) and the regular weighted Sum (WS). In addition to the discrete merging strategies, a continuous merging strategy was proposed, which included the search through a continuous hyperparameter space. While this strategy, on average, performed relatively poorly as low-performing models were included in the results, the hyperparameter analysis evaluated a clear tendency for higher values of the variable `pbm_I` to produce much better results. Looking at Fig. 4.3, we can see that the descending curve would reflect these findings, which is displayed in the image (a) and (b) where `pbm_I` is close or equals one. These findings are dataset specific but also specific to in which order different loss combinations are used. Nevertheless, they provide a good indicator for the user to narrow down the search space fast after some initial training on subsets of the data. For the parameter `pbm_alpha`, some stable results could be observed around values between two and six. A clear drop in the IoU metric is apparent for larger values. Looking at Fig. 4.3 again, we can see that if `pbm_alpha` is chosen too high, the weight contribution for each loss combination is getting very low, which seems to produce a much lower performance.

8.2.2. Qualitative findings

The Qualitative Results section visually showcased some top-performing models. Visual results generally can facilitate debugging but also provide an intuitive understanding of how metrics such as PPV, TPR, or others impact the performance. The qualitative section highlighted differences in these metrics and addition-

ally referred to the subsection »*PPV vs. TPR*« in the appendix, summarizing the prevalence of PPV versus TPR quantitatively. The reference indicated which metrics achieved better results for each loss combination. While averaging strategies like harmonic averaging (HARMONIC) and arithmetic averaging (AVG) typically produced results with higher precision, extreme point strategies such as the min strategy (MIN) and the max strategy (MAX) resulted in models with higher recall.

Even if these results seem to be pretty straightforward, it is crucial to acknowledge that the MIN and MAX strategies performed relatively poorly in terms of IoU which encourages us to question and further investigate the results of figures G.1, G.3 and G.5 which reaffirms the necessity of highlighting different aspects of a model's performance to avoid producing misleading results.

The qualitative analysis also demonstrated the performance increase of underrepresented classes present in the IDRID. It was shown that some classes increased their performance by more than 30% over the top-performing baseline model.

8.2.3. Computation time evaluation

The Computation Time section presented the average training time per model, signifying a high computational effort required to train the proposed framework. It was observed that the effort to train a single model at least doubled when using a loss combination. This finding highlights the importance of techniques that can reduce computation time, leading to the introduction of the ablation study, where the potential of a dataset subset to yield similar results was analyzed.

8.2.4. Ablation study implications

The Ablation Setup section introduced a method designed to analyze the similarity in performance of a set of results from different data subsets. Intuitively, this method compares the overall results of each loss and merge combination across different data subsets. It measures the average and standard deviation of ratios from all loss and merge combinations, where the average ratio denotes whether the performance has increased or decreased when using fewer data. Meanwhile, the standard deviation of these ratios signifies how similar the results are across all observed combinations. Heatmaps, which visually present the calculated results indicating further a high similarity across the subsets, have been created for three data subsets and are included in the appendix. The ablation study was applied exclusively to the Medaka Fish and Skin Lesion Dataset (SLD) and revealed in Section 7.4 a high similarity across models trained on subsets of data specifically for the SLD.

9. Conclusion

This study introduced a unified framework for loss merging to enhance semantic segmentation. Central to this framework are eight unique loss merging strategies that merge six loss functions in various double and triple combinations. The framework, integrated with the U-Net architecture, offers multiple configurations for training diverse models, which were precisely evaluated against baseline models utilizing single-loss functions.

An exhaustive analysis of the results from several perspectives highlighted the potential to boost overall performance for all datasets studied by merging specific types of losses. It was demonstrated that an extensive array of loss combinations and merge strategies surpassed the performance of models trained with single losses. This was particularly evident in the promising results of strategies employing the normalized weighted sum (NWS) and averaging (AVG). Additionally, exceptional performance was observed through a distinct, performance-based method when hyperparameters were calibrated correctly.

However, the increased computational resource demand serves as a notable trade-off, as models trained using loss combinations necessitate up to twice the computational time relative to baseline training with a single loss function in addition to the set of combinations which can be high if many losses and merge strategies are used.

Despite this notable disadvantage, the ablation study showed that the proposed framework could be efficiently deployed on a subset of data, allowing researchers to achieve comparable insights with less training data, thus effectively compromising cost and insight acquisition.

Though the framework was applied to six loss functions and eight merging strategies, its potential is not restricted to these parameters. It is adjustable to incorporate an arbitrary number of functions or strategies, thereby paving the way for formulating novel types of losses in future studies.

Additional avenues for future exploration include extending the framework to further losses and datasets, testing a more refined data subset with even fewer samples or using only a fraction of the number of epochs, to extend the ablation study, or adjusting the model to adapt different batch sizes. Additionally, the time-based merging strategy might be worth evaluating, as presented in Section 4.2.2, which was excluded due to its high computational demand.

Such proposed modifications or extended research might increase the understanding of this area and enhance the practical application of the presented framework.

A. List of abbreviations

ASP	Adaptive Spatial Pooling	101
AI	Artificial Intelligence	ix
AR	Augmented Reality	2
AB	Ada Boost	7
AL	Active Learning	x
Acc	Accuracy	93
BCE	Binary Cross Entropy loss	15
BAIS	Boundary-Aware Instance Segmentation	98
BDV	Big data Visualization	11
BL	Boundary Loss	ix
CNN	Convolutional Neural Network	5
CE	Cross-Entropy Loss	ix
CEM	Context Encoding Module	x
CCE	Categorical Cross Entropy Loss	15
CV	Computer Vision	ix
CT	Classification Tree	7
CS	Customer Segregation	11
CAM	Class Activation Maps	106
DSC	Dice Similarity Coefficient	16
DL	Dice Loss	ix
DMT	Discrete Morse Theory	ix
DNN	Deep Neural Network	33
DA	Domain Adaption	109
DRL	Deep Reinforcement Learning	114
DQN	Deep Q-Network	114
EDA	Encoder-Decoder Architecture	29
ERM	Empirical Risk Minimization	6
FCN	Fully Convolutional Network	97
FL	Focal Loss	ix
FFN	Feed Forward Network	8
FE	Feature Elicitation	11
GB	Gradient Boost	7
GAN	Generative Adversarial Network	110
GDL	Generalized Dice Loss	17
HRNet	High Resolution Network	x
HD	Hausdorf Distance	18

HL Hausdorff Loss 18
ISIC International Skin Imaging Collaboration 48
IDRID Indian Diabetic Retinopathy Image Dataset x
IoU Intersection over Union 26
JI Jaccard index 26
KNN K-nearest Neighbors 7
LR Linear Regression 6
LogR Logistic Regression 7
ML Machine Learning ii
MC Meaningful Compression 11
MR Magnetic Resonance 110
MFD Medaka Fish Dataset x
NLP Natural Language Processing 5
NN Neural Network vii
NPV Negative predictive value 26
OMN Object Mask Network 98
PPV Positive Predictive Value 25
RPN Region Proposal Network 99
RoI Region of Interest 99
RSI Remote Sensing Image 101
RC Robot Control 5
RF Random Forests 7
RT Regression Trees 7
RS Recommender System 11
RL Reinforcement Learning ix
SPP Spatial Pyramid Pooling 29
SR Speech Recognition 5
SVM Support Vector Machine 8
SD Structure Discovery 11
SIFA Synergistic Image and Feature Adaption 110
SDI Søerensen-Dice index 27
SLD Skin Lesion Dataset x
SL Supervised Learning 5
TCE Tilted cross-entropy 34
TERM Tilted empirical risk minimization 34
TL Tversky Loss 17
TBM Tree Based Methods 7
TM Targetet Marketing 11
TPR True Positive Rate 25
TNR True Negative Rate 26
UL Unsupervised Learning ix
wandb Weights and Biases 48

B. Mathematical Formulations

B.1. Softmax

We can use the softmax function to transfer the model scores to probabilities for a multi-class classification problem. For further reference on the softmax implementation, see [86] as:

$$\sigma_i(\hat{y}) := \frac{\exp(\hat{y}_i)}{\sum_{j=1}^{d} \exp(\hat{y}_j)}, 1 \leq i \leq n \tag{B.1}$$

where d is the number of classes and i indicating the current pixel. The function applies a pointwise exponential and divides it by the sum over the exponential as a normalization to ensure that all components sum up to one. Fig. B.1 depicts the transformation of an output score to an output with probabilities using the softmax function.

Fig. B.1. (a-d) Output from CNN with scores transferred to probabilities (e-h) with equation B.1

C. Metric Comparison

It can be challenging to determine the best metric, as users may prioritize different aspects of the classification results depending on the task. Semantic segmentation often involves regions of interest much smaller than the background region. When valid for the entire dataset, the data is considered unbalanced. Therefore, choosing the right metric that accurately reflects changes in unstable situations is crucial.

Table C.1 compares a situation where one pixel is misclassified as a false negative for both balanced and unbalanced labels. Table C.2 compares a similar situation focusing on changing a false positive misclassified pixel. Both tables show that the IoU and DSC metrics reflect the performance change as expected for balanced and imbalanced data. In contrast, Accuracy (Acc) does not change as it includes true negatives in its calculation, providing a very high misleading score for both cases.

False negative misclassification

Balanced	PPV	TPR	TNR	NPV	IoU	Dice	ACC	TP	FP	FN	TN
TP(11) : TN(13)	1.000	0.917	1.000	0.929	0.917	0.957	0.960	11	0	1	13
TP(12) : TN(13)	1.000	1.000	1.000	1.000	1.000	1.000	1.000	12	0	0	13
Difference	0.00%	8.33%	0.00%	7.14%	8.33%	4.35%	4.00%				

Imbalanced	PPV	TPR	TNR	NPV	IoU	Dice	ACC	TP	FP	FN	TN
TP(2) : TN(22)	1.000	0.667	1.000	0.957	0.667	0.800	0.960	2	0	1	22
TP(3) : TN(22)	1.000	1.000	1.000	1.000	1.000	1.000	1.000	3	0	0	22
Difference	0.00%	33.33%	0.00%	4.35%	33.33%	20.00%	4.00%				

Tab. C.1. Tables illustrating the score difference in percent by misclassifying one pixel as false negative for a balanced region at the top, and an imbalanced situation at the bottom. We can see that the IoU and DSC properly reflect the change by a much larger difference percentage in comparison to accuracy. Accuracy does not change as it includes true negatives in its calculation providing in both cases a very high misleading score. As the misclassified pixel was a false negative, precision stays unaffected while recall decreases by 33.33%

False positive misclassification

Balanced	PPV	TPR	TNR	NPV	IoU	Dice	ACC	TP	FP	FN	TN
TP(12) : TN(12)	0.923	1.000	0.923	1.000	0.923	0.960	0.960	12	1	0	12
TP(12) : TN(13)	1.000	1.000	1.000	1.000	1.000	1.000	1.000	12	0	0	13
Difference	7.69%	0.00%	7.69%	0.00%	7.69%	4.00%	4.00%				

Imbalanced	PPV	TPR	TNR	NPV	IoU	Dice	ACC	TP	FP	FN	TN
TP(3) : TN(21)	0.750	1.000	0.955	1.000	0.750	0.857	0.960	3	1	0	21
TP(3) : TN(22)	1.000	1.000	1.000	1.000	1.000	1.000	1.000	3	0	0	22
Difference	25.00%	0.00%	4.55%	0.00%	25.00%	14.29%	4.00%				

Tab. C.2. Similar as the tables above do these tables illustrate the score difference in percent by misclassifying one pixel as false positive for a balanced label at the top, and an imbalanced label at the bottom. We can see again that the IoU and DSC properly reflect the change by a much larger difference percentage in comparison to accuracy. As the misclassified pixel was a true positive, recall here stays unaffected while precision decreases by 25.00%

D. Extended Literature Review

This extended chapter reviews techniques to enhance semantic segmentation models' performance. The following summaries were created during the project's goal-finding process aiming to offer an in-depth perspective on the topic. The techniques discussed in this chapter are being applied to various tasks, but the focus of the cited papers is specifically on their use in semantic segmentation. The methods are categorized to facilitate understanding and comparison. Each technique is introduced with a general overview or survey, followed by a chronological presentation of specific approaches, with the most recent techniques being presented first.

- Data Augmentation
- Multi-scale Processing
- Ensemble Learning
- Transfer Learning
- Weakly-supervised Learning
- Active Learning
- Semi-supervised Learning
- Domain Adaption
- Post-processing
- Multi-task Learning
- Attention Mechanisms
- Reinforcement Learning

D.1. Data Augmentation

»*A survey on image data augmentation for deep learning*« [231] from 2019 provides an extensive study for data augmentation. *Data augmentation* is a technique used to improve deep learning models by enhancing the size and quality of training datasets. It invovles a variety of techniques, including geometric transformations [251], color space augmentations, and neural style transfer [231]. The paper also covers other aspects of data augmentation, such as test-time augmentation, resolution impact, and curriculum learning. Data augmentation aims to increase model performance and expand limited datasets to be more similar to large datasets.

»*A Simple Baseline for Semi-Supervised Semantic Segmentation With Strong Data Augmentation*« [298] from 2021 presents a semi-supervised model with strong data augmentation. Their approach is specific for semi-supervised models where many unlabeled training samples are added to the labeled dataset. To balance

the disadvantage of batch normalization [1] for semi-supervised models (distribution mismatch), they propose a distribution specific batch normalization and forward strongly augmented and weakly-augmented data with different batch statistics. Their method achieves state-of-the-art results in the semi-supervised settings on CityScapes [60] and Pascal VOC [110] datasets.

»*Optimizing Data Augmentation for Semantic Segmentation on Small-Scale Dataset*« [165] from 2019 aims to optimize data augmentation methods for semantic segmentation. The authors summarize important methods in two groups and call them global and local data augmentation. Global data augmentation refers to augmenting images, while local augmentation is specific to objects within the image. They combine different data augmentation techniques and achieve the best results with compression, cropping, and local shift. The authors claim to improve the mean IoU from 73.3% to 91.3%. The model was trained on a small custom dataset built from scratch for sheep body recognition and segmentation.

»*Improving Data Augmentation for Medical Image Segmentation*« [74] from 2018, evaluates an augmentation method for medical images, called »*mixup*« where each sample label pair is a linear combination of two sample label pairs calculated as:

$$x_{mixup} = \lambda x_i + (1 - \lambda) x_j \tag{D.1}$$

$$y_{mixup} = \lambda y_i + (1 - \lambda) y_j \tag{D.2}$$

where $\lambda \; in [0, 1]$ and drawn from $\lambda \sim \beta(\alpha, \alpha)$ for $\alpha \in (0, \infty)$ according to a Beta distribution [2]. This method has been shown to improve performance on several machine-learning tasks. The authors additionally implemented a new method called »*mixmatch*«, a variant from »*mixup*« by considering class prevalence. Both methods are then used to train the BraTS [16] dataset, outperforming training without these methods.

»*Large Scale Labelled Video Data Augmentation for Semantic Segmentation in Driving Scenarios*« [38] from 2017, proposes a simplified version of the label propagation algorithm presented by [15]. *Label propagation* is an augmentation technique that labels unlabeled data points based on their similarity to labeled data points. Thus, label propagation is commonly used in semi-supervised learning, where a dataset contains a lot of unlabeled and few labeled data points. The authors apply their algorithm on the CityScapes [60] and CamVid [35] dataset observing a significant increase in performance.

[1] Batch normalization normalizes activations in hidden layers of deep neural networks and is supposed to improve accuracy and speed up training. While networks without batch normalization tend to diverge with significant learning rates, batch normalization can avoid this problem as the activations are corrected to be zero-mean and of unit standard deviation, enabling more significant gradient steps, yielding faster convergence and helping to bypass local minima [25]

[2] The Beta distribution is a continuous probability distribution defined on the interval $[0, 1]$ and parameterized by two positive shape parameters, α, and β. It helps model random variables with a fixed range related to proportions, percentages, or probabilities. The Beta distribution is a member of the exponential family and is conjugate to the Bernoulli and binomial distributions, making it convenient for Bayesian inference.[82][111]

D.2. Improved Feature Extraction

Feature extraction is a process used to reduce the dimensionality of data and extract meaningful features from it. It differs from feature selection, which involves identifying a subset of the original dimensions containing the most information and discarding the rest. In contrast, feature extraction involves finding a new set of dimensions that are combinations of the original dimensions and are more effective at representing the data [10].

In traditional machine learning techniques, feature extraction is typically performed using techniques such as principal component analysis [3] or linear discriminant analysis [19]. However, feature extraction is usually performed automatically using convolutional layers in deep learning, particularly with CNNs. Early layers in a CNNs might extract low-level features such as edges and corners, while later layers extract higher-level features such as objects or patterns.[109]

Enhancing feature extraction in the context of semantic segmentation means improving the ability of the convolutional layers in the deep learning architecture to extract relevant and useful features for the current task. The following paragraphs discuss various approaches to designing the CNNs architecture and filters to optimize the quality and effectiveness of the extracted features to improve the model's performance.

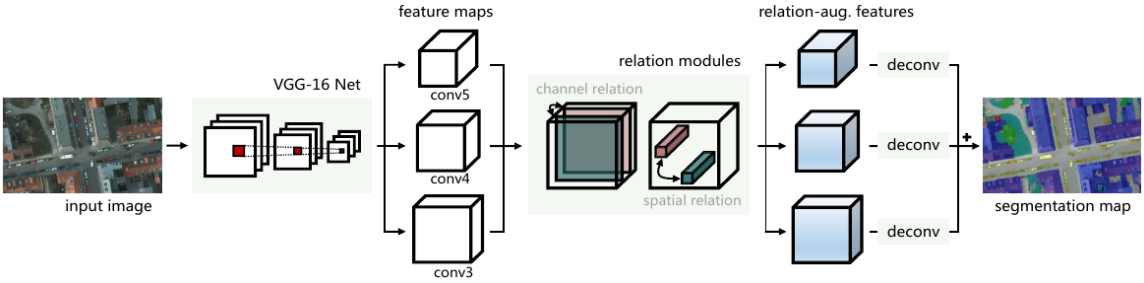


Fig. D.1. Convolutional Neural Network (CNN) architecture featuring a channel and spatial relation module, as shown in the image from [180].

»*Relation Matters: Relational Context-Aware Fully Convolutional Network for Semantic Segmentation of High-Resolution Aerial Images*« [180] from 2020 proposes an innovative approach to enhance feature extraction for semantic segmentation of aerial images using a CNN. See Fig. D.1 for further information on the architecture. Their goal is to capture both long-range spatial relationships and channel-wise information, which helps to address challenges such as visual ambiguities[3] and appearance variations. The effectiveness of this method is demonstrated through ablation studies, which show that the network can perform both channel and spatial reasoning and effectively process relational context. The results of this study suggest that this approach may be a valuable tool for improving semantic segmentation in aerial imagery.

»*Context Encoding for Semantic Segmentation*« [301] from 2018 introduces the Context Encoding Module (CEM) for semantic segmentation using a Fully Convolutional Network (FCN). The module captures the semantic context of scenes and selectively highlights class-dependent feature maps, significantly improving

[3] Visual ambiguities occur when visual information is incomplete, noisy or has multiple interpretations.[272][28][127]

semantic segmentation results with only marginal extra computation cost. See Fig. D.2 for further details on the architecture. The authors achieved new state-of-the-art results on PASCAL-Context [79] and PASCAL VOC 2012 [78] datasets and also explored the module's impact on improving feature representation for image classification on CIFAR-10 [142] dataset.

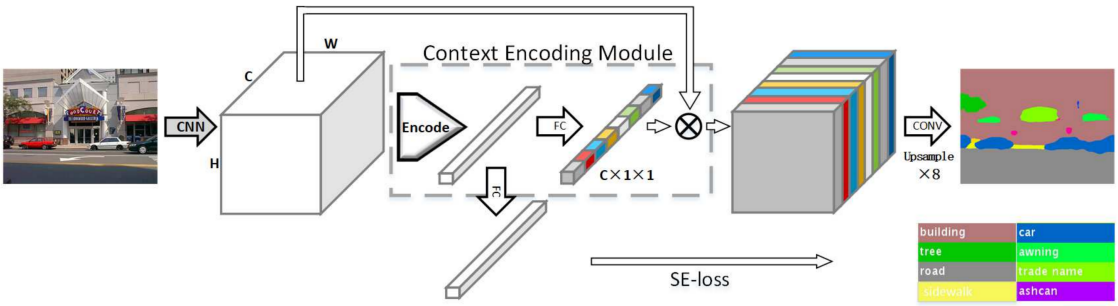


Fig. D.2. Modified FCN with Context Encoding Module (CEM): The class-dependent feature maps are shown within the CEM prior to calculating the SE loss. The SE loss is employed to regularize the training of the CEM, capturing a broader context than the standard per-pixel loss. Image from [301].

»*Shape-aware Instance Segmentation*« [105] from 2016 introduces a new approach to instance-level semantic segmentation, which is the task of jointly detecting, segmenting, and classifying individual objects in an image.[98][27]. The presented methods use bounding boxes as candidate objects and directly predict a binary mask within each box. As this makes them sensitive to errors in the object proposal generation process, such as too small or shifted boxes, the authors propose a new object segment representation based on the distance transform of the object masks. This representation allows robust prediction of masks beyond the bounding boxes' scope. They subsequently design an Object Mask Network (OMN) to infer and decode this representation into the final binary object mask. The OMN is integrated into a multi-task network cascade framework to learn the Boundary-Aware Instance Segmentation (BAIS) network end-to-end. Fig. D.3 contains further information about the architecture used. Experiments on the PASCAL VOC 2012 [78] and CityScapes [60] datasets show the benefits of the proposed approach, outperforming the state-of-the-art in object proposal generation[4] and instance segmentation.

[4]Object proposal generation is a task in CV that aims to generate candidate regions that are likely to contain objects of interest in images. It can be used as a preprocessing step for object recognition models to decrease the search space, improve efficiency and accuracy.[230][281][66]

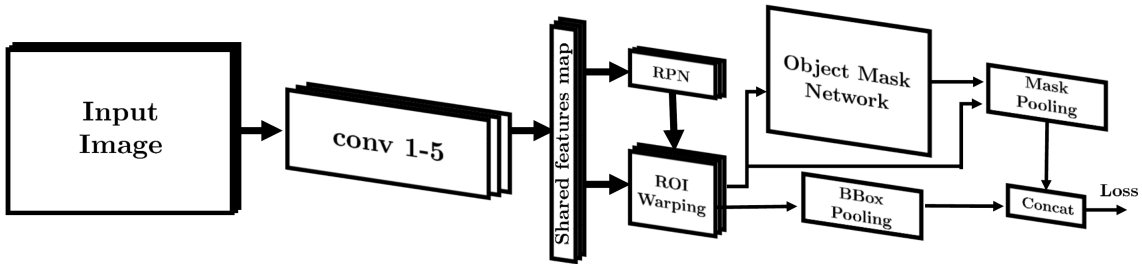


Fig. D.3. The shape-aware instance segmentation network has an input image go through convolutional layers and a Region Proposal Network (RPN)[205] for bounding box proposals, then Region of Interest (RoI) warping and OMN for binary mask generation, followed by feature extraction for classification. The 5-stage Boundary-Aware Instance Segmentation (BAIS) network adds two more stages (OMN and classification module) to the initial three stages. The model uses a multi-task loss during training to encode errors in the bounding box, segmentation, and classification. Image from [205].

D.3. Multi-scale Processing

Multi-scale processing is a technique to extract information from data at multiple scales. It involves processing the data at different resolutions or levels of detail to capture the complete structural information present in the data. The paper »*A Review on Multiscale-Deep-Learning Applications*«[76] from 2022 divides multi-scale DNNapproaches in two categories: multi-scale feature learning and multi-scale feature fusion where the former aims to extract feature maps by examining kernels over several sizes to collect a more extensive range of relevant features and the latter uses features with different resolutions to find patterns over short and long distances. These methods can be implemented as different architectures described in Fig. D.4.

»*Multi-Scale Self-Guided Attention for Medical Image Segmentation*« [238] from 2021 presents a new approach to overcome the limitations of existing CNN models related to redundant use of information and the lack of efficiently modeling long-range feature dependencies resulting in non-optimal discriminative feature representations[5]. The paper describes a multi-scale guided attention network where four feature maps are extracted from 4 ResNet-101 at multiple scales and subsequently fed to a guided attention module with the benefit of removing noisy areas and helping the network emphasize the regions more relevant to semantic classes. Fig. D.5 further explains the concept of this approach. The model was evaluated on three datasets: abdominal organs, cardiovascular structures, and brain tumors. The results of the ablation experiments support the importance of the attention modules in the proposed architecture. The model outperformed state-of-the-art segmentation networks with improved accuracy and reduced standard deviation.

[5] Non-optimal discriminative feature representations are feature representations that are not correctly suited for distinguishing different classes or categories[54]. They may contain noise or irrelevant information that reduces the accuracy or efficiency of pattern recognition [96].

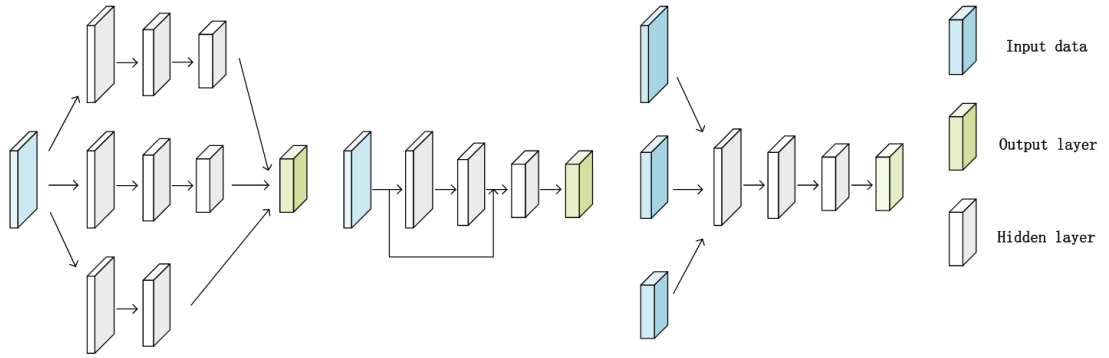


Fig. D.4. Summary of different multi-scale deep learning architectures described in [278]. The left part shows a multi-column network where the input data is propagated into multiple parallel operating branches and concatenated as the final output. The middle part is the skip-net which merges low-level and high-level features as the output, whereas the right part is called multi-scale input, which resizes images to different scales before feeding them to the network. Image from [278].

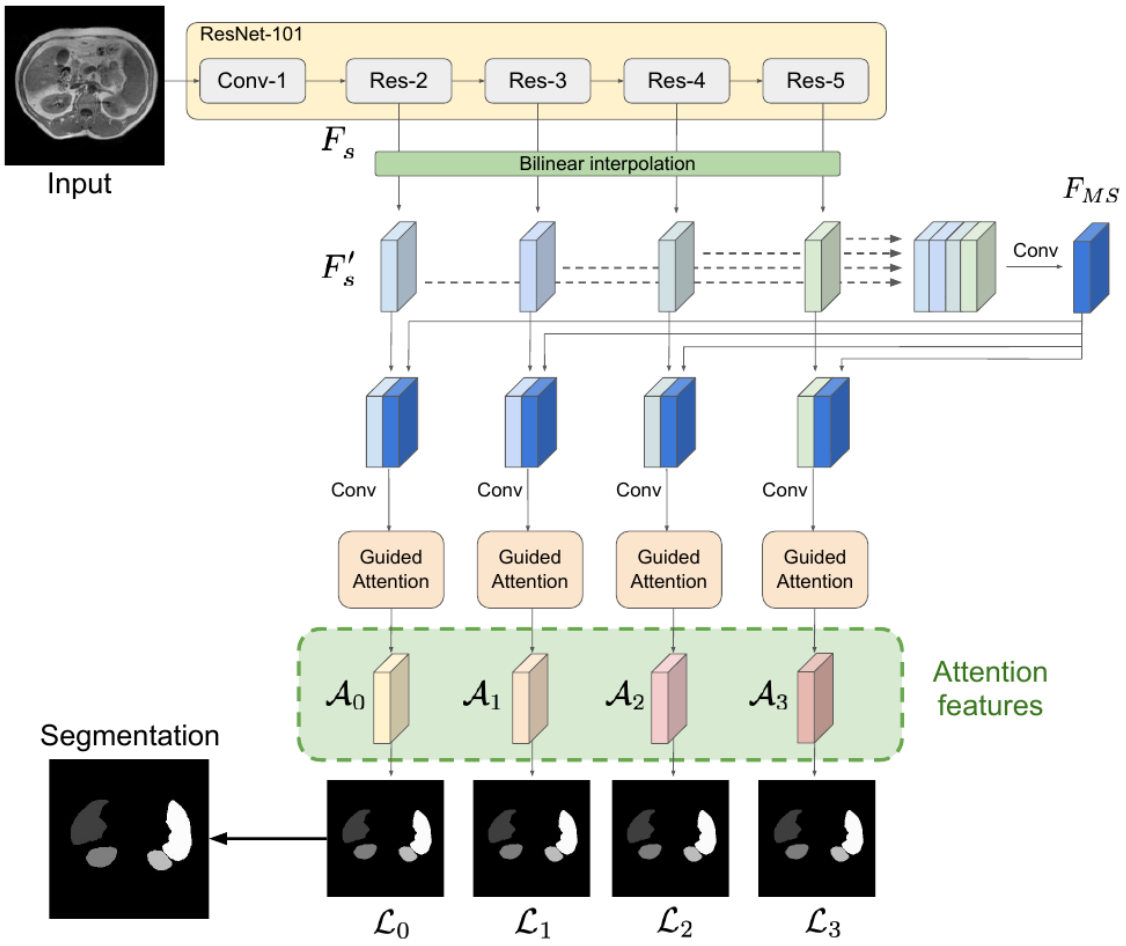


Fig. D.5. Architecture of the multi-scale guided attention network. Image from [238].

»*Multi-Scale Context Aggregation for Semantic Segmentation of Remote Sensing Images*« [304] from 2020 presents a novel architecture for semantic segmentation of Remote Sensing Images (RSIs) proposing a network called HRNet. The HRNet is designed to produce high-resolution features without the decoding stage and overcome the limitations of conventional encoder-decoder-based CNNs, which results in loss of localization accuracy and preservation of spatial details. The proposed architecture enhances the low-to-high features extracted from different branches to strengthen the embedding of scale-related contextual information. The low-resolution features are utilized to model long-term spatial correlations, while high-resolution branches are enhanced with an Adaptive Spatial Pooling (ASP) module to aggregate more local contexts. See Fig. D.6 to view a schematic of the proposed model. The resulting architecture can exploit spatial context at both global and local levels[6], achieving state-of-the-art performance on two RSI datasets.

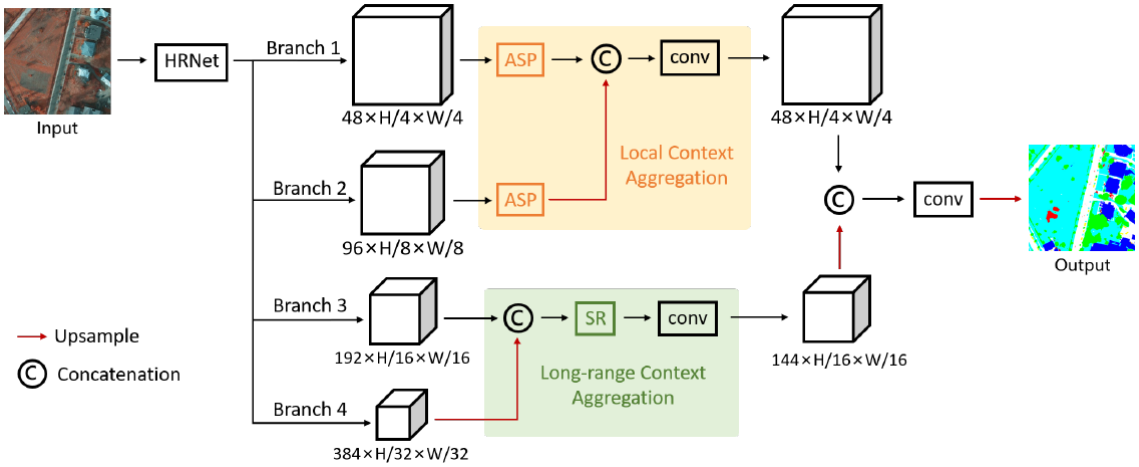


Fig. D.6. HRNet proposed by [304]

[6] Local spatial context is related to information within a small region or neighborhood of the pixels, whereas global spatial context refers to the information across the whole image.

D.4. Ensemble Learning

The paper »*A survey on ensemble learning*« [70] presents an excellent review of the field of ensemble learning, a machine learning technique that combines multiple models to make predictions. The paper begins by introducing the concept of ensemble learning and its history, tracing its roots back to the 1960s and 1970s. The author then categorizes ensemble methods, including bagging, boosting, stacking, and hybrid.

Bagging, or bootstrap aggregating, is a technique that trains multiple models on different subsets of the training data generated using bootstrapping and combines the predictions of these models to make a final prediction. Conversely, Boosting trains models sequentially, giving more weight to samples than previous models in the sequence misclassified. Stacking involves training a meta-model on the predictions of multiple base models, and hybrid methods are combinations of bagging and boosting.

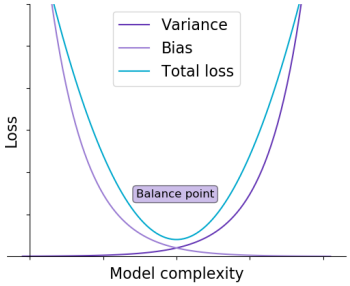


Fig. D.7. Relationship between complexity and learning curve [70]

The impact of those techniques on the relationship between the learning curve and model complexity can be summarized as follows:

Bagging reduces variance and can lead to smoother learning curves as the models are trained on different subsets of the data. The model complexity remains the same.

Boosting reduces bias and can lead to a faster decrease in training error in the beginning, but it can also lead to overfitting if the model complexity is too high. The model complexity increases with each iteration as the model focuses on the samples that were misclassified.

Stacking combines the predictions of multiple models, leading to a reduction in variance, and can improve the model's overall performance.

Hybrid Methods affect the relationship between the learning curve and model complexity depending on the specific combination of bagging and boosting. However, the overall effect is reduced variance and bias, which can result in smoother and faster convergence of the learning curve.

The assessment and comparison of ensemble methods are also covered in the study, along with several techniques for choosing the optimal ensemble and performance indicators like accuracy and F1-score. The author also emphasizes the difficulties and restrictions of ensemble learning, such as the danger of overfitting and the difficulty in choosing suitable base models.

The paper's discussion outlines the importance of ensemble learning for numerous applications in fields such as computer vision, natural language processing, and bioinformatics.

»*A Systematic Evaluation of Ensemble Learning Methods for Fine-Grained Semantic Segmentation of Tuberculosis-Consistent Lesions in Chest Radiographs*« [201] from 2022 evaluates the performance of different ensemble

learning methods for fine-grained semantic segmentation of tuberculosis-consistent lesions in chest radiographs. The study compares the performance of several ensemble methods, including bagging, boosting, and stacking, on a dataset of chest radiographs. The study results show that the stacking method performed better than the other ensemble methods, providing improved performance over the individual models regarding accuracy and computation time. The paper concludes that stacking is a promising approach for fine-grained semantic segmentation in chest radiographs and highlights the importance of considering different ensemble methods for such tasks. Fig. D.8 describes the setup for this method.

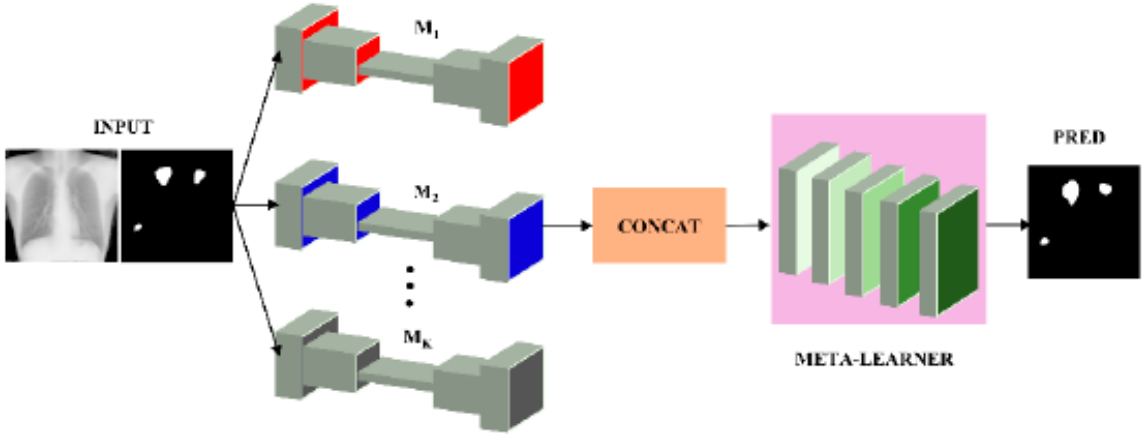


Fig. D.8. Stacking ensemble using the top-K models $M_1, M_2, \cdots M_k$ as input for a fully-convolutional meta-learner. After the top-K models are initialized with their trained weights, the features from the penultimate layers are extracted and concatenated. Subsequently, the meta learner is used for second-level learning [7]. Image from [201].

»*Ensemble Knowledge Transfer for Semantic Segmentation*« [184] from 2018 presents a method for transferring knowledge from existing datasets to a new target dataset for aerial semantic segmentation. The authors introduce AeroScapes, a new dataset of 3269 aerial images annotated with semantic segmentation. To transfer knowledge, the authors use transfer learning techniques to train multiple models for aerial segmentation by fine-tuning through each source domain. See Fig. D.9. The models are then combined into an ensemble. This results in significant improvements (8.12%) in performance over standard baselines.

[7] Second-level learning is a machine learning technique that involves training a model to learn how to select predictions of other models.[33][287]

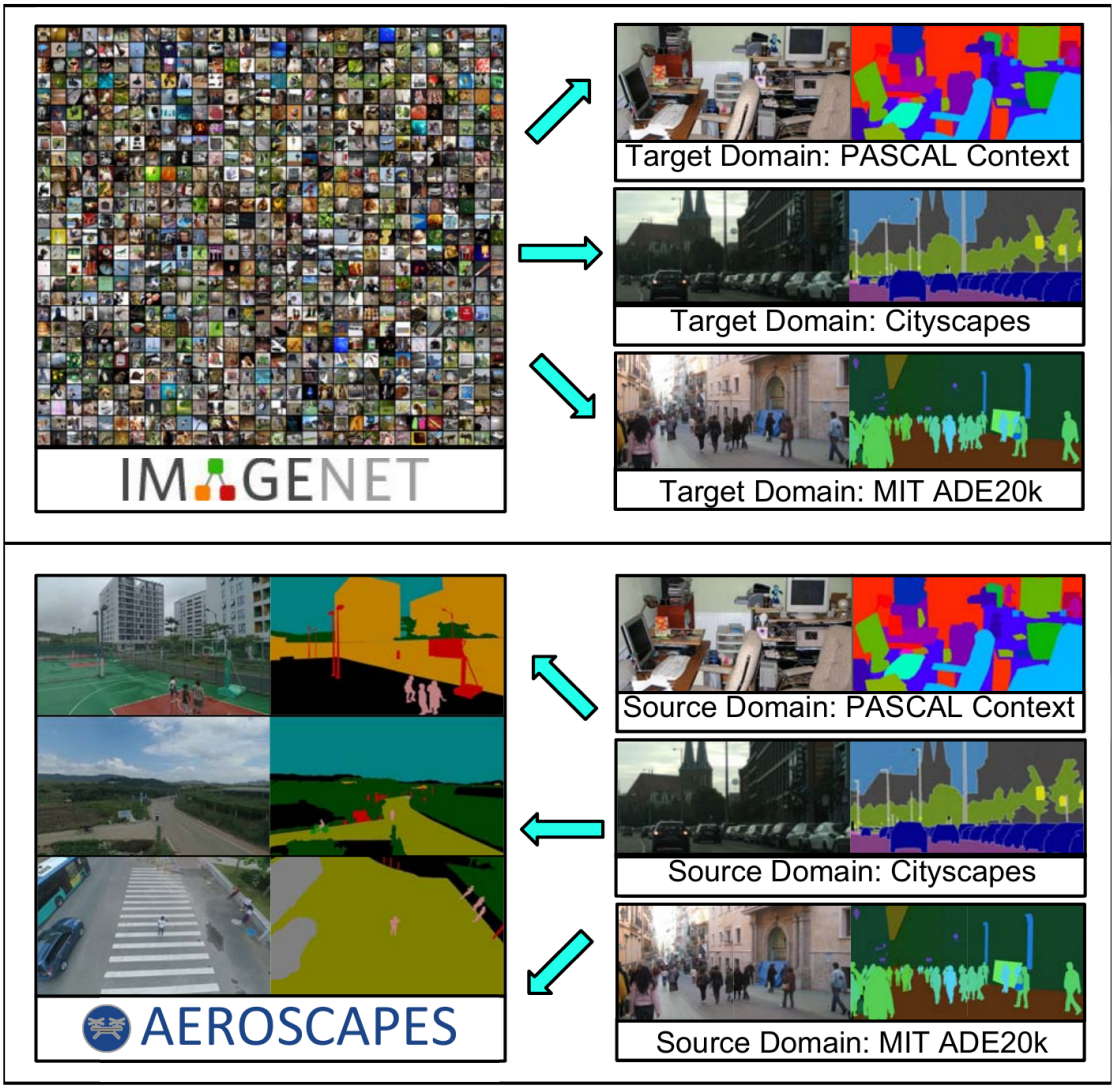


Fig. D.9. Instead of taking advantage of multi-target knowledge transfer (top row), the paper introduces a new technique where specific source domains are fine-tuned to make predictions to a single target domain (bottom row). The concept is called multi-source knowledge transfer. Image from [184].

D.5. Transfer Learning

The paper »*A Comprehensive Survey on Transfer Learning*« [317] from 2019 provides an excellent survey on transfer learning. The authors define transfer learning as improving the performance of target learning on target domains by transferring the knowledge contained in different but related source domains. Note that the two domains should be somehow related to each other. For example, if we train a model to recognize images of cats, we can reuse that model to recognize images of dogs. Even if cats and dogs are not within the same domain, they are related in terms of being both mammals with four legs and fur and generally with

similar features. If the source domain is entirely different from the target domain, successfully reusing a model may be more challenging. An example could be a model trained to recognize text sentiment in English and then reused to recognize text sentiment in a different language. The features might be completely different, making the transfer of knowledge much more challenging. The following section will provide more information on transfer learning related to increased semantic segmentation performance.

»*Transfer Learning via Parameter Regularization for Medical Image Segmentation*« [215] from 2021 presents a new method which aims to reduce »*catastrophic forgetting*« [172] which is a tendency of a network to forget previously learned information when learning new information thoroughly. The authors state that target models are not expected to handle the source task and therefore need to remember it. They propose a transfer learning method that addresses this »*forgetting problem*« by using the parameters of the source network as a regularization instead of an initialization term. The learned parameters of the target model are penalized if they deviate too far from the source network, which helps to overcome the catastrophic forgetting problem and take advantage of the knowledge obtained by the source model. This transfer learning strategy is applied to COVID-19 opacity segmentation and improves the segmentation of coronavirus lesions in chest CT scans.

»*Distant Domain Transfer Learning for Medical Imaging*« [185] from 2021 presents a transfer learning method that obtains knowledge from multiple unlabeled source domains to improve accuracy for training on a labeled target domain to classify CT scans of COVID-19 patients. The source domains do not have to be related to the target domain as in common transfer learning methods. The method is inspired by the fact that even if domains seem unrelated on the instance level, there might still be common properties on the feature level. The authors propose a three-staged process defined as

$$\min_{\theta_E,\theta_D\Theta_C} L = L_R + L_D + L_C \tag{D.3}$$

where L_R consists of a reconstruction loss obtained from an autoencoder with multiple sources and one target domain. The goal is to create robust, unsupervised features that are then forwarded to obtain L_D, which is the domain loss responsible for closing the distance between the source and target domains, necessary to avoid the distribution mismatch between the source and target domains and provide robust, domain-invariant[8] features. The method used in this step is called »*maximum mean discrepancy*« [260]. The last part of D.3 is the classification loss L_C, which is obtained by a fully connected layer added to the encoder of the network. The loss takes samples and output labels from the target domain and complements them as the final component to the overall objective L. Even if the results achieve excellent performance in COVID-19 and pneumonia diagnosis, the method has some drawbacks: case-specificity, complicated source domain selection, and computationally expensive calculation of L_D.

D.6. Weakly-supervised Learning

As mentioned in Section 2.2.2, weakly-supervised learning only uses partial information about the label. The paper »*A brief introduction to weakly supervised learning*« [316] from 2017 provides a detailed introduction of

[8] Domain-invariant features are features that remain consistent across different domains or datasets. In machine learning, a *domain* is defined as a specific distribution of data, such as images of humans versus images of animals.

weakly supervised learning and further specifies weakly supervised learning in »*incomplete supervision*« which only uses a subset of training data and labels, »*inexact supervision*« where labels are coarse-grained[9], and »*inaccurate supervision*« where the given labels are not always ground-truth labels. The following reviews will cover weakly-supervised techniques related to semantic segmentation.

»*Going to Extremes: Weakly Supervised Medical Image Segmentation*« [211] from 2021 proposes a new method for segmenting medical images. The techniques aim to reduce the general effort of generating new training datasets by minimizing manual intervention to just a few clicks. The method consists of three steps. The first step is to create an initial segmentation based on the extreme points annotated manually. The points identify the regions of interest and are subsequently provided as input to the »*random walker algorithm*«. The algorithm can be used for simple segmentation using a set of known labeled pixels to determine the labels of other pixels [94]. The initial segmentation is then used as »*noisy supervision*« to train a CNN to segment organs of interest. The authors claim this type of initialization is relatively robust for six different tasks from medical image segmentation. The third step consists of fine-tuning the network through multiple training iterations. The results show that the proposed model generalizes to unseen data with limitations for organs with diverse interior textures or extremely concave curved shapes, such as the pancreas.

»*Learning Pixel-Level Semantic Affinity With Image-Level Supervision for Weakly Supervised Semantic Segmentation*« [5] from 2018 present a DNN called AffinityNet that predicts semantic affinities on pixel-level while originally trained with image-level class labels only. Image-level labels only indicate the existence of a particular object class and do not inform about object location or shape, which are essential for semantic segmentation. The authors employ the Class Activation Maps (CAM) technique [311] to achieve a rough localization. CAM is a method used to generate visualizations of the significant regions in an image that contribute to a feasible classification decision because the proposed AffinityNet was trained using image-level class labels. As illustrated in Fig. D.10, affinity labels are derived from the CAM by sampling pairs of neighboring coordinates and assigning labels based on class consistency. Affinity labels and training images are then used to train AffinityNet, which outputs semantic affinities within local image areas[10]. With the CAM, those predictions are then used with the random walk algorithm [94] to create output labels for a semantic segmentation network. The authors claim to achieve state-of-the-art performance for models trained with equal supervision and even outperform some older supervised models from the »*early days*«[162].

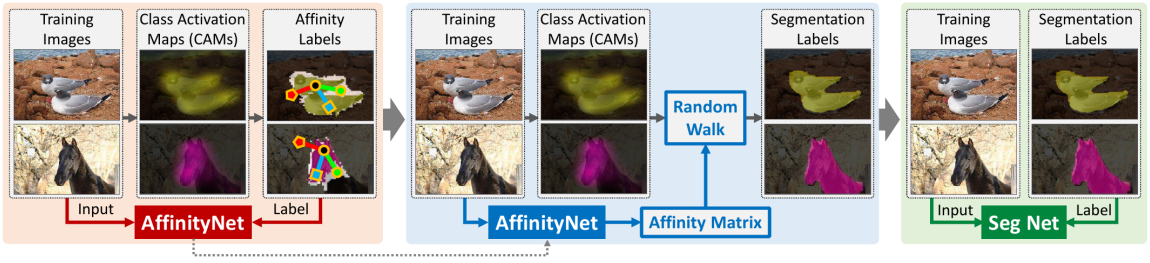


Fig. D.10. Visual concept of AffinityNet and SegNet. The image is taken from [5]

[9] Coarse-grained labels are also called high-level labels which are used to categorize data in a very general and simplified way. Instead of specifying the type of animals in images, coarse-grained labels could provide »*animal*« as the labeled category.

[10] In computer vision, local image areas refer to specific regions or patches of an image that are smaller in size than the entire image. These local areas can be used for various tasks, such as object detection, image segmentation, and feature extraction.

D.7. Active Learning

AL is a ML method where the algorithm chooses the most informative data points from an unlabeled dataset and asks the user to label them, which helps to reduce the number of labeled data necessary for training [223]. See Fig. D.11 for further information. There are several approaches to selecting informative data, such as uncertainty-based sampling [227] or diversity-based [290] sampling. Uncertainty-based sampling is an active learning strategy that selects data points that are the most uncertain and, thus, are highly informative. Uncertainty in this context refers to what degree of confidence the model has in its predictions for certain data points. Examples of uncertainty-based sampling are query-by-committee [131] and maximum entropy sampling [171]. Diversity-based sampling aims to select data points that are diverse or different from those labeled already. Here the goal is to capture the full diversity of the data, which can be useful where labeled data is biased towards a particular subset [4]. Some examples include density-weighted active learning [71] or uncertainty sampling with diversity [277].

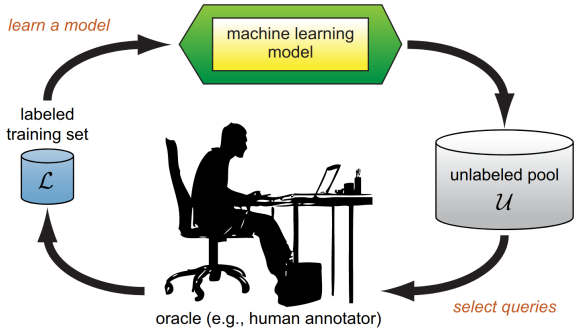


Fig. D.11. Active Learning (AL) cycle. Image from [223]

»*A Survey on Active Learning and Human-in-the-Loop Deep Learning for Medical Image Analysis*« [37] from 2019 presents an extensive survey of active learning and human-in-the-loop deep learning methods for medical image analysis. The authors discuss the importance of AL techniques by addressing the challenges of small data sets, high annotation costs, and large inter- and intraobserver variability[11] when annotating medical data. The paper subsequently describes the active learning process, numerous strategies for selecting informative data points, and advantages such as better and faster results, less needed expertise for labeling tasks, and freeing up expert time.

»*ViewAL: Active Learning With Viewpoint Entropy for Semantic Segmentation*« [235] from 2020 proposes a new active learning framework called ViewAL for semantic segmentation of 3D point clouds. The active learning strategy exploits inconsistent predictions across different image views, whereas traditional uncertainty sampling techniques operate on single images. The authors claim that single image sampling ignores geometric constraints[12] which are essential when examining the quality of network predictions. Generally,

[11] Interobserver variability is the variation in annotations made by different annotators when analyzing the same medical image. This variability can occur due to different training, expertise, or perception pieces. Intraobserver variability is the variation in annotations made by the same annotator caused by fatigue or lack of standardized annotation procedures.[112]

[12] Geometric constraints refer to a set of rules or relationships between objects in space that describe how they are positioned, oriented, or related to each other. For further information on geometric constraints, see [190].

the ViewAL framework involves four steps. The first step trains a regular model on a labeled dataset. The second step uses the trained model to predict unlabeled parts of the data. The third step selects superpixels with high uncertainty from the predictions, which are subsequently requested in the fourth step from a human annotator. The authors evaluate their framework on three public datasets called SceneNet-RGBD [173], ScanNet[63] and Matterport3D[46] achieving promising results.

D.8. Semi-supervised Learning

As defined in Section 2.2.2, semi-supervised learning generally uses labeled and unlabeled data. The paper »*A survey on semi-supervised learning*« from 2019 provides a relatively new survey of semi-supervised learning. The authors define semi-supervised learning as a branch that combines supervised and unsupervised learning to improve performance in one of these tasks by utilizing information associated with the other task. The article starts by covering basic concepts of semi-supervised learning methods and connecting the task to clustering. They further cover wrapper methods which include self-training, co-training, and boosting. After describing unsupervised processing, different intrinsically semi-supervised methods are covered, which aim to include unlabelled samples in the object function without relying on intermediate steps or supervised base learners. The article additionally covers transductive methods, which do not have a training and testing phase, unlike the previously mentioned methods. The significant difference between transductive methods is that they can only predict part of the input space. The predictions are restricted to the unlabelled set given in advance. The last section outlines future perspectives for semi-supervised learning to be a vital tool in uncovering complex structures in data. The authors also estimate that the distinction between clustering and classification will fade.

»*Casting Geometric Constraints in Semantic Segmentation as Semi-Supervised Learning*« [242] from 2019 discusses the challenges of generalizing semantic segmentation to new indoor scenes even if trained with a large dataset such as SUNRGB-D [239]. The problem is known as dataset bias, domain shift, sample selection bias, or class imbalance and root in the complexity of the visual world where datasets describe just some of its aspects [255]. The authors of [242] propose to exploit geometric constraints as a semi-supervised term to create a model which can be used to segment target sequences of ScanNet[63] using only annotations from SUNRGB-D. The proposed method is shown to be effective through quantitative and qualitative experiments.

»*Shape-aware Semi-supervised 3D Semantic Segmentation for Medical Images*« [153] from 2020 propose a novel shape-aware semi-supervised segmentation strategy. The authors develop a multi-task network that simultaneously predicts segmentation masks and signed distance maps. The signed distance maps are predicted for labeled and unlabelled data and combined within a proposed adversarial loss which is further combined with the regular segmentation loss. See Fig. D.12 for further information on the exact layout, which allows the model to generalize to the unlabelled portion of the dataset by enforcing similar distance map distributions. Experiments show that the method outperforms current state-of-the-art approaches with improved shape estimation validating its efficacy.

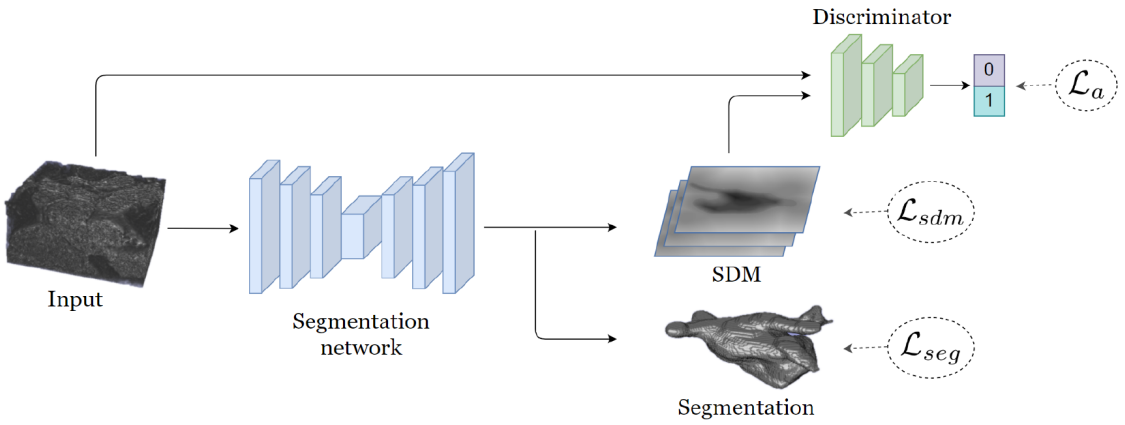


Fig. D.12. Framework of a semi-supervised shape aware implementation. The adversarial loss $\mathcal{L}_\alpha$ takes as input predictions from the labeled and unlabelled part of the dataset and serves as a regularization term to the model. Image from [153].

D.9. Domain Adaption

The paper »*Deep visual domain adaptation: A survey*« [279] from 2018 provides a detailed summary about Domain Adaption (DA). The authors define DA as a case of transfer learning that utilizes labeled data in a source domain to execute new tasks in a target domain. There can be several differences when speaking about source and target domains. A significant difference is the distributional difference. The authors describe several factors, such as illumination, pose, or image quality which can cause differences between a source and a target domain. Another term for these differences is domain shift. An example of domain adaption would be to use a model trained with images from animals recorded in a zoo for images of animals recorded in the wild. DA techniques attempt to solve these types of challenges. While numerous techniques exist for shallow DA, the paper focuses on deep DA. The difference is that the latter is a specific type of DA focused on deep learning models. When describing the algorithms for DA, the authors divide them into two categories called instance-based DA and feature-based DA. To adapt a source domain to a target domain with »*instance-based methods*«, one would have to train a model with specific weights for instances which would reduce the discrepancy between the source and target domain. In other words, we would first compute a weight for each instance in the source domain based on its similarity to the instance in the target domain and then use these weights to train a machine-learning model on the source domain. Instances with higher weights are given more importance during training, while instances with a lower weight are given less importance. To adapt a source domain to a target domain with »*feature-based methods*«, we could identify the most relevant features for the target domain with techniques such as mutual information [270] or correlation analysis [100]. The goal is to match the distribution of the target domain. Subsequently, we can increase or lower the weights to features depending on their relevance and train the model with the chosen weights. The model will then be more sensitive to the features most relevant to the target domain and less sensitive to the less relevant features. The following paragraphs present implementations of DA within the context of semantic segmentation.

»*A Novel Domain Adaptation Framework for Medical Image Segmentation*« [91] from 2018 proposes a novel

DA strategy to face the scarcity of training data in medical imaging. The core idea is to create a synthetic dataset of tumor-bearing Magnetic Resonance (MR) images and adapt those to the domain of real images from BRATS [177]. The synthetic dataset is created with a custom process based on a partial differential equation tumor model [246], which captures the time evolution of enhancing and necrotic tumor concentrations of tumor-induced brain edema. As this synthetic dataset and the dataset from BRATS display a high difference in intensity distributions, a new model called »*CycleGAN*« is introduced to perform domain adaption from the synthetic to the real dataset. »*CycleGAN*« is defined as

$$G : X \rightarrow Y \tag{D.4}$$

where X is the synthetic data and Y the real data. The goal is to learn the mapping G such that the distribution of images from $G(X)$ are indistinguishable from the distribution of Y by using an adversarial loss [13] The DA approach is then combined with a custom segmentation method achieving promising results on the BRATS 18 data. They further claim to have augmented the existing data by twice the number of brain images.

»*Synergistic Image and Feature Adaptation: Towards Cross-Modality Domain Adaptation for Medical Image Segmentation*« [48] from 2019 proposes a new unsupervised domain adaption framework called Synergistic Image and Feature Adaption (SIFA). Their framework aims to address the problem of domain shift in deep learning. The authors describe their approach as a combination of feature and image adaptions to transform domain-invariant[14] features and image appearance for a segmentation task in a target domain. Feature and image adaptions refer to the concept of feature and instance-based methods described above. SIFA is validated on cross-modality medical image segmentation of cardiac structures, which refers to segmenting different anatomical structures through different imaging modalities such as magnetic resonance or computed tomography. The results show that SIFA recovers performance degradation caused by domain shift and outperforms state-of-the-art methods after applying their DA technique.

D.10. Post-processing

Post-processing for semantic segmentation refers to a set of techniques applied to the output of a segmentation algorithm to increase their quality and accuracy. Some standard post-processing techniques used in semantic segmentation are noise reduction [266], edge detection [232], morphological operations [57], conditional random fields [68], region merging or region splitting [195]. The following paragraphs summarize some specific post-processing methods for segmentation.

»*Anatomical Priors for Image Segmentation via Post-Processing with Denoising Autoencoders*« [144] from 2019 introduces a new technique called Post-DAE which stands for post-processing with denoising autoencoders. The idea is to produce anatomically plausible segmentations by improving pixel-level predictions

[13] An adversarial loss is a loss function used in Generative Adversarial Network (GAN)[167], a type of NN architecture used for generating realistic data, such as images [293], audio [69], or videos [297].

[14] Domain invariance refers to the ability of a model or algorithm to perform well on data from a domain that is different from the one on which it was trained. In other words, a model or algorithm that exhibits domain invariance can generalize its performance to new, unseen data from different domains. For further information, see [262].

in a post-processing fashion from any classifier. The denoising autoencoder is defined as

$$\mathcal{L}_{DAE}(S_i) = DSC(S_i, f_{dec}(f_{enc}(\phi(S_i)))) \tag{D.5}$$

where S_i is an anatomically plausible segmentation mask that is attempted to be reconstructed after being transferred in a lower dimensional space. The function $\phi(S_i)$ is part of a mask degradation strategy and aims to produce noisy segmentations used for training. The authors suggest the DL as a loss function as an objective to minimize. Once the model is trained, any prediction from any classifier can be improved with the proposed model to a plausible segmentation. The results presented show the effectiveness of their approach measured by an increase of about 7% of the DSC compared to other post-processing methods such as random forest and conditional random fields.

»*KPRNet: Improving projection-based LiDAR semantic segmentation*« [138] from 2020 is a paper that discusses the segmentation for LiDAR scans within the field of autonomous driving. The authors state that current LiDAR-based segmentation methods can be categorized into point-wise methods acting directly on the 3D point cloud and RGB image segmentation methods. Their proposed method aims to combine both approaches using an improved CNN architecture for 2D projected LiDAR sweeps[15] and a learnable module based on KPConv to replace post-processing steps. The model outperforms the best methods on the SemanticKITTI benchmark, achieving a mIoU of 63.1, outperforming state-of-the-art models. The paper summarizes that the suggested technique achieves better accuracy in segmenting LiDAR scans by combining the strengths of point-wise and traditional segmentation methods.

D.11. Multi-task Learning

The paper »*A Survey on Multi-Task Learning*« [138] from 2020 defines multi-task learning as follows:

"Given m learning tasks $\{T_i\}_{i=1}^m$ where all the tasks or a subset of them are related, multi-task learning aims to learn the m tasks together to improve the learning of a model for each task T_i by using the knowledge contained in all or some of other tasks." - Yu Yhang, Qiang Yang [138]

The paper presents an excellent summary of multi-task learning from perspectives such as algorithmic modeling, applications, and theoretical analyses. The authors divide multi-task learning algorithms into five categories. »*Feature learning approach*« focuses on learning common feature representations for multiple tasks where the learned common feature representation can be a subset or a transformation of the original feature representation. The goal of the »*Low-rank approach*« is to learn a low-dimensional representation of the data that captures shared information across all tasks. The »*task clustering approach*« consists of grouping related tasks based on some similarity measure, allowing the model to extract shared information across related tasks and improve performance. »*Task relation learning*« is about learning quantitative relations between tasks from data automatically, and the »*Decomposition approach*« decomposes the model

[15] LiDAR sweeps are 3D point clouds obtained by scanning the surrounding environment using a LiDAR (Light Detection and Ranging) sensor.

parameters of all tasks into several components such as low-rank or sparse components [16] to better capture the underlying structure of the data and improve performance. The paper further provides information on handling a large number of training data if multiple tasks are involved. It presents multiple applications for multi-task learning techniques, such as CV, bioinformatics, health informatics, speech, NLP, or the web. The following paper presents a technique for applying multi-task learning to semantic segmentation.

»*A Deep Multi-task Learning Framework Coupling Semantic Segmentation and Fully Convolutional LSTM Networks for Urban Change Detection*« [193] from 2021 proposes a deep learning framework for urban change detection which combines semantic segmentation and fully convolutional LSTM networks. As input, the framework uses multi-temporal sensing imagery to identify changes in urban areas over time. Relevant changes could be the construction of a building or its removal. As mentioned above, the framework combines two main components. The semantic segmentation network component is trained to label pixels in the input related to their class. In contrast, the LSTM network component is trained to predict label changes over time. The two components are coupled in a multi-task learning fashion where they share some of their parameters, being trained jointly to optimize the segmentation and change detection task. The framework is further evaluated on several datasets where the results outperform other state-of-the-art methods by at least 2.1% and 4.9% in the Attica VHR and SpaceNet7 [267] dataset, respectively.

D.12. Attention Mechanism

The paper »*A General Survey on Attention Mechanisms in Deep Learning*« [186] from 2021 presents an extensive cross-domain overview of attention mechanisms within the field of deep learning. While attention mechanisms were introduced first in the field of natural language processing, they quickly became prevalent in a variety of tasks such as speech recognition [56], sentiment analysis [148], image captioning [116] or text classification [158]. The main reasons why attention mechanisms became so popular are described as the state-of-the-art results of attention models, the ability to train attention models jointly with base models, the different interpretations of highly complicated network models, and the introduction of the transformer model that proved how effective attention mechanisms could be [269]. The paper consists of three significant chapters, where the first introduces the general functionality of attention models. The second provides a detailed taxonomy of attention models divided into »*Feature-related attention mechanisms*«, »*General attention mechanisms*«, and »*Query-related attention mechanisms*«. The last chapter provides an overview of performance measures for evaluating attention models.

Using the same definitions as in [186], every model that can employ attention is considered a task model, taking any input and carrying out a specific task to produce the desired output. Each task model further consists of four submodels: the feature model, the query model, the attention model, and the output model. The feature and the query model are used to prepare the input for the attention calculation. The core idea is that with the help of the attention mechanism, which consists of a query and feature vector, a specific context vector is created to allow the network to learn how much weight it wants to put on each of the hidden

[16] The low-rank and sparse rank components play different roles in the decomposition approach to multi-task learning. The low-rank component captures the shared information across tasks, and the sparse component captures the task-specific information. By combining these two components, the model can effectively leverage the shared information across tasks while also capturing the unique characteristics of each task.

states of the input sequence. The rest of this section describes some examples which have implemented attention mechanisms in semantic segmentation networks.

»*Semantic Segmentation With Attention Mechanism for Remote Sensing Images*« [307] from 2022 proposes an end-to-end attention-based semantic segmentation network called SSAtNet to classify pixels of remote sensing images. The paper discusses the problematic task of segmenting high-resolution remote sensing images which contain detailed information about ground objects exhibiting large intraclass and small interclass variance[17]. The proposed model applies the attention mechanism to the ASPP module[18] to improve model performance. Additionally, they suggest a pooling index correction scheme which adds the upsampled feature maps with the pooling index map[19] from the encoder to restore fine details. See Fig. D.13 for further information on the proposed network. To further increase quality, the paper describes several data augmentation methods, such as random sampling, rotating, and scale. The model is trained on the ISPRS Vaihing dataset [2], achieving state-of-the-art results compared to several other networks.

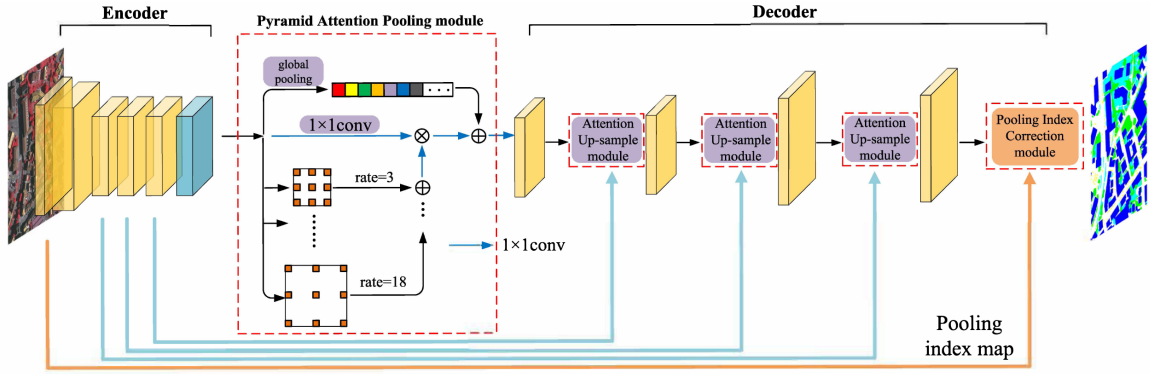


Fig. D.13. Attention-based semantic segmentation framework for remote sensing images. Image from [307].

»*MA-Unet: An improved version of Unet based on multi-scale and attention mechanism for medical image segmentation*« [41] from 2020 presents MA-Unet which is an improved version of the U-Net architecture which that combines multi-scale features and attention to achieve better performance in medical image segmentation tasks. The authors propose this model to overcome the limitations of U-Net when capturing multi-scale context and addressing the imbalance between fore and background classes. The first improvement of MA-Unet is a Multi-scale Context Aggregation module designed to capture the context information at different scales. It uses a series of parallel convolutional layers with different dilation rates to help the model to learn features at several scales to improve segmentation performance. The second improvement over U-Net is a Channel Attention module and a Spatial Attention module, where the former focuses on enhancing the most relevant features in each channel and the latter focus on the most critical spatial locations, allowing the network to pay more attention to the target regions and suppress irrelevant information. The

[17] Intraclass variance refers to the variation within a single group or class of data while interclass variance is the variation between two or more groups or classes of data. The terms are often used in the analysis of variance ANOVA which determines the variability of data within a group or class or the differences between the means of different groups or classes [244].

[18] ASPP stands for Atrous Spatial Pyramid Pooling, a module used in deep neural networks for image segmentation tasks. The ASPP module was first introduced in 2017 by Chen et al. as part of the DeepLabv3+ architecture for semantic segmentation.[51]

[19] Pooling index maps are a type of feature map that contains information about the locations of the maximum values that were selected during the pooling operation in a CNN.

model is evaluated on the ISBI 2015 EM [1] and BraTS 2018 [17] datasets and compared against the original U-Net and several other state-of-the-art methods. The results demonstrate that MA-Unet outperforms the other models and improves U-Net by incorporating multi-scale context and attention.

D.13. Reinforcement Learning

While some minimal concepts about RL were already discussed in Section 2.2.2, this section will first cover some basics about Deep Reinforcement Learning (DRL) which combines deep learning and RL and then relate it to semantic segmentation.

The paper »*Deep Reinforcement Learning: An Overview*« [155] from 2017 presents an extensive overview of DRL explaining key components, algorithms, and applications. The paper highlights the significance of DRL and the impact it has on various fields such as robotics [183], healthcare [161], finance [115][160] and gaming [143]. The paper further provides background on RL, deep learning, and ML and then discusses the difference and critical algorithms of DRL frameworks, such as value-based, policy-based, and model-based frameworks. Some key algorithms mentioned in this paper would be Deep Q-Networks (DQNs) [80], Double DQNs [102], Dueling DQNs [284], Deep Deterministic Policy Gradient [44], Proximal Policy Optimization [221], Trust Region Policy Optimization [220], and Hindsight Experience Replay [11], discussing their advantages, limitations, and improvements over traditional RL methods.

»*Context-Reinforced Semantic Segmentation*« [314] from 2019 introduces a novel Context-reinforced semantic segmentation network called CiSS-Net to improve semantic segmentation performance by leveraging context information from predicted segmentation maps. The network comprises two sub-networks called the Context Net and the Segment Net. The Context Net is designed to learn context information from the predicted segmentation maps, while the Segment Net incorporates the learned context to enhance the segmentation process. The context learning problem is designed as a Markov Decision Process with DRL used to optimize the process. The predicted segmentation maps serve here as the environment and the Context Net as the agent. The method results in an end-to-end trainable network that achieves state-of-the-art performance on several segmentation datasets by improving the mIoU of 3.9% over the baseline models. For further info on the architecture, see Fig. D.14

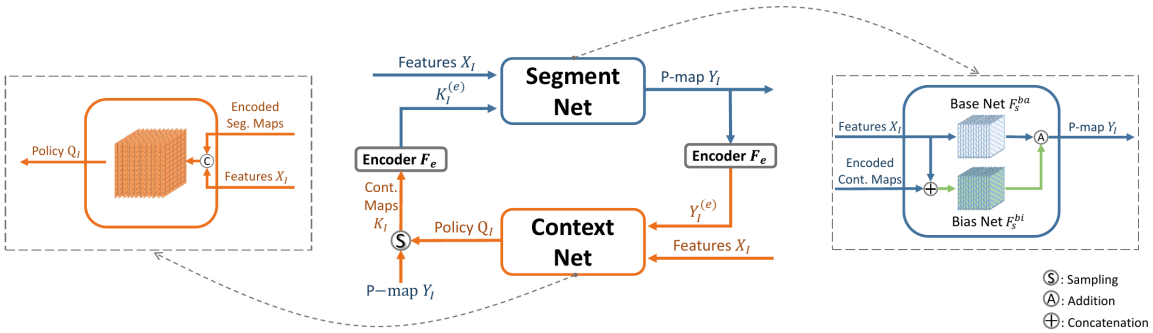


Fig. D.14. The image describes the two sub-networks called Segment Net and Context Net, which mutually benefit each other and work iteratively. The Segment Net is trained based on specific context maps generated by the Context Net to improve the semantic segmentation task further. The image was taken from [314].

E. Configuration Details

E.1. Parameter Definition

This subsection presents a comprehensive overview of the parameters employed by the proposed method. Table E.1 offers a categorized list of all parameters, accompanied by a concise description, the variable name in the source code, the data type, the default value, and a flag indicating the default status. This flag signifies whether the default value has been modified for this project. Subsequent paragraphs explore each category, discussing the associated parameters and the rationale behind the selected values.

Category	Description	Variable name	Data type	Default value	Default status
Checkpoint and model saving	Save top models per metric in training/validation.	`save_top_k`	int	0	✗
Checkpoint and model saving	Flag for pre-trained checkpoint availability.	`checkpoint_available`	bool	False	✓
Data and evaluation	Graphical test samples for W&B logging.	`num_tests`	int	5	✓
Data and evaluation	Train/validation dataset split ratio.	`split_ratio`	int	5	✓
Data and evaluation	Calculates stats for each label & averages them	`avg`	string	'macro'	✓
Data and evaluation	Data selection % for train/val/test subsets.	`selection_percentage`	float (0, 1]	1	✗
Data and evaluation	Image size to train with.	`img_size`	tuple	(256,256)	✗
Hardware and infrastructure	GPU device(s) for training & inference.	`gpu`	int	0	✗
Loss function	Tversky loss alpha & beta parameters.	`tversky_alpha`, `tversky_beta`	float	$\alpha = 0.3, \beta = 0.7$	✓
Loss function	Focal loss alpha & gamma parameters.	`focal_alpha`, `focal_gamma`	float	$\alpha =$ None, $\gamma = 2$	✓
Loss function	Alpha parameter for the Hausdorff loss function.	`hd_alpha`	float	2	✓
Merge strategy WS	Weight contribution per loss for the WS strategy	`ws_1`, `ws_2`, `ws_3`	float	$[1.0, 1.0, 0.1]$	✓
Merge strategy NWS	Weight contribution per loss for the NWS strategy	`nws_1`, `nws_2`, `nws_3`	float	$[1.0, 1.0, 0.1]$	✓
Merge strategy PBM	Alpha parameter to control slope	`pbm_alpha`	float >1	2.0	✗
Merge strategy PBM	I parameter to control ascending or descending	`pbm_I`	[0,1]	1	✗
Merge strategy TBM	Time-based merging flag	`time_based_merging`	bool	False	✗
Merge strategy TBM	Time-based merging for distribution loss control	`time_based_alpha`	float >1	2.0	✓
Merge strategy TBM	Time-based merging for region loss control	`time_based_beta`	float >1	2.0	✓
Merge strategy TBM	Time-based merging for boundary loss control	`time_based_gamma`	float >1	2.0	✓
Merge strategy TBM	Time-based merging for distribution loss control	`time_based_I_1`	float [0,1]	1.0	✗
Merge strategy TBM	Time-based merging for region loss control	`time_based_I_2`	float [0,1]	1.0	✗
Merge strategy TBM	Time-based merging for boundary loss control	time_based_I_3	float [0,1]	1.0	✗
Model architecture	Model architecture layers count	`number_layers`	int	5	✓
Model architecture	Initial features or channels count	`features_start`	int	64	✓
Model architecture	Bilinear upsampling flag	`bilinear`	bool	False	✓
Model architecture	Architecture type to train with	`model_type`	string	'UNET'	✓
Training configuration	Training batch size	`batch_size`	int	1	✓
Training configuration	Initial optimizer learning rate	`learning_rate`	float	0.009	✗
Training configuration	Gradient accumulation steps count	`grad_batches`	int	1	✓
Training configuration	Training epochs count	epochs	int	100	✗
Training configuration	Learning rate finder flag	`auto_learning_rate`	bool	False	✗
Training configuration	Early stopping flag	`early_stopping`	bool	False	✗

Tab. E.1. List of all hyperparameters used in this project. The column `Default status` indicates whether the `Default value` will be used or not.

Checkpoint and Model Saving

- `save_top_k` allows saving the weights or configurations of the top k performing models. For this

project this value was changed according to the task at hand. For inference purposes, this valid was set to 1 which allows to make predictions on a model after training.

- `checkpoint_available` is not a part of the basic configuration for the presented method and is not utilized in this project.

Data and Evaluation

- `num_tests` generates up to n visual testing results, which are logged in the Weights and Biases (wandb) graphical user interface. These results include n samples of $x \in X$, their corresponding ground truth $y \in Y$, and the prediction $\hat{y} \in \hat{Y}$. These results are displayed as shown in Fig. 6.1.

- `split_ratio` determines the proportion of the validation set taken from the training set as $\frac{1}{\texttt{split\_ratio}}$. The value remains at 5 for all experiments.

- `selection_percentage` varies depending on the experiment. In particular, this parameter will change or be included within the sweep configuration for the ablation study, where results are evaluated on dataset subsets.

- `average` defines the statistics or reduction applied when calculating a specific metric. The default value 'macro' means the metric is calculated over each class first and then taken as the average to output the final score. See Section 2.3.3 for further details.

- `img_size` indicates which image size is used. The default value of (256,256) is used for the Medaka fish and ISIC dataset, while (512,512) is used for the IDRID.

Hardware and Infrastructure

- `gpu` is chosen based on current availability.

Loss Function

- `tversky_alpha, tversky_beta` are chosen based on the optimal values as described in [218].

- `focal_alpha, focal_gamma` are chosen based on the optimal values as described in [157].

- `hd_alpha` is selected according to the suggested value described in [129].

Merge strategies

- `ws_1,ws_2,ws_3` are set to $[1, 1, 0.1]$ based on experimental findings, which indicate that boundary-based losses are typically much higher than region-based and distribution-based losses. Therefore, the weights for boundary-based losses are manually set to 0.1 for weighted and normalized weighted sum strategies. As previously discussed, these high values in boundary-based outputs can be attributed to the distance transform. Particularly at the beginning of training, when

segmented regions are still highly inaccurate, the losses from boundary-based types are extremely high and often unstable if trained as a single loss, making it difficult for them to converge. By setting the weight a factor of 10 lower, the algorithm can focus more on distribution and region-based outputs to provide a robust starting point while still incorporating boundary information, albeit at a smaller scale.

- `nws_1,nw_2,nws_3` are set to $[1, 1, 0.1]$, similar to the weighted sum strategy.

- `pbm_alpha`, by design, must be larger than 1. This hyperparameter controls the function's slope, as illustrated in Fig. 4.3. The default value of this hyperparameter is set to 2, which is applied in the basic sweep configuration. To further examine performance-based merging, an individual sweep is configured with values for $\alpha \in 1, 2, 3, 4, 5$.

- `pbm_I` determines whether an ascending or descending strategy is employed, as detailed in Section 4.2. Separate sweep configurations for ascending and descending strategies are used to compare their performance.

- `time_based_merging` is a boolean flag with a default value of false. Individual sweep configurations are necessary to examine the performance of time-based merging strategies. Since this strategy can be applied independently to any other strategy, it will be applied to the top-k results to enhance performance even more.

- $tbm_{\alpha,\beta,\gamma,I_1,I_2,I_3}$ can be used to control the slope and ascending or descending property for time-based merging, as described in Section 4.2. If `time_based_merging` is set to false, these parameters do not affect the use of other strategies.

Model architecture

- `number_layers` is set to the default value of 5, following the recommendation from the original U-Net paper [209].

- `features_start`, which represents the initial number of channels, is set to 64. This default value is also suggested by the U-Net paper [209].

- `bilinear` is set to false. This flag determines the type of upsampling strategy used in the model. When set to false, transposed convolution is employed for upsampling, which has yielded good results in terms of computational efficiency [140].

- `model_type` indicates which type of network architecture is used. Here U-net will be used exclusively.

Training configuration

- `batch_size` is set to 1 throughout the entire experiment. Although this choice results in higher computational costs, it allows for a more detailed examination of high-resolution data, especially in

datasets with few samples and extremely challenging classification tasks.

- `learning_rate` is set to 0.009 as a default value, derived from the average optimal values calculated for specific datasets with a learning rate finder
- `auto_learning_rate` is enabled, particularly for difficult tasks such as the classification for the Indian Diabetic Retinopathy Image Dataset (IDRID).
- `grad_batches` is set to 1 as the default value for the entire experiment.
- `epochs` is set to 100 for the Medaka fish [222] and International Skin Imaging Collaboration (ISIC)[58] datasets, and to 150 for the IDRID.

E.2. Sweep Configuration

Baseline configuration
The baseline configuration entails training six individual loss functions thoroughly detailed in Section 2.3.2. Establishing a baseline is crucial for this project, as it enables comparison between the proposed merging strategies and the baseline performance. Since there are no hyperparameters to search for in this case, the baseline can be conveniently set up using the following sweep configuration.

```
#Loss functions and types
loss_dict = {
    1: {'type': 'db', 'name': 'CE'},
    2: {'type': 'db', 'name': 'Focal'},
    3: {'type': 'rb', 'name': 'Tversky'},
    4: {'type': 'rb', 'name': 'Dice'},
    5: {'type': 'bb', 'name': 'HD'},
    6: {'type': 'bb', 'name': 'Boundary'},
}
#Loss Combinations
loss_combinations = get_loss_combinations(loss_dict,triple=False,
 double=False,baseline=True)
#Configuration
sweep_configuration = {
    'method': 'grid',
    'name': 'baseline',
    'metric': {'goal': 'minimize', 'name': 'val_loss'},
    'parameters':
    {
        'selection_percentage': {'values': [0.32, 0.64, 1]},
        'loss': {'values': loss_combinations},
    }
}
```

Listing E.1 The baseline configuration comprises training six loss functions for each data selection percentage, leading to a total of 18 models to be trained.

Discrete configuration

Considering the discrete number of combinations and strategies presented in Section 4.2, a custom sweep configuration can be devised to iterate through all loss combinations and merge strategies from the averaging, static weighting, and extreme point types. This configuration encompasses parameters that do not require optimization. The subsequent sweep configuration setup results in a total of 20 losses $\cdot\, 6$ merge strategies $\cdot\, 3$dataset selections $= 360$ models generated from this discrete parameter space.

```
#Loss functions and types
loss_dict = {
    1: {'type': 'db', 'name': 'CE'},
    2: {'type': 'db', 'name': 'Focal'},
    3: {'type': 'rb', 'name': 'Tversky'},
    4: {'type': 'rb', 'name': 'Dice'},
    5: {'type': 'bb', 'name': 'HD'},
    6: {'type': 'bb', 'name': 'Boundary'},
}
#Loss Combinations
loss_combinations = get_loss_combinations(loss_dict,triple=True,double
 =True,baseline=False)
#Merge Strategies
merge_strategies =['max_strategy','min_strategy','harmonic','
 arithmetic','weighted_sum','normalized_weighted_sum']
#Configuration
sweep_configuration = {
    'method': 'grid',
    'name': 'discrete_configuration',
    'metric': {'goal': 'minimize', 'name': 'val_loss'},
    'parameters':
    {
        'selection_percentage': {'values': [0.32, 0.64, 1]},
        'strategy': {'values': merge_strategies},
        'loss': {'values': loss_combinations},
    }
}
```

Listing E.2 The discrete sweep configuration setup is composed of a dictionary that includes six loss functions, along with their corresponding types. This dictionary is provided to the `get_loss_combinations` function, which generates a list of 20 double and triple loss combinations, as previously introduced in 4.4. Moreover, the configuration encompasses six merge strategies, excluding time-based and performance-based strategies. Both the loss combinations and merge strategies are then incorporated into the sweep configuration dictionary, which creates one run for each combination. The objective of this configuration is to minimize validation loss. It is important to note that this configuration does not have any specific hyperparameter set, creating $20 \cdot 6 \cdot 3 = 360$ distinct models.

Continous configuration

This type of setup is based on Hyperband, an early stopping technique for continuous hyperparameter optimization based on the Successive Halving algorithm. It is designed to allocate resources efficiently to the most promising configurations by adaptively allocating more resources to configurations that perform well in the early stages of training. The goal is to find the best configuration with a limited budget of resources. Weights and Biases (wandb) implements the algorithm as follows:

1. ***Initialization***: Sample configurations from the search space.

2. ***Dividing the resource budget***: Divide the budget into brackets with different resource allocation strategies.

3. ***Successive Halving***: Within each bracket, start with multiple configurations, allocate resources, evaluate performance, discard poorly performing ones, and allocate more resources to the remaining ones. Repeat until one configuration is left.

4. ***Hyperband iteration***: Repeat steps 2-3 for several iterations, defined by `max_iter`.

5. ***Selection of the best configuration***: Choose the configuration with the best performance after all iterations.

For a more detailed description of this algorithm, refer to [151].

The following two setups contain two configurations that encompass performance-based and time-based merging strategies, including continuous hyperparameters to be searched for.

```
#Loss functions and types
loss_dict = {
    1: {'type': 'db', 'name': 'CE'},
    2: {'type': 'db', 'name': 'Focal'},
    3: {'type': 'rb', 'name': 'Tversky'},
    4: {'type': 'rb', 'name': 'Dice'},
    5: {'type': 'bb', 'name': 'HD'},
    6: {'type': 'bb', 'name': 'Boundary'},
}
#Loss Combinations
loss_combinations = get_loss_combinations(loss_dict,triple=True,double
 =True,baseline=False)
#Merge Strategies
merge_strategies=['performance_based_merging']
#Configuration
sweep_configuration = {
    'method': 'random',
    'name': 'continous_configuration',
    'metric': {'goal': 'minimize', 'name': 'val_loss'},
    'early_terminate': {'type': 'hyperband','max_iter': 150, 'eta':3,
 's': 2},
    'parameters':
    {
        'selection_percentage': {'values': [0.32, 0.64, 1]},
        'strategy': {'values': merge_strategies},
        'loss': {'values': loss_combinations},
        'pbm_I': {'min': 0.0, 'max': 1.0},
        'pbm_alpha': {'min': 1.0, 'max': 10.0}
    }
```

Listing E.3 The `Performance-based` configuration investigates 20 distinct loss function combinations alongside the performance-based merging strategy which involves incorporating a continuous search space. By utilizing a random selection approach within the Hyperband framework, the sweep configuration aims to identify an optimal set of hyperparameters effectively.

```
#Loss functions and types
loss_dict = {
    1: {'type': 'db', 'name': 'CE'},
    2: {'type': 'db', 'name': 'Focal'},
    3: {'type': 'rb', 'name': 'Tversky'},
    4: {'type': 'rb', 'name': 'Dice'},
    5: {'type': 'bb', 'name': 'HD'},
    6: {'type': 'bb', 'name': 'Boundary'},
}
#Loss Combinations
loss_combinations = get_loss_combinations(loss_dict,triple=True,double
 =True,baseline=False)
#Merge Strategies
merge_strategies=['max_strategy', 'min_strategy', 'arithmetic', '
 harmonic', 'weighted_sum', 'normalized_weighted_sum','
 performance_based_merging']
#Configuration
self.sweep_configuration = {
    'method': 'random',
    'name': 'continous_configuration',
    'metric': {'goal': 'minimize', 'name': 'val_loss'},
    'early_terminate': {'type': 'hyperband','max_iter': 150,'s': 5},
    'parameters':
    {
        'selection_percentage': {'values': [0.32, 0.64, 1]},
        'strategy': {'values': self.merge_strategies},
        'loss': {'values': self.loss_combinations},
        'pbm_I': {'min': 0.0, 'max': 1.0},
        'pbm_alpha': {'min': 1.0, 'max': 10.0},
        'tbm_alpha': {'min': 1.0, 'max': 10.0},
        'tbm_beta': {'min': 1.0, 'max': 10.0},
        'tbm_gamma': {'min': 1.0, 'max': 10.0},
        'tbm_I_1': {'min': 0.0, 'max': 1.0},
        'tbm_I_2': {'min': 0.0, 'max': 1.0},
        'tbm_I_3': {'min': 0.0, 'max': 1.0},
    }
}
```

Listing E.4 The `Time-based` configuration on the other hand is applied to all other merge strategies in combination with the 20 loss combinations as described in Chapter 4. As the parameters are also from a continuous space, the same Hyperband algorithm is used as described for the `Performance-based` configuration.

F. Code Implementation Details

F.1. U-net

The section describes the U-Net implementation used in more detail consisting of four classes which are described as follows:

1. **DoubleConv module:** This module consists of two consecutive 3x3 convolutional layers, each followed by batch normalization and a ReLU activation function. This double convolution structure is used to increase the non-linearity of the network but also the receptive field, which allows it to capture more contextual information from the input image.

2. **Down module:** This module combines a 2x2 pooling layer with a stride of 2, followed by the DoubleConv module from (1). The max-pooling layer is responsible for reducing the size of the intermediate feature representations and increasing their receptive field.

3. **Up module:** This module upsamples the feature representations and concatenates feature maps from the corresponding level in the encoding path. Subsequently, the DoubleConv module from (1) is applied. The upsampling is performed using a transposed convolution where the weights of the convolution kernel are learnable, allowing the network to adapt its upsampling strategy during the training process. Refer to [87] for further information on transposed convolution.

4. **UNet class:** The primary U-Net model is composed of several layers, which include the DoubleConv module, a set of Down modules from (2) for the encoding path, and a set of Up modules for the decoding path, closing with a final 1x1 convolution layer for mapping the feature maps to the desired number of output classes. Fig. 2.20 illustrates a detailed representation of the used network architecture.

Fig. F.1 provides a more detailed representation of a UML class diagram.

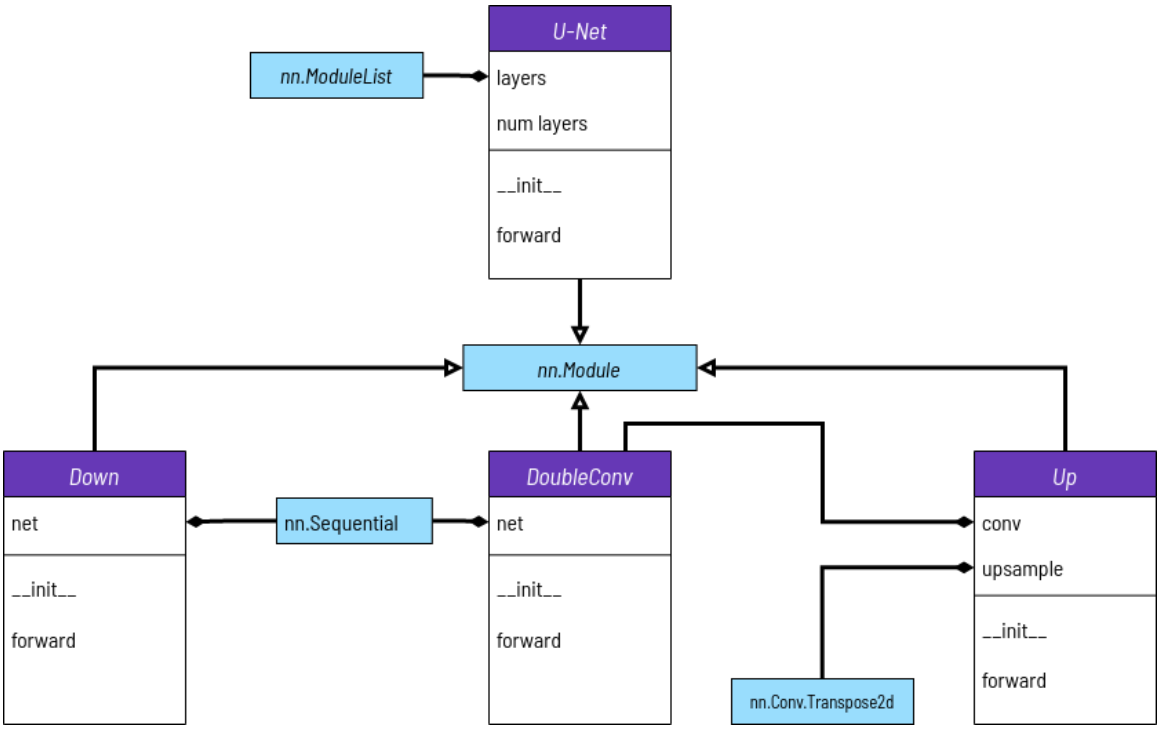


Fig. F.1. U-Net implementation modeled as a UML diagram. Each class consists of three compartments containing the name, the attributes, and the operations. Solid lines with hollow arrowheads indicate *inheritance*, and a filled diamond shape at the end of the line connecting the »*whole*« class to the »*part*« class is called a *composition* [1].

[1] A composition relationship is a strong form of the »*whole-part*« relationship, where the lifetime of the »*part*« objects is strictly tied to the lifetime of the »*whole*« object. In this case, the diamond shape is filled (usually black) and is located at the end of the line connected to the »*whole*« class. When the »*whole*« object is destroyed, the »*part*« objects are also destroyed.

G. Supplementary Results

This chapter offers an extensive analysis of the top-performing models across various datasets. It is a valuable supplement to the main chapter and a critical resource for further discussion and exploration.

For the Medaka Fish Dataset (MFD) and Skin Lesion Dataset (SLD), the baseline results encompass the top three models for each loss function and selection percentage, equating to nine models for each loss type. In contrast, all models for the Indian Diabetic Retinopathy Image Dataset (IDRID) dataset were trained on the full dataset. As such, the number of top models is three per loss type.

Discrete results include the top-performing models categorized by loss combination and merge strategy. Double and triple loss combinations are presented separately. Similar to the baseline results, the top three models for each loss and selection percentage are presented, resulting in nine models for each loss or merge strategy concerning the MFD and SLD. However, only three models are listed for the IDRID dataset.

The continuous results present the top three performers across all datasets when available. However, it is essential to note that the hyperband search algorithm may have eliminated certain loss combinations or merge strategies, as mentioned in the Section E.2, which might result in fewer combinations available for review.

The specific results contain trainings from distinct, promising combinations trained repetitively to reaffirm the validity by providing an equal iteration count as the baseline per loss which was not feasible for the entire range of discrete and continuous combinations due to lack of computational resources.

The section on PPV vs. TPR provides two distinct maps for each dataset, graphically representing the prevalence of PPV and TPR for every combination of loss and merge strategy. A higher numerical value in these maps signifies a more significant number of models with a superior TPR. Conversely, values below zero indicate a prevalence of models with a superior PPV. These maps effectively visualize the trade-off between PPV and TPR across different model configurations.

In addition to the top-performing models, this chapter presents supplementary results from the ablation study discussed in Section 7.4, which includes heatmaps illustrating results across different dataset sizes. Since only the MFD and SLD were trained using data subsets, the corresponding sections will be limited to these two datasets.

G.1. Medaka fish

G.1.1. Baseline

No.	R-time(m)	Loss type	Loss	Selection %	Iou	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
1	26	DB	CE	1.00	0.746	0.980	0.748	0.591	0.667	0.854	0.837	PPV
2	20	DB	CE	0.64	0.738	0.980	0.754	0.633	0.583	0.879	0.821	PPV
3	17	DB	CE	0.64	0.725	0.979	0.704	0.518	0.698	0.791	0.861	TPR
4	27	DB	*CE*	*1.00*	*0.721*	*0.981*	*0.773*	*0.604*	*0.528*	*0.860*	*0.790*	PPV
5	19	DB	CE	0.64	0.721	0.979	0.685	0.589	0.631	0.841	0.827	PPV
6	28	DB	CE	1.00	0.717	0.979	0.721	0.595	0.571	0.848	0.817	PPV
7	13	DB	CE	0.32	0.716	0.978	0.781	0.632	0.475	0.849	0.818	PPV
8	11	DB	*CE*	*0.32*	*0.710*	*0.973*	*0.762*	*0.485*	*0.621*	*0.811*	*0.852*	TPR
9	9	DB	CE	0.32	0.673	0.976	0.776	0.604	0.335	0.851	0.729	PPV
			CE Result		***0.719***	***0.978***	***0.745***	***0.584***	***0.568***	***0.843***	***0.817***	**PPV**
10	24	DB	FL	1.00	0.752	0.981	0.819	0.590	0.619	0.867	0.840	PPV
11	20	DB	*FL*	*0.64*	*0.731*	*0.982*	*0.763*	*0.516*	*0.665*	*0.839*	*0.819*	PPV
12	18	DB	FL	0.64	0.719	0.979	0.714	0.562	0.622	0.832	0.818	PPV
13	24	DB	FL	1.00	0.717	0.978	0.763	0.597	0.530	0.860	0.813	PPV
14	12	DB	FL	0.32	0.715	0.977	0.773	0.583	0.529	0.863	0.813	PPV
15	9	DB	*FL*	*0.32*	*0.709*	*0.977*	*0.797*	*0.681*	*0.383*	*0.885*	*0.793*	PPV
16	26	DB	FL	1.00	0.700	0.982	0.757	0.438	0.621	0.859	0.756	PPV
17	20	DB	FL	0.64	0.699	0.977	0.790	0.556	0.473	0.856	0.790	PPV
18	11	DB	FL	0.32	0.697	0.976	0.766	0.539	0.505	0.838	0.787	PPV
			FL Result		***0.716***	***0.979***	***0.771***	***0.562***	***0.550***	***0.855***	***0.803***	**PPV**
19	12	RB	DL	0.32	0.708	0.977	0.814	0.644	0.396	0.902	0.762	PPV
20	12	RB	DL	0.32	0.702	0.979	0.773	0.481	0.576	0.778	0.838	TPR
21	18	RB	DL	0.64	0.700	0.975	0.688	0.612	0.524	0.833	0.811	PPV
22	16	RB	*DL*	*0.64*	*0.698*	*0.980*	*0.684*	*0.514*	*0.613*	*0.790*	*0.844*	TPR
23	28	RB	*DL*	*1.00*	*0.697*	*0.977*	*0.798*	*0.612*	*0.402*	*0.894*	*0.736*	PPV
24	11	RB	DL	0.32	0.694	0.979	0.786	0.423	0.589	0.798	0.800	TPR
25	19	RB	DL	0.64	0.689	0.975	0.734	0.597	0.452	0.802	0.793	PPV
26	28	RB	DL	1.00	0.631	0.979	0.642	0.326	0.580	0.717	0.777	TPR
27	26	RB	DL	1.00	0.615	0.958	0.548	0.462	0.493	0.688	0.804	TPR
			DL Result		***0.682***	***0.975***	***0.718***	***0.519***	***0.514***	***0.800***	***0.796***	**PPV**
28	11	RB	TL	0.32	0.744	0.981	0.743	0.578	0.672	0.811	0.873	TPR
29	25	RB	TL	1.00	0.738	0.979	0.708	0.627	0.638	0.835	0.855	TPR
30	18	RB	TL	0.64	0.731	0.980	0.719	0.639	0.587	0.869	0.819	PPV
31	17	RB	TL	0.64	0.730	0.979	0.761	0.560	0.622	0.844	0.826	PPV
32	27	RB	TL	1.00	0.730	0.979	0.749	0.569	0.624	0.836	0.856	TPR
33	11	RB	TL	0.32	0.697	0.973	0.624	0.470	0.722	0.748	0.904	TPR
34	16	RB	TL	0.64	0.692	0.977	0.780	0.610	0.399	0.833	0.764	PPV
35	9	RB	TL	0.32	0.578	0.945	0.337	0.317	0.714	0.647	0.833	TPR
36	27	RB	TL	1.00	0.403	0.959	0.583	0.000	0.070	0.470	0.508	TPR
			TL Result		***0.671***	***0.972***	***0.667***	***0.485***	***0.561***	***0.766***	***0.804***	**TPR**
37	91	BB	HL_1	1.00	0.752	0.981	0.637	0.717	0.675	0.846	0.858	TPR
38	90	BB	HL_1	1.00	0.752	0.982	0.634	0.800	0.591	0.885	0.818	PPV
39	59	BB	HL_1	0.64	0.737	0.982	0.573	0.780	0.614	0.898	0.780	PPV
40	92	BB	HL_1	1.00	0.715	0.981	0.562	0.719	0.599	0.819	0.821	TPR
41	28	BB	HL_1	0.32	0.676	0.976	0.514	0.771	0.442	0.859	0.746	PPV
42	60	BB	HL_1	0.64	0.673	0.977	0.444	0.720	0.549	0.834	0.751	PPV
43	31	BB	HL_1	0.32	0.658	0.974	0.546	0.590	0.519	0.853	0.741	PPV

No.	R-time(m)	Loss type	Loss	Selection %	Iou	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
44	31	BB	HL_1	0.32	0.619	0.978	0.279	0.700	0.518	0.803	0.680	PPV
45	56	BB	HL_1	0.64	0.486	0.964	0.531	0.000	0.448	0.653	0.515	PPV
			HL Result		***0.674***	***0.977***	***0.524***	***0.644***	***0.551***	***0.828***	***0.746***	**PPV**
46	13	BB	BL	0.32	0.214	0.809	0.004	0.001	0.042	0.252	0.261	TPR
47	18	BB	BL	0.64	0.170	0.648	0.002	0.020	0.011	0.249	0.283	TPR
48	24	BB	BL	1.00	0.170	0.641	0.028	0.009	0.002	0.247	0.235	PPV
49	27	BB	BL	1.00	0.153	0.564	0.013	0.001	0.033	0.249	0.262	TPR
50	24	BB	BL	1.00	0.139	0.489	0.015	0.012	0.040	0.262	0.262	PPV
51	20	BB	BL	0.64	0.125	0.462	0.002	0.001	0.035	0.240	0.189	PPV
52	16	BB	BL	0.64	0.123	0.403	0.028	0.005	0.058	0.266	0.273	TPR
53	11	BB	BL	0.32	0.100	0.333	0.034	0.015	0.019	0.266	0.373	TPR
54	9	BB	BL	0.32	0.094	0.331	0.026	0.000	0.019	0.238	0.230	PPV
			BL Result		***0.143***	***0.520***	***0.017***	***0.007***	***0.029***	***0.252***	***0.263***	**TPR**
			Grand Average		***0.601***	***0.900***	***0.574***	***0.467***	***0.462***	***0.724***	***0.705***	**PPV**

Tab. G.1. Complete list of the top 3 baseline models for each loss function and selection percentage on the Medaka fish dataset. The columns $IoU_0, \ldots, IoU_3$ represent the four classes, Background, Bulbus, Atrium and Ventricle. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

G.1.2. Discrete

Loss Combination

No.	R-time(m)	Loss	Strategy	Selection %	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
1	27	BL,CE	AVG	0.64	0.765	0.983	0.788	0.675	0.613	0.885	0.844	PPV
2	25	BL,CE	MAX	0.64	0.740	0.982	0.768	0.568	0.641	0.872	0.798	PPV
3	38	BL,CE	WS	1.00	0.733	0.981	0.749	0.569	0.632	0.831	0.840	TPR
4	38	BL,CE	AVG	1.00	0.715	0.978	0.737	0.564	0.579	0.846	0.806	PPV
5	27	BL,CE	NWS	0.64	0.711	0.978	0.749	0.635	0.483	0.852	0.790	PPV
6	38	BL,CE	NWS	1.00	0.688	0.971	0.691	0.458	0.630	0.776	0.851	TPR
7	16	BL,CE	WS	0.32	0.665	0.973	0.738	0.654	0.295	0.791	0.745	PPV
8	16	BL,CE	HARMONIC	0.32	0.663	0.976	0.755	0.436	0.486	0.764	0.781	TPR
9	16	BL,CE	NWS	0.32	0.633	0.972	0.707	0.460	0.393	0.761	0.762	TPR
		BL,CE Result			***0.701***	***0.977***	***0.743***	***0.558***	***0.528***	***0.820***	***0.802***	***PPV***
10	27	BL,FL	HARMONIC	0.64	0.749	0.980	0.690	0.576	0.750	0.811	0.892	TPR
11	39	BL,FL	HARMONIC	1.00	0.729	0.977	0.796	0.508	0.635	0.848	0.843	PPV
12	27	BL,FL	NWS	0.64	0.720	0.979	0.774	0.619	0.509	0.882	0.795	PPV
13	17	BL,FL	HARMONIC	0.32	0.716	0.977	0.769	0.583	0.535	0.859	0.825	PPV
14	32	BL,FL	MIN	1.00	0.701	0.976	0.758	0.572	0.499	0.844	0.803	PPV
15	39	BL,FL	WS	1.00	0.701	0.976	0.801	0.639	0.386	0.896	0.756	PPV
16	27	BL,FL	AVG	0.64	0.681	0.973	0.721	0.402	0.630	0.801	0.798	PPV
17	17	BL,FL	WS	0.32	0.679	0.976	0.783	0.553	0.402	0.799	0.776	PPV
18	17	BL,FL	NWS	0.32	0.657	0.975	0.793	0.573	0.287	0.813	0.730	PPV
		BL,FL Result			***0.704***	***0.977***	***0.765***	***0.558***	***0.515***	***0.839***	***0.802***	***PPV***
19	26	CE,DL	WS	1.00	0.766	0.982	0.771	0.674	0.638	0.879	0.855	PPV
20	10	CE,DL	WS	0.32	0.752	0.980	0.775	0.693	0.561	0.887	0.828	PPV
21	26	CE,DL	HARMONIC	1.00	0.741	0.980	0.790	0.645	0.549	0.880	0.820	PPV
22	26	CE,DL	MAX	1.00	0.725	0.980	0.765	0.566	0.588	0.841	0.816	PPV
23	17	CE,DL	MIN	0.64	0.717	0.979	0.723	0.610	0.557	0.841	0.811	PPV
24	18	CE,DL	HARMONIC	0.64	0.715	0.978	0.723	0.578	0.582	0.834	0.843	TPR
25	10	CE,DL	MIN	0.32	0.708	0.977	0.773	0.629	0.454	0.844	0.808	PPV
26	18	CE,DL	AVG	0.64	0.707	0.979	0.724	0.631	0.493	0.878	0.763	PPV
27	10	CE,DL	MAX	0.32	0.698	0.976	0.701	0.681	0.432	0.868	0.793	PPV
		CE,DL Result			***0.725***	***0.979***	***0.749***	***0.634***	***0.539***	***0.861***	***0.815***	***PPV***
28	25	CE,TL	AVG	1.00	0.751	0.981	0.756	0.682	0.585	0.871	0.837	PPV
29	25	CE,TL	MIN	1.00	0.744	0.980	0.713	0.655	0.628	0.844	0.854	TPR
30	17	CE,TL	NWS	0.64	0.738	0.981	0.771	0.586	0.614	0.858	0.828	PPV
31	25	CE,TL	WS	1.00	0.738	0.980	0.687	0.608	0.676	0.822	0.866	TPR
32	17	CE,TL	HARMONIC	0.64	0.728	0.978	0.772	0.468	0.694	0.817	0.839	TPR
33	10	CE,TL	NWS	0.32	0.699	0.976	0.689	0.625	0.505	0.779	0.850	TPR
34	9	CE,TL	HARMONIC	0.32	0.676	0.977	0.792	0.553	0.384	0.841	0.719	PPV
35	9	CE,TL	WS	0.32	0.675	0.974	0.762	0.547	0.418	0.835	0.769	PPV
36	17	CE,TL	MAX	0.64	0.656	0.969	0.602	0.460	0.594	0.747	0.828	TPR
		CE,TL Result			***0.712***	***0.977***	***0.727***	***0.576***	***0.566***	***0.824***	***0.821***	***PPV***
37	26	DL,BL	AVG	0.64	0.764	0.981	0.761	0.654	0.658	0.888	0.847	PPV
38	26	DL,BL	HARMONIC	0.64	0.697	0.976	0.717	0.568	0.528	0.854	0.804	PPV
39	16	DL,BL	NWS	0.32	0.690	0.975	0.746	0.641	0.397	0.865	0.780	PPV
40	26	DL,BL	WS	0.64	0.654	0.975	0.659	0.592	0.389	0.900	0.681	PPV
41	38	DL,BL	AVG	1.00	0.637	0.970	0.652	0.556	0.368	0.818	0.749	PPV
42	38	DL,BL	HARMONIC	1.00	0.622	0.973	0.657	0.364	0.496	0.737	0.767	TPR
43	16	DL,BL	HARMONIC	0.32	0.600	0.975	0.697	0.280	0.446	0.691	0.733	TPR
44	38	DL,BL	NWS	1.00	0.546	0.965	0.621	0.294	0.304	0.726	0.597	PPV
45	16	DL,BL	AVG	0.32	0.477	0.971	0.352	0.307	0.280	0.660	0.514	PPV
		DL,BL Result			***0.632***	***0.974***	***0.651***	***0.473***	***0.430***	***0.793***	***0.719***	***PPV***
46	98	DL,HL_1	NWS	1.00	0.755	0.981	0.760	0.648	0.630	0.851	0.846	PPV

No.	R-time(m)	Loss	Strategy	Selection %	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
47	98	DL,HL_1	AVG	1.00	0.753	0.980	0.692	0.684	0.656	0.865	0.868	TPR
48	34	DL,HL_1	WS	0.32	0.734	0.978	0.741	0.574	0.643	0.834	0.868	TPR
49	98	DL,HL_1	WS	1.00	0.726	0.979	0.725	0.666	0.536	0.867	0.821	PPV
50	33	DL,HL_1	HARMONIC	0.32	0.715	0.978	0.688	0.615	0.581	0.834	0.844	TPR
51	34	DL,HL_1	MAX	0.32	0.708	0.978	0.667	0.593	0.595	0.835	0.813	PPV
52	63	DL,HL_1	MAX	0.64	0.708	0.982	0.806	0.473	0.570	0.816	0.788	PPV
53	63	DL,HL_1	AVG	0.64	0.664	0.976	0.701	0.442	0.536	0.771	0.790	TPR
54	64	DL,HL_1	NWS	0.64	0.633	0.974	0.722	0.357	0.478	0.764	0.713	PPV
		DL,HL_1 Result			***0.711***	***0.978***	***0.722***	***0.561***	***0.581***	***0.826***	***0.817***	***PPV***
55	27	FL,DL	HARMONIC	1.00	0.752	0.981	0.744	0.616	0.668	0.848	0.857	TPR
56	11	FL,DL	AVG	0.32	0.717	0.973	0.770	0.552	0.572	0.799	0.843	TPR
57	12	FL,DL	HARMONIC	0.32	0.712	0.977	0.696	0.583	0.590	0.817	0.864	TPR
58	19	FL,DL	HARMONIC	0.64	0.711	0.978	0.775	0.621	0.472	0.875	0.781	PPV
59	27	FL,DL	AVG	1.00	0.708	0.977	0.755	0.626	0.475	0.883	0.775	PPV
60	11	FL,DL	MAX	0.32	0.707	0.977	0.798	0.564	0.488	0.848	0.801	PPV
61	27	FL,DL	WS	1.00	0.704	0.981	0.787	0.483	0.566	0.853	0.771	PPV
62	19	FL,DL	MAX	0.64	0.679	0.979	0.730	0.422	0.585	0.800	0.791	PPV
63	19	FL,DL	NWS	0.64	0.677	0.974	0.782	0.486	0.465	0.853	0.742	PPV
		FL,DL Result			***0.707***	***0.977***	***0.760***	***0.550***	***0.542***	***0.842***	***0.803***	***PPV***
64	18	FL,TL	MAX	0.64	0.733	0.980	0.751	0.588	0.610	0.840	0.832	PPV
65	27	FL,TL	MAX	1.00	0.720	0.977	0.719	0.528	0.657	0.807	0.869	TPR
66	11	FL,TL	HARMONIC	0.32	0.711	0.972	0.678	0.470	0.723	0.754	0.910	TPR
67	27	FL,TL	AVG	1.00	0.704	0.977	0.730	0.554	0.556	0.831	0.813	PPV
68	27	FL,TL	HARMONIC	1.00	0.702	0.974	0.738	0.535	0.562	0.827	0.828	TPR
69	11	FL,TL	WS	0.32	0.687	0.975	0.759	0.552	0.461	0.839	0.788	PPV
70	11	FL,TL	AVG	0.32	0.674	0.976	0.741	0.416	0.564	0.756	0.821	TPR
71	18	FL,TL	HARMONIC	0.64	0.655	0.974	0.657	0.439	0.551	0.733	0.812	TPR
72	18	FL,TL	WS	0.64	0.628	0.973	0.753	0.405	0.380	0.768	0.727	PPV
		FL,TL Result			***0.690***	***0.975***	***0.725***	***0.499***	***0.563***	***0.795***	***0.822***	***TPR***
73	98	HL_1,CE	NWS	1.00	0.759	0.982	0.779	0.644	0.631	0.897	0.818	PPV
74	64	HL_1,CE	MAX	0.64	0.755	0.983	0.803	0.562	0.671	0.885	0.813	PPV
75	65	HL_1,CE	WS	0.64	0.748	0.980	0.757	0.692	0.564	0.880	0.850	PPV
76	98	HL_1,CE	HARMONIC	1.00	0.742	0.981	0.783	0.584	0.618	0.878	0.804	PPV
77	97	HL_1,CE	MAX	1.00	0.730	0.979	0.727	0.604	0.611	0.843	0.838	PPV
78	65	HL_1,CE	HARMONIC	0.64	0.728	0.980	0.768	0.616	0.549	0.874	0.814	PPV
79	35	HL_1,CE	MAX	0.32	0.712	0.978	0.715	0.560	0.594	0.806	0.842	TPR
80	35	HL_1,CE	HARMONIC	0.32	0.706	0.977	0.644	0.627	0.578	0.819	0.841	TPR
81	36	HL_1,CE	NWS	0.32	0.698	0.976	0.725	0.632	0.459	0.878	0.778	PPV
		HL_1,CE Result			***0.731***	***0.980***	***0.745***	***0.613***	***0.586***	***0.862***	***0.822***	***PPV***
82	66	HL_1,FL	AVG	0.64	0.766	0.984	0.757	0.602	0.723	0.880	0.839	PPV
83	100	HL_1,FL	HARMONIC	1.00	0.746	0.980	0.741	0.570	0.691	0.814	0.907	TPR
84	36	HL_1,FL	AVG	0.32	0.743	0.979	0.703	0.610	0.680	0.841	0.875	TPR
85	99	HL_1,FL	AVG	1.00	0.733	0.979	0.760	0.628	0.563	0.877	0.806	PPV
86	99	HL_1,FL	NWS	1.00	0.717	0.976	0.747	0.451	0.693	0.803	0.839	TPR
87	66	HL_1,FL	HARMONIC	0.64	0.709	0.977	0.699	0.549	0.611	0.846	0.806	PPV
88	37	HL_1,FL	HARMONIC	0.32	0.701	0.978	0.714	0.493	0.618	0.788	0.837	TPR
89	36	HL_1,FL	MIN	0.32	0.691	0.978	0.741	0.508	0.538	0.808	0.788	PPV
90	66	HL_1,FL	NWS	0.64	0.683	0.978	0.729	0.512	0.514	0.834	0.777	PPV
		HL_1,FL Result			***0.721***	***0.979***	***0.732***	***0.547***	***0.626***	***0.832***	***0.830***	***PPV***
91	38	TL,BL	WS	1.00	0.741	0.979	0.690	0.571	0.726	0.806	0.888	TPR
92	26	TL,BL	WS	0.64	0.733	0.980	0.718	0.649	0.582	0.851	0.825	PPV
93	15	TL,BL	HARMONIC	0.32	0.730	0.978	0.707	0.580	0.653	0.843	0.843	TPR
94	26	TL,BL	AVG	0.64	0.707	0.973	0.743	0.528	0.585	0.775	0.866	TPR
95	38	TL,BL	AVG	1.00	0.704	0.980	0.697	0.609	0.531	0.816	0.792	PPV
96	26	TL,BL	NWS	0.64	0.661	0.969	0.562	0.465	0.650	0.766	0.849	TPR
97	38	TL,BL	NWS	1.00	0.647	0.978	0.736	0.436	0.439	0.814	0.686	PPV

No.	R-time(m)	Loss	Strategy	Selection %	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
98	15	TL,BL	WS	0.32	0.630	0.971	0.773	0.455	0.319	0.795	0.684	PPV
99	15	TL,BL	NWS	0.32	0.453	0.963	0.454	0.275	0.120	0.709	0.601	PPV
		TL,BL Result			***0.667***	***0.975***	***0.676***	***0.508***	***0.512***	***0.797***	***0.782***	***PPV***
100	63	TL,HL_1	MAX	0.64	0.760	0.982	0.684	0.618	0.755	0.817	0.893	TPR
101	97	TL,HL_1	NWS	1.00	0.720	0.978	0.752	0.609	0.542	0.854	0.812	PPV
102	61	TL,HL_1	MIN	0.64	0.718	0.981	0.797	0.567	0.527	0.892	0.764	PPV
103	63	TL,HL_1	AVG	0.64	0.705	0.977	0.776	0.644	0.422	0.861	0.761	PPV
104	94	TL,HL_1	MIN	1.00	0.698	0.980	0.764	0.541	0.507	0.844	0.768	PPV
105	33	TL,HL_1	NWS	0.32	0.685	0.976	0.687	0.411	0.667	0.756	0.858	TPR
106	96	TL,HL_1	MAX	1.00	0.681	0.974	0.787	0.487	0.477	0.824	0.788	PPV
107	33	TL,HL_1	AVG	0.32	0.679	0.968	0.630	0.417	0.701	0.730	0.912	TPR
108	33	TL,HL_1	WS	0.32	0.671	0.980	0.703	0.416	0.586	0.767	0.804	TPR
		TL,HL_1 Result			***0.702***	***0.977***	***0.731***	***0.523***	***0.576***	***0.816***	***0.818***	***TPR***
		Grand Average			***0.700***	***0.977***	***0.727***	***0.550***	***0.547***	***0.826***	***0.804***	***PPV***

Tab. G.2. Table lists the top 3 results for every double loss combination. The column »*Strategy*« indicates which type of loss merging strategy has been used. The columns $IoU_0, \ldots, IoU_3$ represent the four classes, Background, Bulbus, Atrium and Ventricle. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

No.	R-time(m)	Loss	Strategy	Selection %	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
1	39	CE,DL,BL	HARMONIC	1.00	0.767	0.983	0.783	0.642	0.659	0.929	0.815	PPV
2	40	CE,DL,BL	NWS	1.00	0.754	0.981	0.791	0.614	0.629	0.864	0.861	PPV
3	39	CE,DL,BL	WS	1.00	0.718	0.976	0.782	0.454	0.661	0.815	0.842	TPR
4	17	CE,DL,BL	HARMONIC	0.32	0.701	0.978	0.793	0.656	0.378	0.840	0.755	PPV
5	28	CE,DL,BL	HARMONIC	0.64	0.700	0.980	0.801	0.510	0.511	0.823	0.778	PPV
6	28	CE,DL,BL	AVG	0.64	0.695	0.975	0.769	0.456	0.580	0.836	0.786	PPV
7	18	CE,DL,BL	NWS	0.32	0.694	0.973	0.795	0.530	0.475	0.821	0.815	PPV
8	28	CE,DL,BL	NWS	0.64	0.673	0.973	0.714	0.419	0.587	0.807	0.772	PPV
9	17	CE,DL,BL	WS	0.32	0.634	0.971	0.810	0.411	0.344	0.784	0.689	PPV
		CE,DL,BL Result			***0.704***	***0.977***	***0.782***	***0.521***	***0.536***	***0.836***	***0.790***	***PPV***
10	39	CE,DL,HL_1	HARMONIC	0.32	0.742	0.981	0.810	0.552	0.626	0.896	0.785	PPV
11	66	CE,DL,HL_1	MIN	0.64	0.734	0.979	0.648	0.652	0.658	0.836	0.854	TPR
12	69	CE,DL,HL_1	NWS	0.64	0.733	0.981	0.783	0.540	0.627	0.857	0.807	PPV
13	101	CE,DL,HL_1	MIN	1.00	0.712	0.978	0.765	0.583	0.522	0.839	0.798	PPV
14	103	CE,DL,HL_1	AVG	1.00	0.708	0.978	0.757	0.611	0.485	0.868	0.781	PPV
15	70	CE,DL,HL_1	HARMONIC	0.64	0.706	0.979	0.762	0.535	0.547	0.885	0.743	PPV
16	39	CE,DL,HL_1	WS	0.32	0.703	0.977	0.718	0.538	0.581	0.821	0.820	PPV
17	39	CE,DL,HL_1	AVG	0.32	0.700	0.975	0.769	0.498	0.559	0.806	0.815	TPR
18	102	CE,DL,HL_1	WS	1.00	0.686	0.977	0.733	0.578	0.457	0.826	0.762	PPV
		CE,DL,HL_1 Result			***0.714***	***0.978***	***0.750***	***0.565***	***0.562***	***0.848***	***0.796***	***PPV***
19	39	CE,TL,BL	NWS	1.00	0.765	0.984	0.775	0.630	0.672	0.847	0.861	TPR
20	28	CE,TL,BL	HARMONIC	0.64	0.749	0.982	0.765	0.602	0.647	0.880	0.822	PPV
21	17	CE,TL,BL	AVG	0.32	0.749	0.982	0.791	0.652	0.569	0.872	0.838	PPV
22	39	CE,TL,BL	AVG	1.00	0.717	0.979	0.761	0.612	0.516	0.856	0.788	PPV
23	27	CE,TL,BL	AVG	0.64	0.702	0.977	0.699	0.571	0.563	0.821	0.824	TPR
24	28	CE,TL,BL	NWS	0.64	0.690	0.974	0.679	0.441	0.667	0.773	0.837	TPR
25	39	CE,TL,BL	HARMONIC	1.00	0.664	0.977	0.764	0.503	0.415	0.785	0.729	PPV
26	17	CE,TL,BL	HARMONIC	0.32	0.651	0.973	0.810	0.582	0.241	0.790	0.707	PPV
27	17	CE,TL,BL	WS	0.32	0.612	0.978	0.623	0.527	0.320	0.872	0.695	PPV
		CE,TL,BL Result			***0.700***	***0.979***	***0.741***	***0.569***	***0.512***	***0.833***	***0.789***	***PPV***
28	102	CE,TL,HL_1	WS	1.00	0.754	0.982	0.741	0.622	0.670	0.857	0.840	PPV
29	102	CE,TL,HL_1	HARMONIC	1.00	0.749	0.980	0.691	0.612	0.714	0.835	0.868	TPR
30	68	CE,TL,HL_1	WS	0.64	0.740	0.981	0.705	0.591	0.681	0.848	0.831	PPV
31	68	CE,TL,HL_1	AVG	0.64	0.728	0.980	0.743	0.590	0.601	0.856	0.835	PPV
32	66	CE,TL,HL_1	MIN	0.64	0.717	0.981	0.796	0.518	0.572	0.866	0.765	PPV

No.	R-time(m)	Loss	Strategy	Selection %	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
33	102	CE,TL,HL_1	AVG	1.00	0.714	0.976	0.730	0.543	0.607	0.802	0.844	TPR
34	38	CE,TL,HL_1	MAX	0.32	0.697	0.978	0.678	0.461	0.670	0.770	0.872	TPR
35	39	CE,TL,HL_1	NWS	0.32	0.643	0.978	0.709	0.406	0.477	0.772	0.740	PPV
36	39	CE,TL,HL_1	AVG	0.32	0.630	0.974	0.726	0.438	0.381	0.797	0.682	PPV
		CE,TL,HL_1 Result			***0.708***	***0.979***	***0.724***	***0.531***	***0.597***	***0.822***	***0.809***	***PPV***
37	40	FL,DL,BL	AVG	1.00	0.762	0.982	0.781	0.623	0.662	0.877	0.830	PPV
38	28	FL,DL,BL	HARMONIC	0.64	0.735	0.979	0.749	0.648	0.563	0.867	0.819	PPV
39	40	FL,DL,BL	NWS	1.00	0.722	0.980	0.748	0.555	0.606	0.856	0.788	PPV
40	40	FL,DL,BL	HARMONIC	1.00	0.706	0.975	0.769	0.584	0.495	0.853	0.812	PPV
41	18	FL,DL,BL	AVG	0.32	0.699	0.977	0.732	0.602	0.484	0.832	0.789	PPV
42	18	FL,DL,BL	NWS	0.32	0.690	0.973	0.693	0.571	0.524	0.823	0.833	TPR
43	18	FL,DL,BL	WS	0.32	0.665	0.974	0.783	0.501	0.400	0.835	0.741	PPV
44	28	FL,DL,BL	AVG	0.64	0.658	0.977	0.654	0.388	0.612	0.840	0.722	PPV
45	28	FL,DL,BL	WS	0.64	0.621	0.975	0.770	0.350	0.389	0.785	0.676	PPV
		FL,DL,BL Result			***0.695***	***0.977***	***0.742***	***0.536***	***0.526***	***0.841***	***0.779***	***PPV***
46	105	FL,DL,HL_1	HARMONIC	1.00	0.743	0.981	0.746	0.638	0.609	0.871	0.836	PPV
47	106	FL,DL,HL_1	NWS	1.00	0.720	0.980	0.728	0.545	0.626	0.855	0.792	PPV
48	71	FL,DL,HL_1	WS	0.64	0.718	0.980	0.778	0.520	0.594	0.845	0.799	PPV
49	104	FL,DL,HL_1	MAX	1.00	0.706	0.977	0.738	0.635	0.474	0.863	0.796	PPV
50	71	FL,DL,HL_1	NWS	0.64	0.706	0.977	0.715	0.590	0.541	0.835	0.816	PPV
51	70	FL,DL,HL_1	MIN	0.64	0.689	0.977	0.679	0.541	0.558	0.864	0.771	PPV
52	41	FL,DL,HL_1	MAX	0.32	0.679	0.977	0.739	0.469	0.533	0.854	0.746	PPV
53	41	FL,DL,HL_1	MIN	0.32	0.660	0.974	0.794	0.524	0.346	0.822	0.713	PPV
54	42	FL,DL,HL_1	NWS	0.32	0.626	0.976	0.770	0.279	0.478	0.802	0.684	PPV
		FL,DL,HL_1 Result			***0.694***	***0.978***	***0.743***	***0.527***	***0.529***	***0.846***	***0.772***	***PPV***
55	40	FL,TL,BL	AVG	1.00	0.751	0.981	0.776	0.600	0.647	0.872	0.837	PPV
56	18	FL,TL,BL	WS	0.32	0.725	0.980	0.689	0.565	0.666	0.814	0.850	TPR
57	18	FL,TL,BL	AVG	0.32	0.709	0.973	0.632	0.518	0.712	0.763	0.910	TPR
58	18	FL,TL,BL	HARMONIC	0.32	0.696	0.974	0.741	0.506	0.563	0.812	0.840	TPR
59	28	FL,TL,BL	AVG	0.64	0.694	0.974	0.642	0.592	0.567	0.829	0.832	TPR
60	40	FL,TL,BL	HARMONIC	1.00	0.665	0.974	0.791	0.550	0.344	0.910	0.699	PPV
61	40	FL,TL,BL	WS	1.00	0.659	0.968	0.759	0.343	0.568	0.752	0.808	TPR
62	28	FL,TL,BL	NWS	0.64	0.650	0.964	0.714	0.429	0.494	0.697	0.851	TPR
63	28	FL,TL,BL	WS	0.64	0.528	0.964	0.496	0.476	0.175	0.744	0.684	PPV
		FL,TL,BL Result			***0.675***	***0.972***	***0.693***	***0.509***	***0.526***	***0.799***	***0.813***	***TPR***
64	71	FL,TL,HL_1	NWS	0.64	0.743	0.981	0.701	0.594	0.696	0.810	0.855	TPR
65	106	FL,TL,HL_1	NWS	1.00	0.732	0.979	0.737	0.604	0.607	0.833	0.847	TPR
66	104	FL,TL,HL_1	AVG	1.00	0.731	0.980	0.772	0.598	0.573	0.864	0.820	PPV
67	104	FL,TL,HL_1	MAX	1.00	0.720	0.980	0.718	0.530	0.654	0.808	0.848	TPR
68	70	FL,TL,HL_1	AVG	0.64	0.710	0.976	0.703	0.494	0.665	0.805	0.855	TPR
69	40	FL,TL,HL_1	AVG	0.32	0.701	0.973	0.711	0.470	0.648	0.786	0.871	TPR
70	70	FL,TL,HL_1	MIN	0.64	0.688	0.981	0.715	0.472	0.583	0.810	0.803	PPV
71	41	FL,TL,HL_1	HARMONIC	0.32	0.669	0.974	0.728	0.572	0.401	0.829	0.738	PPV
72	41	FL,TL,HL_1	NWS	0.32	0.654	0.978	0.683	0.448	0.506	0.763	0.742	PPV
		FL,TL,HL_1 Result			***0.705***	***0.978***	***0.719***	***0.531***	***0.593***	***0.812***	***0.820***	***TPR***
		Grand Average			***0.699***	***0.977***	***0.737***	***0.536***	***0.548***	***0.830***	***0.796***	***PPV***

Tab. G.3. Table lists the top 3 results for every triple loss combination. The column »*Strategy*« indicates which type of loss merging strategy has been used. The columns $IoU_0, \ldots, IoU_3$ represent the four classes, Background, Bulbus, Atrium and Ventricle. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

Merge Strategy

No.	R-time(m)	Loss	Strategy	Selection %	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
1	66	HL_1,FL	AVG	0.64	0.766	0.984	0.757	0.602	0.723	0.880	0.839	PPV
2	27	BL,CE	AVG	0.64	0.765	0.983	0.788	0.675	0.613	0.885	0.844	PPV
3	26	DL,BL	AVG	0.64	0.764	0.981	0.761	0.654	0.658	0.888	0.847	PPV
4	98	DL,HL_1	AVG	1.00	0.753	0.980	0.692	0.684	0.656	0.865	0.868	TPR
5	25	CE,TL	AVG	1.00	0.751	0.981	0.756	0.682	0.585	0.871	0.837	PPV
6	36	HL_1,FL	AVG	0.32	0.743	0.979	0.703	0.610	0.680	0.841	0.875	TPR
7	99	HL_1,FL	AVG	1.00	0.733	0.979	0.760	0.628	0.563	0.877	0.806	PPV
8	11	FL,DL	AVG	0.32	0.717	0.973	0.770	0.552	0.572	0.799	0.843	TPR
9	10	CE,DL	AVG	0.32	0.689	0.978	0.745	0.549	0.486	0.885	0.717	PPV
			AVG Result		***0.742***	***0.980***	***0.748***	***0.626***	***0.615***	***0.866***	***0.831***	***PPV***
10	27	FL,DL	HARMONIC	1.00	0.752	0.981	0.744	0.616	0.668	0.848	0.857	TPR
11	27	BL,FL	HARMONIC	0.64	0.749	0.980	0.690	0.576	0.750	0.811	0.892	TPR
12	100	HL_1,FL	HARMONIC	1.00	0.746	0.980	0.741	0.570	0.691	0.814	0.907	TPR
13	98	HL_1,CE	HARMONIC	1.00	0.742	0.981	0.783	0.584	0.618	0.878	0.804	PPV
14	15	TL,BL	HARMONIC	0.32	0.730	0.978	0.707	0.580	0.653	0.843	0.843	TPR
15	65	HL_1,CE	HARMONIC	0.64	0.728	0.980	0.768	0.616	0.549	0.874	0.814	PPV
16	17	CE,TL	HARMONIC	0.64	0.728	0.978	0.772	0.468	0.694	0.817	0.839	TPR
17	17	BL,FL	HARMONIC	0.32	0.716	0.977	0.769	0.583	0.535	0.859	0.825	PPV
18	33	DL,HL_1	HARMONIC	0.32	0.715	0.978	0.688	0.615	0.581	0.834	0.844	TPR
			HARMONIC Result		***0.734***	***0.979***	***0.740***	***0.579***	***0.638***	***0.842***	***0.847***	***TPR***
19	63	TL,HL_1	MAX	0.64	0.760	0.982	0.684	0.618	0.755	0.817	0.893	TPR
20	64	HL_1,CE	MAX	0.64	0.755	0.983	0.803	0.562	0.671	0.885	0.813	PPV
21	25	BL,CE	MAX	0.64	0.740	0.982	0.768	0.568	0.641	0.872	0.798	PPV
22	97	HL_1,CE	MAX	1.00	0.730	0.979	0.727	0.604	0.611	0.843	0.838	PPV
23	26	CE,DL	MAX	1.00	0.725	0.980	0.765	0.566	0.588	0.841	0.816	PPV
24	27	FL,TL	MAX	1.00	0.720	0.977	0.719	0.528	0.657	0.807	0.869	TPR
25	35	HL_1,CE	MAX	0.32	0.712	0.978	0.715	0.560	0.594	0.806	0.842	TPR
26	34	DL,HL_1	MAX	0.32	0.708	0.978	0.667	0.593	0.595	0.835	0.813	PPV
27	11	FL,DL	MAX	0.32	0.707	0.977	0.798	0.564	0.488	0.848	0.801	PPV
			MAX Result		***0.728***	***0.980***	***0.738***	***0.574***	***0.622***	***0.839***	***0.832***	***PPV***
28	25	CE,TL	MIN	1.00	0.744	0.980	0.713	0.655	0.628	0.844	0.854	TPR
29	61	TL,HL_1	MIN	0.64	0.718	0.981	0.797	0.567	0.527	0.892	0.764	PPV
30	17	CE,DL	MIN	0.64	0.717	0.979	0.723	0.610	0.557	0.841	0.811	PPV
31	10	CE,DL	MIN	0.32	0.708	0.977	0.773	0.629	0.454	0.844	0.808	PPV
32	32	BL,FL	MIN	1.00	0.701	0.976	0.758	0.572	0.499	0.844	0.803	PPV
33	94	TL,HL_1	MIN	1.00	0.698	0.980	0.764	0.541	0.507	0.844	0.768	PPV
34	36	HL_1,FL	MIN	0.32	0.691	0.978	0.741	0.508	0.538	0.808	0.788	PPV
35	66	HL_1,FL	MIN	0.64	0.675	0.980	0.717	0.507	0.497	0.815	0.804	PPV
36	11	FL,DL	MIN	0.32	0.672	0.973	0.784	0.503	0.429	0.761	0.787	TPR
			MIN Result		***0.703***	***0.978***	***0.752***	***0.566***	***0.515***	***0.833***	***0.798***	***PPV***
37	98	HL_1,CE	NWS	1.00	0.759	0.982	0.779	0.644	0.631	0.897	0.818	PPV
38	98	DL,HL_1	NWS	1.00	0.755	0.981	0.760	0.648	0.630	0.851	0.846	PPV
39	17	CE,TL	NWS	0.64	0.738	0.981	0.771	0.586	0.614	0.858	0.828	PPV
40	97	TL,HL_1	NWS	1.00	0.720	0.978	0.752	0.609	0.542	0.854	0.812	PPV
41	27	BL,FL	NWS	0.64	0.720	0.979	0.774	0.619	0.509	0.882	0.795	PPV
42	27	BL,CE	NWS	0.64	0.711	0.978	0.749	0.635	0.483	0.852	0.790	PPV
43	10	CE,TL	NWS	0.32	0.699	0.976	0.689	0.625	0.505	0.779	0.850	TPR
44	36	HL_1,CE	NWS	0.32	0.698	0.976	0.725	0.632	0.459	0.878	0.778	PPV
45	16	DL,BL	NWS	0.32	0.690	0.975	0.746	0.641	0.397	0.865	0.780	PPV
			NWS Result		***0.721***	***0.978***	***0.749***	***0.627***	***0.530***	***0.857***	***0.811***	***PPV***
46	26	CE,DL	WS	1.00	0.766	0.982	0.771	0.674	0.638	0.879	0.855	PPV
47	10	CE,DL	WS	0.32	0.752	0.980	0.775	0.693	0.561	0.887	0.828	PPV

No.	R-time(m)	Loss	Strategy	Selection %	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
48	65	HL_1,CE	WS	0.64	0.748	0.980	0.757	0.692	0.564	0.880	0.850	PPV
49	38	TL,BL	WS	1.00	0.741	0.979	0.690	0.571	0.726	0.806	0.888	TPR
50	25	CE,TL	WS	1.00	0.738	0.980	0.687	0.608	0.676	0.822	0.866	TPR
51	34	DL,HL_1	WS	0.32	0.734	0.978	0.741	0.574	0.643	0.834	0.868	TPR
52	26	TL,BL	WS	0.64	0.733	0.980	0.718	0.649	0.582	0.851	0.825	PPV
53	11	FL,TL	WS	0.32	0.687	0.975	0.759	0.552	0.461	0.839	0.788	PPV
54	27	BL,FL	WS	0.64	0.672	0.973	0.776	0.662	0.279	0.827	0.731	PPV
			WS Result		***0.730***	***0.979***	***0.742***	***0.631***	***0.570***	***0.847***	***0.833***	***PPV***
			Grand Sum		***0.726***	***0.979***	***0.745***	***0.600***	***0.582***	***0.847***	***0.825***	***PPV***

Tab. G.4. Table contains 54 trained models representing the top 3 double loss combinations for every selection percentage and discrete configuration setting. The columns $IoU_0, \ldots, IoU_3$ represent the four classes, Background, Bulbus, Atrium and Ventricle. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

No.	R-time(m)	Loss	Strategy	Selection %	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
1	40	FL,DL,BL	AVG	1.00	0.762	0.982	0.781	0.623	0.662	0.877	0.830	PPV
2	40	FL,TL,BL	AVG	1.00	0.751	0.981	0.776	0.600	0.647	0.872	0.837	PPV
3	17	CE,TL,BL	AVG	0.32	0.749	0.982	0.791	0.652	0.569	0.872	0.838	PPV
4	104	FL,TL,HL_1	AVG	1.00	0.731	0.980	0.772	0.598	0.573	0.864	0.820	PPV
5	68	CE,TL,HL_1	AVG	0.64	0.728	0.980	0.743	0.590	0.601	0.856	0.835	PPV
6	70	FL,TL,HL_1	AVG	0.64	0.710	0.976	0.703	0.494	0.665	0.805	0.855	TPR
7	18	FL,TL,BL	AVG	0.32	0.709	0.973	0.632	0.518	0.712	0.763	0.910	TPR
8	27	CE,TL,BL	AVG	0.64	0.702	0.977	0.699	0.571	0.563	0.821	0.824	TPR
9	40	FL,TL,HL_1	AVG	0.32	0.701	0.973	0.711	0.470	0.648	0.786	0.871	TPR
			AVG Result		***0.727***	***0.978***	***0.734***	***0.568***	***0.627***	***0.835***	***0.847***	***TPR***
10	39	CE,DL,BL	HARMONIC	1.00	0.767	0.983	0.783	0.642	0.659	0.929	0.815	PPV
11	102	CE,TL,HL_1	HARMONIC	1.00	0.749	0.980	0.691	0.612	0.714	0.835	0.868	TPR
12	28	CE,TL,BL	HARMONIC	0.64	0.749	0.982	0.765	0.602	0.647	0.880	0.822	PPV
13	105	FL,DL,HL_1	HARMONIC	1.00	0.743	0.981	0.746	0.638	0.609	0.871	0.836	PPV
14	39	CE,DL,HL_1	HARMONIC	0.32	0.742	0.981	0.810	0.552	0.626	0.896	0.785	PPV
15	28	FL,DL,BL	HARMONIC	0.64	0.735	0.979	0.749	0.648	0.563	0.867	0.819	PPV
16	70	CE,DL,HL_1	HARMONIC	0.64	0.706	0.979	0.762	0.535	0.547	0.885	0.743	PPV
17	17	CE,DL,BL	HARMONIC	0.32	0.701	0.978	0.793	0.656	0.378	0.840	0.755	PPV
18	18	FL,TL,BL	HARMONIC	0.32	0.696	0.974	0.741	0.506	0.563	0.812	0.840	TPR
			HARMONIC Result		***0.732***	***0.980***	***0.760***	***0.599***	***0.590***	***0.868***	***0.809***	***PPV***
19	104	FL,TL,HL_1	MAX	1.00	0.720	0.980	0.718	0.530	0.654	0.808	0.848	TPR
20	104	FL,DL,HL_1	MAX	1.00	0.706	0.977	0.738	0.635	0.474	0.863	0.796	PPV
21	39	CE,DL,HL_1	MAX	0.32	0.697	0.972	0.630	0.498	0.690	0.759	0.895	TPR
22	38	CE,TL,HL_1	MAX	0.32	0.697	0.978	0.678	0.461	0.670	0.770	0.872	TPR
23	68	CE,DL,HL_1	MAX	0.64	0.696	0.976	0.776	0.568	0.463	0.881	0.755	PPV
24	68	CE,TL,HL_1	MAX	0.64	0.690	0.975	0.639	0.428	0.719	0.751	0.882	TPR
25	70	FL,DL,HL_1	MAX	0.64	0.687	0.976	0.768	0.491	0.512	0.831	0.762	PPV
26	41	FL,DL,HL_1	MAX	0.32	0.679	0.977	0.739	0.469	0.533	0.854	0.746	PPV
27	101	CE,TL,HL_1	MAX	1.00	0.627	0.974	0.668	0.377	0.489	0.718	0.763	TPR
			MAX Result		***0.689***	***0.976***	***0.706***	***0.495***	***0.578***	***0.804***	***0.813***	***TPR***
28	66	CE,DL,HL_1	MIN	0.64	0.734	0.979	0.648	0.652	0.658	0.836	0.854	TPR
29	66	CE,TL,HL_1	MIN	0.64	0.717	0.981	0.796	0.518	0.572	0.866	0.765	PPV
30	101	CE,DL,HL_1	MIN	1.00	0.712	0.978	0.765	0.583	0.522	0.839	0.798	PPV
31	70	FL,DL,HL_1	MIN	0.64	0.689	0.977	0.679	0.541	0.558	0.864	0.771	PPV
32	41	FL,DL,HL_1	MIN	0.32	0.660	0.974	0.794	0.524	0.346	0.822	0.713	PPV
33	40	FL,TL,HL_1	MIN	0.32	0.652	0.969	0.566	0.497	0.578	0.767	0.838	TPR
34	37	CE,DL,HL_1	MIN	0.32	0.648	0.978	0.668	0.381	0.565	0.779	0.780	TPR
35	104	FL,DL,HL_1	MIN	1.00	0.643	0.974	0.781	0.402	0.414	0.756	0.732	PPV
36	104	FL,TL,HL_1	MIN	1.00	0.643	0.979	0.798	0.372	0.423	0.799	0.698	PPV
			MIN Result		***0.677***	***0.976***	***0.722***	***0.497***	***0.515***	***0.814***	***0.772***	***PPV***

No.	R-time(m)	Loss	Strategy	Selection %	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
37	39	CE,TL,BL	NWS	1.00	0.765	0.984	0.775	0.630	0.672	0.847	0.861	TPR
38	40	CE,DL,BL	NWS	1.00	0.754	0.981	0.791	0.614	0.629	0.864	0.861	PPV
39	71	FL,TL,HL_1	NWS	0.64	0.743	0.981	0.701	0.594	0.696	0.810	0.855	TPR
40	69	CE,DL,HL_1	NWS	0.64	0.733	0.981	0.783	0.540	0.627	0.857	0.807	PPV
41	106	FL,TL,HL_1	NWS	1.00	0.732	0.979	0.737	0.604	0.607	0.833	0.847	TPR
42	71	FL,DL,HL_1	NWS	0.64	0.706	0.977	0.715	0.590	0.541	0.835	0.816	PPV
43	18	CE,DL,BL	NWS	0.32	0.694	0.973	0.795	0.530	0.475	0.821	0.815	PPV
44	18	FL,DL,BL	NWS	0.32	0.690	0.973	0.693	0.571	0.524	0.823	0.833	TPR
45	18	FL,TL,BL	NWS	0.32	0.685	0.975	0.699	0.498	0.568	0.765	0.832	TPR
			NWS Result		***0.722***	***0.978***	***0.743***	***0.575***	***0.593***	***0.828***	***0.836***	***TPR***
46	102	CE,TL,HL_1	WS	1.00	0.754	0.982	0.741	0.622	0.670	0.857	0.840	PPV
47	68	CE,TL,HL_1	WS	0.64	0.740	0.981	0.705	0.591	0.681	0.848	0.831	PPV
48	18	FL,TL,BL	WS	0.32	0.725	0.980	0.689	0.565	0.666	0.814	0.850	TPR
49	39	CE,DL,BL	WS	1.00	0.718	0.976	0.782	0.454	0.661	0.815	0.842	TPR
50	71	FL,DL,HL_1	WS	0.64	0.718	0.980	0.778	0.520	0.594	0.845	0.799	PPV
51	39	CE,DL,HL_1	WS	0.32	0.703	0.977	0.718	0.538	0.581	0.821	0.820	PPV
52	106	FL,TL,HL_1	WS	1.00	0.696	0.974	0.696	0.399	0.717	0.757	0.857	TPR
53	18	FL,DL,BL	WS	0.32	0.665	0.974	0.783	0.501	0.400	0.835	0.741	PPV
54	69	CE,DL,HL_1	WS	0.64	0.659	0.973	0.797	0.535	0.331	0.815	0.727	PPV
			WS Result		***0.709***	***0.977***	***0.743***	***0.525***	***0.589***	***0.823***	***0.812***	***PPV***
			Grand Average		***0.709***	***0.978***	***0.735***	***0.543***	***0.582***	***0.829***	***0.815***	***PPV***

Tab. G.5. Table contains 54 trained models representing the top 3 triple loss combinations for every selection percentage and discrete configuration setting. The columns $IoU_0, \ldots, IoU_3$ represent the four classes, Background, Bulbus, Atrium and Ventricle. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

G.1.3. Continuous

No.	R-time(m)	Loss	Selection %	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR	pbm_alpha	pbm_I
1	20	BL,CE	0.32	0.043	0.171	0.000	0.000	0.000	0.202	0.048	PPV	6.304	0.975
2	34	BL,CE	1.00	0.035	0.000	0.006	0.000	0.136	0.036	0.308	TPR	9.241	0.942
3	26	BL,CE	0.32	0.024	0.065	0.001	0.000	0.031	0.257	0.259	TPR	9.000	0.586
		BL,CE Result		***0.034***	***0.079***	***0.002***	***0.000***	***0.056***	***0.165***	***0.205***	***TPR***	***8.182***	***0.834***
4	38	BL,FL	1.00	0.734	0.979	0.811	0.526	0.621	0.880	0.793	PPV	3.391	0.928
5	40	BL,FL	0.64	0.646	0.958	0.705	0.391	0.528	0.672	0.898	TPR	4.000	1.000
6	23	BL,FL	0.32	0.235	0.941	0.000	0.000	0.000	0.235	0.250	TPR	5.000	0.552
		BL,FL Result		***0.538***	***0.959***	***0.505***	***0.306***	***0.383***	***0.595***	***0.647***	***TPR***	***4.130***	***0.827***
7	36	CE,DL	1.00	0.739	0.985	0.785	0.545	0.640	0.843	0.829	PPV	9.000	0.904
8	46	CE,DL	1.00	0.736	0.978	0.713	0.527	0.727	0.796	0.899	TPR	10.000	0.697
9	47	CE,DL	1.00	0.734	0.980	0.752	0.574	0.629	0.862	0.819	PPV	8.000	0.940
		CE,DL Result		***0.736***	***0.981***	***0.750***	***0.548***	***0.665***	***0.834***	***0.849***	***TPR***	***9.000***	***0.847***
10	27	CE,TL	0.64	0.739	0.980	0.733	0.555	0.689	0.823	0.871	TPR	7.000	1.000
11	28	CE,TL	1.00	0.738	0.979	0.697	0.570	0.708	0.808	0.890	TPR	9.758	0.521
12	29	CE,TL	1.00	0.733	0.978	0.671	0.611	0.672	0.807	0.878	TPR	8.151	0.550
		CE,TL Result		***0.737***	***0.979***	***0.700***	***0.579***	***0.690***	***0.813***	***0.879***	***TPR***	***8.303***	***0.690***
13	57	DL,BL	1.00	0.541	0.971	0.350	0.345	0.499	0.675	0.630	PPV	2.889	0.884
14	36	DL,BL	1.00	0.532	0.966	0.408	0.515	0.239	0.783	0.564	PPV	1.818	0.575
15	35	DL,BL	0.64	0.004	0.000	0.000	0.017	0.000	0.004	0.243	TPR	9.444	0.419
		DL,BL Result		***0.359***	***0.646***	***0.253***	***0.292***	***0.246***	***0.488***	***0.479***	***PPV***	***4.717***	***0.626***
16	54	DL,HL_1	1.00	0.697	0.970	0.643	0.639	0.536	0.764	0.859	TPR	7.676	0.996
17	91	DL,HL_1	1.00	0.662	0.974	0.783	0.447	0.444	0.824	0.743	PPV	2.738	0.905
18	51	DL,HL_1	1.00	0.631	0.970	0.787	0.468	0.299	0.826	0.685	PPV	7.802	0.854
		DL,HL_1 Result		***0.663***	***0.972***	***0.737***	***0.518***	***0.426***	***0.804***	***0.762***	***PPV***	***6.072***	***0.918***
19	44	FL,DL	1.00	0.714	0.978	0.744	0.550	0.582	0.856	0.803	PPV	9.000	0.721
20	24	FL,DL	0.32	0.706	0.980	0.655	0.578	0.611	0.821	0.828	TPR	4.000	0.661
21	27	FL,DL	1.00	0.702	0.979	0.787	0.593	0.450	0.815	0.764	PPV	7.851	0.818
		FL,DL Result		***0.707***	***0.979***	***0.729***	***0.574***	***0.548***	***0.831***	***0.798***	***PPV***	***6.950***	***0.733***
22	86	FL,TL	1.00	0.750	0.980	0.720	0.594	0.705	0.826	0.899	TPR	10.000	0.611
23	41	FL,TL	1.00	0.741	0.978	0.700	0.601	0.683	0.819	0.884	TPR	8.000	0.918
24	22	FL,TL	0.64	0.734	0.978	0.701	0.520	0.737	0.796	0.897	TPR	4.163	0.879
		FL,TL Result		***0.741***	***0.979***	***0.707***	***0.572***	***0.708***	***0.814***	***0.893***	***TPR***	***7.388***	***0.803***
25	61	HL_1,CE	0.64	0.735	0.981	0.655	0.610	0.694	0.849	0.830	PPV	5.586	0.708
26	57	HL_1,CE	1.00	0.705	0.978	0.763	0.600	0.481	0.811	0.793	PPV	4.893	0.780
27	31	HL_1,CE	0.32	0.619	0.971	0.609	0.441	0.455	0.745	0.739	PPV	3.746	0.297
		HL_1,CE Result		***0.686***	***0.977***	***0.676***	***0.550***	***0.543***	***0.802***	***0.788***	***PPV***	***4.742***	***0.595***
28	56	HL_1,FL	1.00	0.700	0.977	0.732	0.573	0.517	0.865	0.778	PPV	4.892	0.646
29	94	HL_1,FL	1.00	0.676	0.976	0.768	0.542	0.418	0.890	0.720	PPV	3.787	0.705
30	59	HL_1,FL	0.64	0.665	0.974	0.661	0.408	0.619	0.785	0.809	TPR	5.876	0.975
		HL_1,FL Result		***0.680***	***0.976***	***0.721***	***0.508***	***0.518***	***0.847***	***0.769***	***PPV***	***4.852***	***0.775***
31	39	TL,BL	0.60	0.691	0.977	0.763	0.489	0.533	0.831	0.785	PPV	1.000	1.000
32	46	TL,BL	0.64	0.688	0.973	0.585	0.617	0.578	0.800	0.876	TPR	1.000	0.958
33	36	TL,BL	0.64	0.191	0.765	0.000	0.000	0.000	0.232	0.203	PPV	5.000	0.905
		TL,BL Result		***0.523***	***0.905***	***0.450***	***0.369***	***0.370***	***0.621***	***0.621***	***TPR***	***2.333***	***0.954***
34	59	TL,HL_1	0.64	0.596	0.965	0.617	0.296	0.505	0.709	0.760	TPR	6.428	0.677
35	86	TL,HL_1	1.00	0.576	0.950	0.567	0.275	0.511	0.639	0.840	TPR	7.543	0.351
36	85	TL,HL_1	1.00	0.525	0.932	0.395	0.217	0.554	0.569	0.842	TPR	7.886	0.359
		TL,HL_1 Result		***0.565***	***0.949***	***0.526***	***0.262***	***0.523***	***0.639***	***0.814***	***TPR***	***7.286***	***0.462***
		Grand Average		***0.581***	***0.865***	***0.563***	***0.423***	***0.473***	***0.688***	***0.709***	***TPR***	***6.163***	***0.755***

Tab. G.6. Table lists the top 3 results for every double loss combination using the performance-based continuous merge strategy. The columns $IoU_0, \ldots, IoU_3$ represent the four classes, Background, Bulbus, Atrium and Ventricle. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

No.	R-time(m)	Loss	Selection %	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR	pbm_alpha	pbm_I
1	53	CE,DL,BL	1.00	0.738	0.980	0.718	0.631	0.623	0.840	0.846	TPR	4.993	0.979
2	35	CE,DL,BL	0.64	0.477	0.874	0.079	0.481	0.475	0.547	0.768	TPR	8.772	0.258
		CE,DL,BL Result		***0.608***	***0.927***	***0.398***	***0.556***	***0.549***	***0.694***	***0.807***	***TPR***	***6.883***	***0.619***
3	97	CE,DL,HL_1	1.00	0.723	0.980	0.762	0.580	0.569	0.857	0.810	PPV	4.227	0.989
4	61	CE,DL,HL_1	0.64	0.590	0.971	0.570	0.372	0.445	0.767	0.649	PPV	5.448	0.486
5	27	CE,DL,HL_1	0.32	0.235	0.941	0.000	0.000	0.000	0.235	0.250	TPR	8.000	0.000
		CE,DL,HL_1 Result		***0.516***	***0.964***	***0.444***	***0.317***	***0.338***	***0.620***	***0.570***	***PPV***	***5.892***	***0.492***
6	55	CE,TL,BL	1.00	0.772	0.983	0.771	0.653	0.682	0.878	0.851	PPV	4.000	0.905
7	60	CE,TL,BL	1.00	0.732	0.977	0.682	0.600	0.668	0.796	0.885	TPR	8.000	0.773
8	54	CE,TL,BL	1.00	0.597	0.969	0.494	0.436	0.490	0.730	0.769	TPR	6.668	0.254
		CE,TL,BL Result		***0.700***	***0.976***	***0.649***	***0.563***	***0.613***	***0.801***	***0.835***	***TPR***	***6.223***	***0.644***
9	94	CE,TL,HL_1	1.00	0.745	0.982	0.801	0.579	0.617	0.873	0.820	PPV	5.223	0.261
10	31	CE,TL,HL_1	0.32	0.702	0.972	0.583	0.508	0.747	0.761	0.921	TPR	6.131	0.150
11	35	CE,TL,HL_1	0.32	0.554	0.923	0.458	0.219	0.615	0.614	0.910	TPR	2.000	0.877
		CE,TL,HL_1 Result		***0.667***	***0.959***	***0.614***	***0.435***	***0.660***	***0.749***	***0.884***	***TPR***	***4.451***	***0.429***
12	36	FL,DL,BL	0.64	0.737	0.981	0.725	0.549	0.695	0.852	0.822	PPV	1.000	0.991
13	57	FL,DL,BL	1.00	0.710	0.973	0.724	0.484	0.657	0.803	0.877	TPR	8.000	0.585
14	40	FL,DL,BL	1.00	0.687	0.978	0.793	0.444	0.533	0.888	0.732	PPV	4.214	0.715
		FL,DL,BL Result		***0.711***	***0.978***	***0.747***	***0.492***	***0.628***	***0.848***	***0.810***	***PPV***	***4.405***	***0.764***
15	64	FL,DL,HL_1	0.64	0.737	0.980	0.738	0.576	0.655	0.837	0.846	TPR	6.797	0.466
16	91	FL,DL,HL_1	1.00	0.676	0.975	0.760	0.512	0.458	0.831	0.732	PPV	8.160	0.151
17	91	FL,DL,HL_1	1.00	0.665	0.968	0.549	0.487	0.657	0.742	0.866	TPR	9.071	0.037
		FL,DL,HL_1 Result		***0.693***	***0.975***	***0.682***	***0.525***	***0.590***	***0.804***	***0.815***	***TPR***	***8.010***	***0.218***
18	24	FL,TL,BL	0.32	0.007	0.000	0.000	0.000	0.030	0.007	0.250	TPR	3.369	0.045
		FL,TL,BL Result		***0.007***	***0.000***	***0.000***	***0.000***	***0.030***	***0.007***	***0.250***	***TPR***	***3.369***	***0.045***
19	56	FL,TL,HL_1	1.00	0.755	0.982	0.764	0.626	0.649	0.855	0.854	PPV	9.396	0.902
20	87	FL,TL,HL_1	1.00	0.695	0.978	0.774	0.415	0.614	0.797	0.819	TPR	3.000	0.702
21	32	FL,TL,HL_1	0.32	0.631	0.970	0.551	0.467	0.535	0.715	0.816	TPR	5.306	0.059
		FL,TL,HL_1 Result		***0.694***	***0.977***	***0.696***	***0.503***	***0.599***	***0.789***	***0.830***	***TPR***	***5.901***	***0.554***
		Grand Average		***0.627***	***0.921***	***0.585***	***0.458***	***0.543***	***0.725***	***0.766***	***TPR***	***5.799***	***0.504***

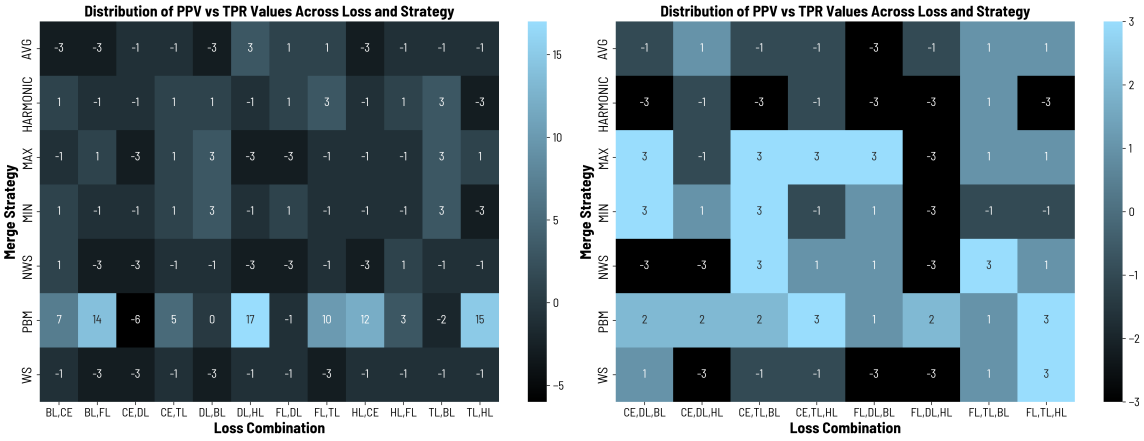
Tab. G.7. Table lists the top 3 results for every triple loss combination using the performance-based continuous merge strategy. The columns $IoU_0, \ldots, IoU_3$ represent the four classes, Background, Bulbus, Atrium and Ventricle. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

G.1.4. Specific

No.	R-time(m)	Loss	Selection %	Strategy	IoU	IoU 0	IoU 1	IoU 2	IoU 3	PPV	TPR	PPV vs. TPR
1	115	FL,DL,BL	1.00	PBM	0.742	0.981	0.757	0.629	0.601	0.860	0.829	PPV
2	108	FL,DL,BL	1.00	PBM	0.717	0.978	0.778	0.556	0.556	0.845	0.824	PPV
3	113	FL,DL,BL	1.00	PBM	0.711	0.976	0.779	0.534	0.556	0.815	0.840	TPR
4	62	FL,DL,BL	1.00	PBM	0.696	0.974	0.644	0.539	0.628	0.773	0.859	TPR
5	128	FL,DL,BL	1.00	PBM	0.693	0.980	0.760	0.399	0.632	0.799	0.814	TPR
6	110	FL,DL,BL	1.00	PBM	0.649	0.974	0.605	0.456	0.561	0.757	0.816	TPR
7	74	FL,DL,BL	0.64	PBM	0.769	0.982	0.790	0.666	0.637	0.877	0.860	PPV
8	45	FL,DL,BL	0.64	PBM	0.761	0.982	0.780	0.584	0.696	0.862	0.850	PPV
9	68	FL,DL,BL	0.64	PBM	0.759	0.983	0.799	0.653	0.602	0.900	0.812	PPV
10	74	FL,DL,BL	0.64	PBM	0.725	0.979	0.772	0.555	0.594	0.872	0.807	PPV
11	76	FL,DL,BL	0.64	PBM	0.709	0.975	0.630	0.578	0.654	0.814	0.867	TPR
12	61	FL,DL,BL	0.64	PBM	0.707	0.980	0.705	0.564	0.580	0.820	0.823	TPR
13	14	FL,DL,BL	0.32	PBM	0.712	0.979	0.667	0.561	0.642	0.825	0.823	PPV
14	43	FL,DL,BL	0.32	PBM	0.699	0.979	0.695	0.592	0.531	0.877	0.767	PPV
15	41	FL,DL,BL	0.32	PBM	0.694	0.975	0.728	0.638	0.434	0.841	0.802	PPV
16	44	FL,DL,BL	0.32	PBM	0.688	0.978	0.762	0.498	0.512	0.799	0.792	PPV
17	14	FL,DL,BL	0.32	PBM	0.677	0.978	0.742	0.443	0.546	0.800	0.812	PPV
18	16	FL,DL,BL	0.32	PBM	0.407	0.969	0.219	0.017	0.421	0.562	0.488	PPV
				Grand Average	***0.695***	***0.978***	***0.701***	***0.526***	***0.577***	***0.816***	***0.805***	***PPV***

Tab. G.8. Table lists six models for each selection percentage trained repetitively on the loss combination FL, DL and BL using the arithmetic averaging (AVG) strategy for merging. These results serve as part of a final comparison against baseline results providing an identical iteration number. The columns $IoU_0, \ldots, IoU_3$ represent the four classes, Background, Bulbus, Atrium and Ventricle. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

G.1.5. PPV vs. TPR

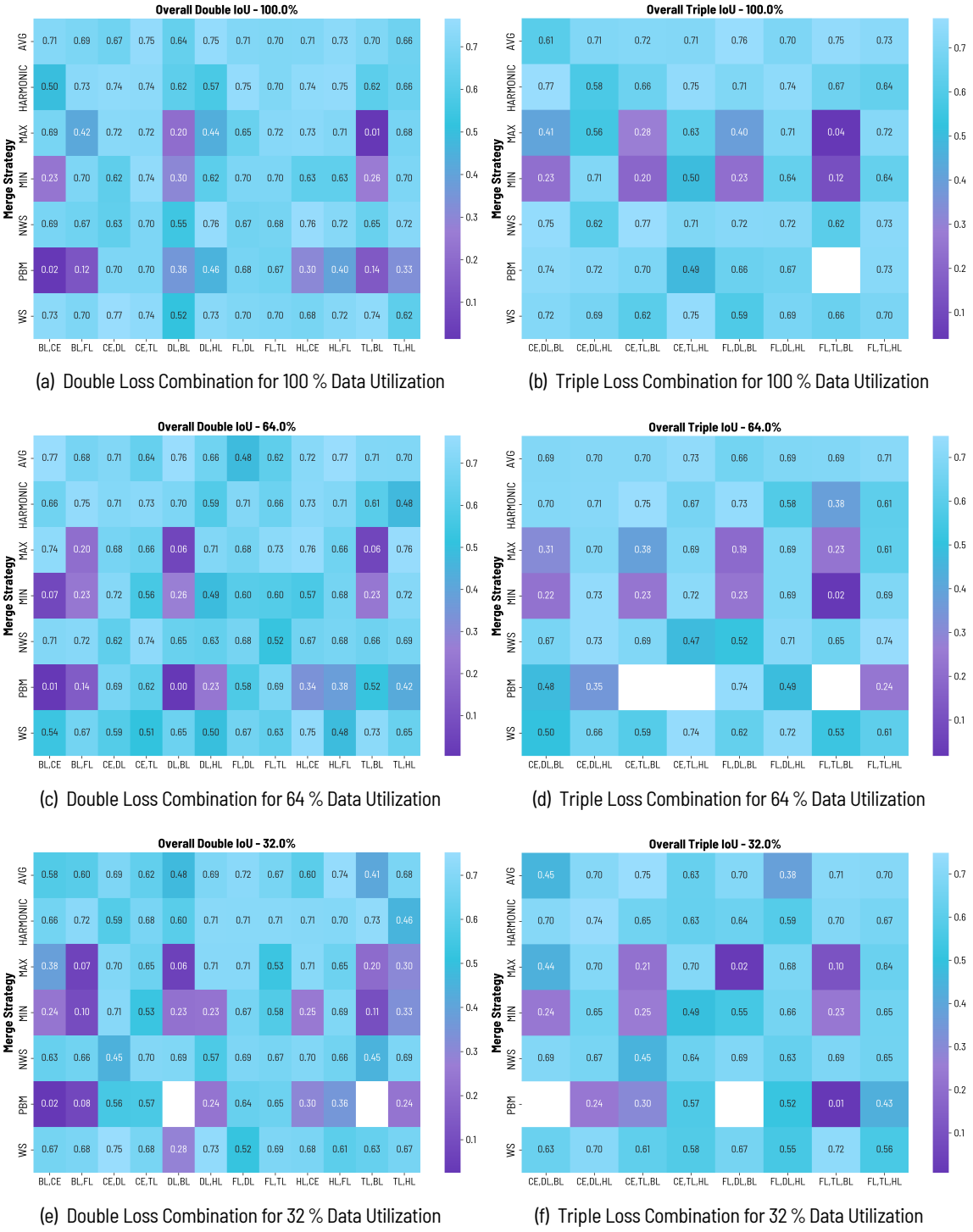


(a) Distribution of PPV vs. TPR for double losses

(b) Distribution of PPV vs. TPR for triple losses

Fig. G.1. Heatmaps visualizing the relative prevalence as a difference map of PPV and TPR across various loss combinations and merge strategies. Each cell represents a unique combination. Positive values indicate a predominance of TPR, while negative values signify a predominance of PPV.

G.1.6. Ablation study



(a) Double Loss Combination for 100 % Data Utilization

(b) Triple Loss Combination for 100 % Data Utilization

(c) Double Loss Combination for 64 % Data Utilization

(d) Triple Loss Combination for 64 % Data Utilization

(e) Double Loss Combination for 32 % Data Utilization

(f) Triple Loss Combination for 32 % Data Utilization

Fig. G.2. Overall results generated from different dataset sizes. The left column (a,c,e) illustrates results from double loss combinations, while the right column (b,d,f) depicts results from triple loss combinations.

G.2. Skin lesion

G.2.1. Baseline

No.	R-time(m)	Loss type	Loss	selection %	Iou	PPV	TPR	PPV vs. TPR
1	55	DB	*CE*	0.64	*0.667*	*0.812*	*0.824*	TPR
2	29	DB	*CE*	0.32	*0.640*	*0.780*	*0.819*	TPR
3	29	DB	*CE*	0.32	*0.626*	*0.807*	*0.752*	PPV
4	29	DB	*CE*	0.32	*0.584*	*0.745*	*0.699*	PPV
5	63	DB	*CE*	0.64	*0.580*	*0.752*	*0.704*	PPV
6	83	DB	*CE*	1.00	*0.532*	*0.753*	*0.670*	PPV
7	97	DB	*CE*	1.00	*0.528*	*0.769*	*0.641*	PPV
8	55	DB	CE	0.64	0.528	0.753	0.643	PPV
9	83	DB	CE	1.00	0.500	0.738	0.611	PPV
			CE Result		***0.576***	***0.768***	***0.707***	**PPV**
10	55	DB	FL	0.64	0.666	0.841	0.780	PPV
11	84	DB	FL	1.00	0.657	0.829	0.788	PPV
12	83	DB	FL	1.00	0.623	0.821	0.727	PPV
13	55	DB	FL	0.64	0.623	0.784	0.799	TPR
14	83	DB	FL	1.00	0.618	0.822	0.718	PPV
15	54	DB	FL	0.64	0.607	0.814	0.746	PPV
16	31	DB	FL	0.32	0.589	0.778	0.708	PPV
17	31	DB	FL	0.32	0.588	0.789	0.742	PPV
18	30	DB	FL	0.32	0.586	0.754	0.752	PPV
			FL Result		***0.618***	***0.803***	***0.751***	**PPV**
19	85	RB	DL	1.00	0.659	0.811	0.807	PPV
20	84	RB	DL	1.00	0.657	0.802	0.812	TPR
21	56	RB	DL	0.64	0.650	0.785	0.800	TPR
22	100	RB	DL	1.00	0.643	0.787	0.827	TPR
23	29	RB	DL	0.32	0.624	0.779	0.800	TPR
24	54	RB	DL	0.64	0.598	0.801	0.728	PPV
25	55	RB	DL	0.64	0.585	0.797	0.686	PPV
26	35	RB	DL	0.32	0.504	0.697	0.629	PPV
27	28	RB	DL	0.32	0.427	0.523	0.550	TPR
			DL Result		***0.594***	***0.754***	***0.738***	**PPV**
28	86	RB	TL	1.00	0.660	0.795	0.833	TPR
29	99	RB	TL	1.00	0.647	0.808	0.794	PPV
30	55	RB	TL	0.64	0.646	0.791	0.818	TPR
31	84	RB	TL	1.00	0.628	0.783	0.821	TPR
32	28	RB	TL	0.32	0.627	0.761	0.759	PPV
33	54	RB	TL	0.64	0.605	0.788	0.771	PPV

No.	R-time(m)	Loss type	Loss	selection %	Iou	PPV	TPR	PPV vs. TPR
34	34	RB	TL	0.32	0.600	0.784	0.733	PPV
35	64	RB	TL	0.64	0.591	0.765	0.728	PPV
36	29	RB	TL	0.32	0.563	0.726	0.712	PPV
			TL Result		***0.618***	***0.778***	***0.774***	**PPV**
37	141	BB	HL_1	1.00	0.647	0.816	0.801	PPV
38	92	BB	HL_1	0.64	0.625	0.801	0.786	PPV
39	46	BB	HL_1	0.32	0.599	0.809	0.730	PPV
40	139	BB	HL_1	1.00	0.595	0.793	0.741	PPV
41	91	BB	HL_1	0.64	0.594	0.771	0.784	TPR
42	91	BB	HL_1	0.64	0.593	0.823	0.727	PPV
43	47	BB	HL_1	0.32	0.584	0.756	0.747	PPV
44	48	BB	HL_1	0.32	0.574	0.760	0.737	PPV
45	213	BB	HL_1	1.00	0.566	0.758	0.726	PPV
			HL_1 Result		***0.597***	***0.788***	***0.753***	**PPV**
46	69	BB	BL	1.00	0.384	0.560	0.526	PPV
47	24	BB	BL	0.32	0.382	0.533	0.555	TPR
48	104	BB	BL	1.00	0.382	0.516	0.497	PPV
49	24	BB	BL	0.32	0.375	0.603	0.664	TPR
50	25	BB	BL	0.32	0.371	0.545	0.508	PPV
51	65	BB	BL	1.00	0.338	0.523	0.532	TPR
52	68	BB	BL	0.64	0.266	0.509	0.515	TPR
53	44	BB	BL	0.64	0.238	0.446	0.450	TPR
54	44	BB	BL	0.64	0.229	0.472	0.478	TPR
			BL Result		***0.329***	***0.523***	***0.525***	**TPR**
			Grand Average		***0.556***	***0.736***	***0.708***	**PPV**

Tab. G.9. The table presents the average values of various metrics of 3 baseline models for each loss function. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

G.2.2. Discrete

Loss Combination

No.	R-time(m)	Loss	Strategy	Selection %	IoU	PPV	TPR	PPV vs. TPR
1	101	BL,CE	HARMONIC	0.64	0.640	0.814	0.791	PPV
2	152	BL,CE	NWS	1.00	0.637	0.803	0.778	PPV
3	152	BL,CE	WS	1.00	0.631	0.818	0.762	PPV
4	152	BL,CE	HARMONIC	1.00	0.618	0.803	0.738	PPV
5	101	BL,CE	NWS	0.64	0.600	0.795	0.719	PPV
6	56	BL,CE	WS	0.32	0.583	0.789	0.704	PPV
7	56	BL,CE	NWS	0.32	0.578	0.723	0.786	TPR
8	101	BL,CE	AVG	0.64	0.568	0.767	0.708	PPV
9	56	BL,CE	AVG	0.32	0.524	0.709	0.649	PPV
		BL,CE Result			***0.598***	***0.780***	***0.737***	***PPV***
10	102	BL,FL	HARMONIC	0.64	0.644	0.842	0.762	PPV
11	102	BL,FL	AVG	0.64	0.633	0.792	0.792	TPR
12	102	BL,FL	NWS	0.64	0.631	0.811	0.758	PPV
13	153	BL,FL	AVG	1.00	0.599	0.805	0.712	PPV
14	56	BL,FL	NWS	0.32	0.584	0.736	0.719	PPV
15	152	BL,FL	NWS	1.00	0.583	0.742	0.791	TPR
16	56	BL,FL	WS	0.32	0.582	0.745	0.721	PPV
17	56	BL,FL	HARMONIC	0.32	0.580	0.795	0.696	PPV
18	153	BL,FL	HARMONIC	1.00	0.568	0.790	0.704	PPV
		BL,FL Result			***0.600***	***0.784***	***0.739***	***PPV***
19	109	CE,DL	AVG	1.00	0.662	0.820	0.810	PPV
20	109	CE,DL	NWS	1.00	0.657	0.818	0.808	PPV
21	72	CE,DL	HARMONIC	0.64	0.643	0.829	0.765	PPV
22	72	CE,DL	WS	0.64	0.629	0.789	0.786	PPV
23	109	CE,DL	MIN	1.00	0.624	0.832	0.751	PPV
24	38	CE,DL	MIN	0.32	0.564	0.773	0.693	PPV
25	39	CE,DL	AVG	0.32	0.560	0.736	0.738	TPR
26	71	CE,DL	AVG	0.64	0.559	0.780	0.665	PPV
27	38	CE,DL	MAX	0.32	0.538	0.722	0.675	PPV
		CE,DL Result			***0.604***	***0.789***	***0.744***	***PPV***
28	109	CE,TL	AVG	1.00	0.642	0.821	0.763	PPV
29	71	CE,TL	NWS	0.64	0.639	0.821	0.774	PPV
30	108	CE,TL	MAX	1.00	0.627	0.835	0.749	PPV
31	109	CE,TL	WS	1.00	0.624	0.804	0.756	PPV
32	70	CE,TL	MIN	0.64	0.618	0.770	0.773	TPR
33	36	CE,TL	MIN	0.32	0.585	0.731	0.754	TPR

No.	R-time(m)	Loss	Strategy	Selection %	IoU	PPV	TPR	PPV vs. TPR
34	37	CE,TL	NWS	0.32	0.584	0.762	0.717	PPV
35	71	CE,TL	HARMONIC	0.64	0.563	0.801	0.674	PPV
36	37	CE,TL	HARMONIC	0.32	0.539	0.719	0.655	PPV
		CE,TL Result			***0.602***	***0.785***	***0.735***	***PPV***
37	152	DL,BL	NWS	1.00	0.671	0.817	0.830	TPR
38	152	DL,BL	WS	1.00	0.642	0.794	0.793	PPV
39	88	DL,BL	MIN	0.64	0.632	0.801	0.778	PPV
40	132	DL,BL	MIN	1.00	0.621	0.787	0.790	TPR
41	55	DL,BL	WS	0.32	0.608	0.790	0.775	PPV
42	101	DL,BL	AVG	0.64	0.588	0.793	0.698	PPV
43	55	DL,BL	HARMONIC	0.32	0.579	0.777	0.767	PPV
44	55	DL,BL	NWS	0.32	0.550	0.775	0.690	PPV
45	101	DL,BL	WS	0.64	0.505	0.708	0.626	PPV
		DL,BL Result			***0.600***	***0.782***	***0.750***	***PPV***
46	233	DL,HL_1	WS	1.00	0.625	0.827	0.742	PPV
47	232	DL,HL_1	HARMONIC	1.00	0.613	0.821	0.736	PPV
48	151	DL,HL_1	WS	0.64	0.595	0.799	0.738	PPV
49	229	DL,HL_1	MIN	1.00	0.580	0.754	0.719	PPV
50	80	DL,HL_1	HARMONIC	0.32	0.571	0.761	0.693	PPV
51	80	DL,HL_1	NWS	0.32	0.559	0.770	0.700	PPV
52	80	DL,HL_1	AVG	0.32	0.555	0.732	0.754	TPR
53	151	DL,HL_1	AVG	0.64	0.510	0.729	0.617	PPV
54	150	DL,HL_1	MAX	0.64	0.463	0.667	0.574	PPV
		DL,HL_1 Result			***0.563***	***0.762***	***0.697***	***PPV***
55	74	FL,DL	MIN	0.64	0.658	0.823	0.806	PPV
56	41	FL,DL	MIN	0.32	0.648	0.800	0.798	PPV
57	112	FL,DL	NWS	1.00	0.640	0.816	0.768	PPV
58	111	FL,DL	MAX	1.00	0.639	0.836	0.768	PPV
59	75	FL,DL	HARMONIC	0.64	0.600	0.792	0.735	PPV
60	41	FL,DL	WS	0.32	0.596	0.786	0.768	PPV
61	41	FL,DL	HARMONIC	0.32	0.587	0.796	0.726	PPV
62	112	FL,DL	HARMONIC	1.00	0.556	0.797	0.690	PPV
63	74	FL,DL	NWS	0.64	0.547	0.773	0.655	PPV
		FL,DL Result			***0.608***	***0.802***	***0.746***	***PPV***
64	74	FL,TL	WS	0.64	0.643	0.825	0.764	PPV
65	112	FL,TL	AVG	1.00	0.607	0.791	0.740	PPV
66	112	FL,TL	WS	1.00	0.604	0.772	0.770	PPV
67	74	FL,TL	NWS	0.64	0.603	0.773	0.771	PPV
68	40	FL,TL	MAX	0.32	0.597	0.735	0.755	TPR
69	40	FL,TL	WS	0.32	0.570	0.792	0.708	PPV
70	40	FL,TL	AVG	0.32	0.559	0.725	0.681	PPV
71	73	FL,TL	MAX	0.64	0.545	0.777	0.656	PPV
72	113	FL,TL	HARMONIC	1.00	0.540	0.789	0.647	PPV
		FL,TL Result			***0.585***	***0.776***	***0.721***	***PPV***

No.	R-time(m)	Loss	Strategy	Selection %	IoU	PPV	TPR	PPV vs. TPR
73	232	HL_1,CE	WS	1.00	0.642	0.797	0.819	TPR
74	152	HL_1,CE	NWS	0.64	0.631	0.800	0.785	PPV
75	152	HL_1,CE	MIN	0.64	0.622	0.800	0.780	PPV
76	81	HL_1,CE	HARMONIC	0.32	0.602	0.764	0.758	PPV
77	82	HL_1,CE	MAX	0.32	0.586	0.768	0.760	PPV
78	232	HL_1,CE	MAX	1.00	0.581	0.730	0.780	TPR
79	152	HL_1,CE	WS	0.64	0.580	0.784	0.731	PPV
80	81	HL_1,CE	WS	0.32	0.563	0.797	0.706	PPV
81	231	HL_1,CE	MIN	1.00	0.538	0.787	0.642	PPV
		HL_1,CE Result			***0.594***	***0.781***	***0.751***	***PPV***
82	152	HL_1,FL	MAX	0.64	0.636	0.775	0.839	TPR
83	232	HL_1,FL	MIN	1.00	0.619	0.822	0.742	PPV
84	81	HL_1,FL	MIN	0.32	0.610	0.773	0.758	PPV
85	233	HL_1,FL	WS	1.00	0.601	0.783	0.723	PPV
86	152	HL_1,FL	MIN	0.64	0.597	0.762	0.725	PPV
87	153	HL_1,FL	AVG	0.64	0.577	0.799	0.709	PPV
88	82	HL_1,FL	NWS	0.32	0.561	0.777	0.685	PPV
89	82	HL_1,FL	HARMONIC	0.32	0.560	0.751	0.685	PPV
90	231	HL_1,FL	MAX	1.00	0.552	0.775	0.687	PPV
		HL_1,FL Result			***0.590***	***0.780***	***0.728***	***PPV***
91	152	TL,BL	WS	1.00	0.669	0.832	0.805	PPV
92	151	TL,BL	AVG	1.00	0.653	0.825	0.776	PPV
93	153	TL,BL	HARMONIC	1.00	0.633	0.794	0.780	PPV
94	100	TL,BL	WS	0.64	0.613	0.778	0.758	PPV
95	54	TL,BL	NWS	0.32	0.585	0.749	0.739	PPV
96	54	TL,BL	AVG	0.32	0.577	0.752	0.735	PPV
97	55	TL,BL	HARMONIC	0.32	0.567	0.774	0.720	PPV
98	100	TL,BL	NWS	0.64	0.540	0.750	0.656	PPV
99	88	TL,BL	MIN	0.64	0.487	0.708	0.594	PPV
		TL,BL Result			***0.592***	***0.774***	***0.729***	***PPV***
100	151	TL,HL_1	MIN	0.64	0.645	0.795	0.817	TPR
101	152	TL,HL_1	NWS	0.64	0.617	0.785	0.786	TPR
102	231	TL,HL_1	HARMONIC	1.00	0.612	0.810	0.729	PPV
103	80	TL,HL_1	MAX	0.32	0.596	0.788	0.745	PPV
104	229	TL,HL_1	MIN	1.00	0.583	0.761	0.724	PPV
105	80	TL,HL_1	AVG	0.32	0.543	0.791	0.688	PPV
106	80	TL,HL_1	WS	0.32	0.539	0.797	0.687	PPV
107	150	TL,HL_1	MAX	0.64	0.518	0.777	0.625	PPV
108	232	TL,HL_1	AVG	1.00	0.517	0.728	0.711	PPV
		TL,HL_1 Result			***0.575***	***0.781***	***0.723***	***PPV***
		Grand Average			***0.593***	***0.781***	***0.733***	***PPV***

Tab. G.10. Table lists the top 3 results for every double loss combination. The column »*Strategy*« indicates which type of loss merging strategy has been used. The »*IoU*« column reflects the average outcomes contrasting the foreground class against the background class. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

No.	R-time(m)	Loss	Strategy	Selection %	IoU	PPV	TPR	PPV vs. TPR
1	89	CE,DL,BL	MIN	0.64	0.651	0.786	0.834	TPR
2	101	CE,DL,BL	HARMONIC	0.64	0.613	0.787	0.738	PPV
3	101	CE,DL,BL	WS	0.64	0.613	0.770	0.788	TPR
4	154	CE,DL,BL	AVG	1.00	0.611	0.821	0.717	PPV
5	155	CE,DL,BL	HARMONIC	1.00	0.601	0.820	0.706	PPV
6	154	CE,DL,BL	WS	1.00	0.594	0.798	0.725	PPV
7	49	CE,DL,BL	MIN	0.32	0.588	0.743	0.812	TPR
8	54	CE,DL,BL	HARMONIC	0.32	0.583	0.757	0.762	TPR
9	54	CE,DL,BL	WS	0.32	0.569	0.779	0.695	PPV
		CE,DL,BL Result			***0.603***	***0.784***	***0.753***	***PPV***
10	149	CE,DL,HL_1	MIN	0.64	0.634	0.804	0.792	PPV
11	77	CE,DL,HL_1	MAX	0.32	0.633	0.816	0.757	PPV
12	231	CE,DL,HL_1	HARMONIC	1.00	0.621	0.808	0.746	PPV
13	232	CE,DL,HL_1	WS	1.00	0.614	0.793	0.758	PPV
14	151	CE,DL,HL_1	HARMONIC	0.64	0.610	0.786	0.779	PPV
15	77	CE,DL,HL_1	HARMONIC	0.32	0.608	0.765	0.756	PPV
16	77	CE,DL,HL_1	AVG	0.32	0.586	0.793	0.738	PPV
17	230	CE,DL,HL_1	MIN	1.00	0.586	0.764	0.719	PPV
18	150	CE,DL,HL_1	AVG	0.64	0.576	0.795	0.692	PPV
		CE,DL,HL_1 Result			***0.608***	***0.792***	***0.749***	***PPV***
19	99	CE,TL,BL	AVG	0.64	0.648	0.816	0.797	PPV
20	132	CE,TL,BL	MIN	1.00	0.632	0.812	0.750	PPV
21	153	CE,TL,BL	AVG	1.00	0.630	0.816	0.754	PPV
22	101	CE,TL,BL	NWS	0.64	0.629	0.794	0.788	PPV
23	153	CE,TL,BL	WS	1.00	0.628	0.800	0.770	PPV
24	100	CE,TL,BL	HARMONIC	0.64	0.597	0.759	0.777	TPR
25	52	CE,TL,BL	NWS	0.32	0.579	0.767	0.727	PPV
26	47	CE,TL,BL	MIN	0.32	0.574	0.746	0.777	TPR
27	52	CE,TL,BL	HARMONIC	0.32	0.561	0.792	0.683	PPV
		CE,TL,BL Result			***0.609***	***0.789***	***0.758***	***PPV***
28	231	CE,TL,HL_1	HARMONIC	1.00	0.672	0.808	0.837	TPR
29	232	CE,TL,HL_1	WS	1.00	0.636	0.828	0.762	PPV
30	230	CE,TL,HL_1	MAX	1.00	0.620	0.818	0.765	PPV
31	75	CE,TL,HL_1	AVG	0.32	0.590	0.802	0.722	PPV
32	76	CE,TL,HL_1	NWS	0.32	0.564	0.747	0.692	PPV
33	148	CE,TL,HL_1	MIN	0.64	0.551	0.790	0.673	PPV
34	148	CE,TL,HL_1	MAX	0.64	0.538	0.761	0.657	PPV
35	149	CE,TL,HL_1	HARMONIC	0.64	0.537	0.733	0.697	PPV
36	74	CE,TL,HL_1	MAX	0.32	0.536	0.740	0.676	PPV
		CE,TL,HL_1 Result			***0.583***	***0.781***	***0.720***	***PPV***
37	159	FL,DL,BL	HARMONIC	1.00	0.670	0.799	0.851	TPR
38	104	FL,DL,BL	HARMONIC	0.64	0.648	0.815	0.808	PPV
39	148	FL,DL,BL	MIN	1.00	0.623	0.800	0.775	PPV
40	160	FL,DL,BL	NWS	1.00	0.610	0.808	0.723	PPV
41	103	FL,DL,BL	WS	0.64	0.609	0.809	0.761	PPV

No.	R-time(m)	Loss	Strategy	Selection %	IoU	PPV	TPR	PPV vs. TPR
42	56	FL,DL,BL	HARMONIC	0.32	0.592	0.797	0.719	PPV
43	53	FL,DL,BL	MIN	0.32	0.586	0.742	0.732	PPV
44	56	FL,DL,BL	WS	0.32	0.582	0.760	0.725	PPV
45	104	FL,DL,BL	AVG	0.64	0.534	0.731	0.639	PPV
		FL,DL,BL Result			***0.606***	***0.784***	***0.748***	***PPV***
46	233	FL,DL,HL_1	MIN	1.00	0.652	0.806	0.818	TPR
47	80	FL,DL,HL_1	WS	0.32	0.640	0.817	0.784	PPV
48	151	FL,DL,HL_1	MIN	0.64	0.631	0.815	0.769	PPV
49	235	FL,DL,HL_1	NWS	1.00	0.626	0.799	0.769	PPV
50	234	FL,DL,HL_1	MAX	1.00	0.619	0.822	0.758	PPV
51	80	FL,DL,HL_1	HARMONIC	0.32	0.575	0.749	0.698	PPV
52	80	FL,DL,HL_1	MAX	0.32	0.574	0.784	0.707	PPV
53	151	FL,DL,HL_1	MAX	0.64	0.572	0.817	0.700	PPV
54	153	FL,DL,HL_1	AVG	0.64	0.561	0.743	0.710	PPV
		FL,DL,HL_1 Result			***0.606***	***0.795***	***0.746***	***PPV***
55	155	FL,TL,BL	NWS	1.00	0.644	0.804	0.814	TPR
56	155	FL,TL,BL	WS	1.00	0.639	0.777	0.833	TPR
57	154	FL,TL,BL	AVG	1.00	0.630	0.821	0.756	PPV
58	53	FL,TL,BL	AVG	0.32	0.599	0.780	0.723	PPV
59	101	FL,TL,BL	HARMONIC	0.64	0.584	0.764	0.738	PPV
60	53	FL,TL,BL	NWS	0.32	0.574	0.765	0.688	PPV
61	101	FL,TL,BL	NWS	0.64	0.573	0.776	0.700	PPV
62	101	FL,TL,BL	WS	0.64	0.568	0.798	0.694	PPV
63	50	FL,TL,BL	MIN	0.32	0.550	0.779	0.664	PPV
		FL,TL,BL Result			***0.596***	***0.785***	***0.734***	***PPV***
64	233	FL,TL,HL_1	HARMONIC	1.00	0.657	0.818	0.794	PPV
65	231	FL,TL,HL_1	MIN	1.00	0.653	0.800	0.820	TPR
66	150	FL,TL,HL_1	WS	0.64	0.633	0.787	0.812	TPR
67	233	FL,TL,HL_1	NWS	1.00	0.631	0.819	0.769	PPV
68	75	FL,TL,HL_1	MIN	0.32	0.610	0.785	0.765	PPV
69	150	FL,TL,HL_1	AVG	0.64	0.599	0.784	0.748	PPV
70	74	FL,TL,HL_1	MAX	0.32	0.597	0.772	0.723	PPV
71	76	FL,TL,HL_1	HARMONIC	0.32	0.592	0.769	0.731	PPV
72	148	FL,TL,HL_1	MAX	0.64	0.560	0.739	0.708	PPV
		FL,TL,HL_1 Result			***0.614***	***0.786***	***0.763***	***PPV***
		Grand Average			***0.603***	***0.787***	***0.746***	***PPV***

Tab. G.11. Table lists the top 3 results for every triple loss combination. The column »*Strategy*« indicates which type of loss merging strategy has been used. The IoU column reflects the average outcomes contrasting the foreground class against the background class. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

Merge Strategy

No.	R-time(m)	Loss	Strategy	Selection %	IoU	PPV	TPR	PPV vs. TPR
1	109	CE,DL	AVG	1.00	0.662	0.820	0.810	PPV
2	151	TL,BL	AVG	1.00	0.653	0.825	0.776	PPV
3	109	CE,TL	AVG	1.00	0.642	0.821	0.763	PPV
4	102	BL,FL	AVG	0.64	0.633	0.792	0.792	TPR
5	101	DL,BL	AVG	0.64	0.588	0.793	0.698	PPV
6	54	TL,BL	AVG	0.32	0.577	0.752	0.735	PPV
7	153	HL_1,FL	AVG	0.64	0.577	0.799	0.709	PPV
8	39	CE,DL	AVG	0.32	0.560	0.736	0.738	TPR
9	81	HL_1,FL	AVG	0.32	0.559	0.735	0.712	PPV
			AVG Result		***0.606***	***0.786***	***0.748***	***PPV***
10	102	BL,FL	HARMONIC	0.64	0.644	0.842	0.762	PPV
11	72	CE,DL	HARMONIC	0.64	0.643	0.829	0.765	PPV
12	101	BL,CE	HARMONIC	0.64	0.640	0.814	0.791	PPV
13	153	TL,BL	HARMONIC	1.00	0.633	0.794	0.780	PPV
14	152	BL,CE	HARMONIC	1.00	0.618	0.803	0.738	PPV
15	232	DL,HL_1	HARMONIC	1.00	0.613	0.821	0.736	PPV
16	81	HL_1,CE	HARMONIC	0.32	0.602	0.764	0.758	PPV
17	41	FL,DL	HARMONIC	0.32	0.587	0.796	0.726	PPV
18	56	BL,FL	HARMONIC	0.32	0.580	0.795	0.696	PPV
			HARMONIC Result		***0.618***	***0.806***	***0.750***	***PPV***
19	111	FL,DL	MAX	1.00	0.639	0.836	0.768	PPV
20	152	HL_1,FL	MAX	0.64	0.636	0.775	0.839	TPR
21	108	CE,TL	MAX	1.00	0.627	0.835	0.749	PPV
22	40	FL,TL	MAX	0.32	0.597	0.735	0.755	TPR
23	80	TL,HL_1	MAX	0.32	0.596	0.788	0.745	PPV
24	108	CE,DL	MAX	1.00	0.592	0.761	0.723	PPV
25	82	HL_1,CE	MAX	0.32	0.586	0.768	0.760	PPV
26	73	FL,TL	MAX	0.64	0.545	0.777	0.656	PPV
27	150	TL,HL_1	MAX	0.64	0.518	0.777	0.625	PPV
			MAX Result		***0.593***	***0.784***	***0.736***	***PPV***
28	74	FL,DL	MIN	0.64	0.658	0.823	0.806	PPV
29	41	FL,DL	MIN	0.32	0.648	0.800	0.798	PPV
30	151	TL,HL_1	MIN	0.64	0.645	0.795	0.817	TPR
31	88	DL,BL	MIN	0.64	0.632	0.801	0.778	PPV
32	109	CE,DL	MIN	1.00	0.624	0.832	0.751	PPV
33	132	DL,BL	MIN	1.00	0.621	0.787	0.790	TPR
34	232	HL_1,FL	MIN	1.00	0.619	0.822	0.742	PPV
35	81	HL_1,FL	MIN	0.32	0.610	0.773	0.758	PPV

No.	R-time(m)	Loss	Strategy	Selection %	IoU	PPV	TPR	PPV vs. TPR
36	36	CE,TL	MIN	0.32	0.585	0.731	0.754	TPR
			MIN Result		***0.627***	***0.796***	***0.777***	***PPV***
37	152	DL,BL	NWS	1.00	0.671	0.817	0.830	TPR
38	109	CE,DL	NWS	1.00	0.657	0.818	0.808	PPV
39	112	FL,DL	NWS	1.00	0.640	0.816	0.768	PPV
40	71	CE,TL	NWS	0.64	0.639	0.821	0.774	PPV
41	102	BL,FL	NWS	0.64	0.631	0.811	0.758	PPV
42	152	HL_1,CE	NWS	0.64	0.631	0.800	0.785	PPV
43	54	TL,BL	NWS	0.32	0.585	0.749	0.739	PPV
44	37	CE,TL	NWS	0.32	0.584	0.762	0.717	PPV
45	56	BL,FL	NWS	0.32	0.584	0.736	0.719	PPV
			NWS Result		***0.625***	***0.792***	***0.766***	***PPV***
46	152	TL,BL	WS	1.00	0.669	0.832	0.805	PPV
47	74	FL,TL	WS	0.64	0.643	0.825	0.764	PPV
48	232	HL_1,CE	WS	1.00	0.642	0.797	0.819	TPR
49	152	DL,BL	WS	1.00	0.642	0.794	0.793	PPV
50	72	CE,DL	WS	0.64	0.629	0.789	0.786	PPV
51	100	TL,BL	WS	0.64	0.613	0.778	0.758	PPV
52	55	DL,BL	WS	0.32	0.608	0.790	0.775	PPV
53	41	FL,DL	WS	0.32	0.596	0.786	0.768	PPV
54	56	BL,CE	WS	0.32	0.583	0.789	0.704	PPV
			WS Result		***0.625***	***0.798***	***0.775***	***PPV***
			Grand Average		***0.616***	***0.794***	***0.759***	***PPV***

Tab. G.12. Table contains 54 trained models representing the top 3 double loss combinations for every selection percentage and discrete configuration setting. The IoU column reflects the average outcomes contrasting the foreground class against the background class. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

No.	R-time(m)	Loss	Strategy	Selection %	IoU	PPV	TPR	PPV vs. TPR
1	99	CE,TL,BL	AVG	0.64	0.648	0.816	0.797	PPV
2	153	CE,TL,BL	AVG	1.00	0.630	0.816	0.754	PPV
3	154	FL,TL,BL	AVG	1.00	0.630	0.821	0.756	PPV
4	154	CE,DL,BL	AVG	1.00	0.611	0.821	0.717	PPV
5	53	FL,TL,BL	AVG	0.32	0.599	0.780	0.723	PPV
6	150	FL,TL,HL_1	AVG	0.64	0.599	0.784	0.748	PPV
7	75	CE,TL,HL_1	AVG	0.32	0.590	0.802	0.722	PPV
8	77	CE,DL,HL_1	AVG	0.32	0.586	0.793	0.738	PPV
9	150	CE,DL,HL_1	AVG	0.64	0.576	0.795	0.692	PPV
			AVG Result		***0.608***	***0.803***	***0.739***	***PPV***

No.	R-time(m)	Loss	Strategy	Selection %	IoU	PPV	TPR	PPV vs. TPR
10	231	CE,TL,HL_1	HARMONIC	1.00	0.672	0.808	0.837	TPR
11	159	FL,DL,BL	HARMONIC	1.00	0.670	0.799	0.851	TPR
12	233	FL,TL,HL_1	HARMONIC	1.00	0.657	0.818	0.794	PPV
13	104	FL,DL,BL	HARMONIC	0.64	0.648	0.815	0.808	PPV
14	101	CE,DL,BL	HARMONIC	0.64	0.613	0.787	0.738	PPV
15	151	CE,DL,HL_1	HARMONIC	0.64	0.610	0.786	0.779	PPV
16	77	CE,DL,HL_1	HARMONIC	0.32	0.608	0.765	0.756	PPV
17	76	FL,TL,HL_1	HARMONIC	0.32	0.592	0.769	0.731	PPV
18	56	FL,DL,BL	HARMONIC	0.32	0.592	0.797	0.719	PPV
			HARMONIC Result		***0.629***	***0.794***	***0.779***	***PPV***
19	77	CE,DL,HL_1	MAX	0.32	0.633	0.816	0.757	PPV
20	230	CE,TL,HL_1	MAX	1.00	0.620	0.818	0.765	PPV
21	234	FL,DL,HL_1	MAX	1.00	0.619	0.822	0.758	PPV
22	74	CE,TL,HL_1	MAX	0.32	0.617	0.797	0.772	PPV
23	74	FL,TL,HL_1	MAX	0.32	0.597	0.772	0.723	PPV
24	149	CE,DL,HL_1	MAX	0.64	0.575	0.825	0.689	PPV
25	151	FL,DL,HL_1	MAX	0.64	0.572	0.817	0.700	PPV
26	148	FL,TL,HL_1	MAX	0.64	0.560	0.739	0.708	PPV
27	229	CE,TL,HL_1	MAX	1.00	0.545	0.750	0.659	PPV
			MAX Result		***0.593***	***0.795***	***0.726***	***PPV***
28	231	FL,TL,HL_1	MIN	1.00	0.653	0.800	0.820	TPR
29	233	FL,DL,HL_1	MIN	1.00	0.652	0.806	0.818	TPR
30	89	CE,DL,BL	MIN	0.64	0.651	0.786	0.834	TPR
31	149	CE,DL,HL_1	MIN	0.64	0.634	0.804	0.792	PPV
32	132	CE,TL,BL	MIN	1.00	0.632	0.812	0.750	PPV
33	151	FL,DL,HL_1	MIN	0.64	0.631	0.815	0.769	PPV
34	75	FL,TL,HL_1	MIN	0.32	0.610	0.785	0.765	PPV
35	49	CE,DL,BL	MIN	0.32	0.588	0.743	0.812	TPR
36	53	FL,DL,BL	MIN	0.32	0.586	0.742	0.732	PPV
			MIN Result		***0.626***	***0.788***	***0.788***	***PPV***
37	149	CE,TL,HL_1	NWS	0.64	0.646	0.803	0.824	TPR
38	155	FL,TL,BL	NWS	1.00	0.644	0.804	0.814	TPR
39	233	FL,TL,HL_1	NWS	1.00	0.631	0.819	0.769	PPV
40	101	CE,TL,BL	NWS	0.64	0.629	0.794	0.788	PPV
41	235	FL,DL,HL_1	NWS	1.00	0.626	0.799	0.769	PPV
42	52	CE,TL,BL	NWS	0.32	0.579	0.767	0.727	PPV
43	53	FL,TL,BL	NWS	0.32	0.574	0.765	0.688	PPV
44	101	FL,TL,BL	NWS	0.64	0.573	0.776	0.700	PPV
45	76	CE,TL,HL_1	NWS	0.32	0.564	0.747	0.692	PPV
			NWS Result		***0.607***	***0.786***	***0.752***	***PPV***
46	80	FL,DL,HL_1	WS	0.32	0.640	0.817	0.784	PPV
47	155	FL,TL,BL	WS	1.00	0.639	0.777	0.833	TPR
48	232	CE,TL,HL_1	WS	1.00	0.636	0.828	0.762	PPV
49	150	FL,TL,HL_1	WS	0.64	0.633	0.787	0.812	TPR
50	153	CE,TL,BL	WS	1.00	0.628	0.800	0.770	PPV

No.	R-time(m)	Loss	Strategy	Selection %	IoU	PPV	TPR	PPV vs. TPR
51	101	CE,DL,BL	WS	0.64	0.613	0.770	0.788	TPR
52	103	FL,DL,BL	WS	0.64	0.609	0.809	0.761	PPV
53	56	FL,DL,BL	WS	0.32	0.582	0.760	0.725	PPV
54	54	CE,DL,BL	WS	0.32	0.569	0.779	0.695	PPV
			WS Result		***0.617***	***0.792***	***0.770***	***PPV***
			Grand Average		***0.613***	***0.793***	***0.759***	***PPV***

Tab. G.13. Table contains 54 trained models representing the top 3 triple loss combinations for every selection percentage and discrete configuration setting. The IoU column reflects the average outcomes contrasting the foreground class against the background class. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

G.2.3. Continuous

No.	R-time(m)	Loss	Selection %	IoU	PPV	TPR	PPV vs. TPR	pbm_alpha	pbm_l
1	75	BL,CE	0.64	0.612	0.816	0.742	PPV	3.122	0.961
2	107	BL,CE	1.00	0.118	0.118	0.500	TPR	8.482	0.644
3	99	BL,CE	0.64	0.130	0.606	0.508	PPV	5.927	0.761
		BL,CE Result		***0.287***	***0.513***	***0.583***	***TPR***	***5.844***	***0.789***
4	98	BL,FL	0.64	0.381	0.381	0.500	TPR	8.238	0.970
5	38	BL,FL	0.32	0.121	0.121	0.500	TPR	9.974	0.621
6	73	BL,FL	0.64	0.116	0.116	0.500	TPR	8.421	0.573
		BL,FL Result		***0.206***	***0.206***	***0.500***	***TPR***	***8.878***	***0.721***
7	36	CE,DL	0.32	0.592	0.784	0.700	PPV	7.632	0.572
8	58	CE,DL	0.64	0.541	0.753	0.665	PPV	9.186	0.672
9	288	CE,DL	1.00	0.509	0.730	0.619	PPV	2.341	0.497
		CE,DL Result		***0.547***	***0.756***	***0.661***	***PPV***	***6.386***	***0.580***
10	41	CE,TL	0.32	0.557	0.783	0.678	PPV	5.613	0.169
11	74	CE,TL	0.64	0.595	0.823	0.702	PPV	9.064	0.194
12	32	CE,TL	0.32	0.627	0.781	0.788	TPR	1.616	0.332
		CE,TL Result		***0.593***	***0.796***	***0.723***	***PPV***	***5.431***	***0.232***
13	40	DL,BL	0.32	0.637	0.784	0.826	TPR	3.571	0.990
14	70	DL,BL	0.64	0.513	0.688	0.708	TPR	6.435	0.924
15	111	DL,BL	1.00	0.083	0.115	0.288	TPR	9.667	0.532
		DL,BL Result		***0.411***	***0.529***	***0.607***	***TPR***	***6.558***	***0.815***
16	53	DL,HL	0.32	0.644	0.766	0.837	TPR	9.078	0.942
17	234	DL,HL	1.00	0.633	0.814	0.759	PPV	1.492	0.819
		DL,HL Result		***0.639***	***0.790***	***0.798***	***TPR***	***5.285***	***0.881***
18	113	FL,DL	1.00	0.604	0.793	0.745	PPV	8.792	0.966
19	36	FL,DL	0.32	0.599	0.806	0.715	PPV	8.231	0.008
20	36	FL,DL	0.32	0.540	0.742	0.675	PPV	9.333	0.396
		FL,DL Result		***0.581***	***0.780***	***0.712***	***PPV***	***8.786***	***0.456***
21	71	FL,TL	0.64	0.642	0.810	0.768	PPV	9.128	0.062
22	58	FL,TL	0.64	0.645	0.795	0.806	TPR	1.605	0.541
23	93	FL,TL	1.00	0.632	0.790	0.766	PPV	3.285	0.087
		FL,TL Result		***0.640***	***0.798***	***0.780***	***PPV***	***4.673***	***0.230***
24	75	HL,CE	0.32	0.445	0.646	0.573	PPV	9.000	0.469
25	54	HL,CE	0.32	0.629	0.781	0.779	PPV	7.954	0.938
26	53	HL,CE	0.32	0.585	0.720	0.714	PPV	6.801	0.870
		HL,CE Result		***0.553***	***0.716***	***0.689***	***PPV***	***7.918***	***0.759***
27	222	HL,FL	1.00	0.608	0.778	0.787	TPR	8.950	0.654
28	96	HL,FL	0.64	0.593	0.766	0.702	PPV	6.902	0.939
29	53	HL,FL	0.32	0.591	0.814	0.725	PPV	4.918	0.922
		HL,FL Result		***0.597***	***0.786***	***0.738***	***PPV***	***6.923***	***0.838***
30	107	TL,BL	1.00	0.584	0.752	0.749	PPV	6.633	0.955
31	43	TL,BL	0.32	0.379	0.379	0.500	TPR	2.175	0.906
		TL,BL Result		***0.481***	***0.565***	***0.624***	***TPR***	***4.404***	***0.931***
32	50	TL,HL	0.32	0.579	0.752	0.726	PPV	3.723	0.897
33	235	TL,HL	1.00	0.606	0.734	0.827	TPR	2.684	0.612

No.	R-time(m)	Loss	Selection %	IoU	PPV	TPR	PPV vs. TPR	pbm_alpha	pbm_l
34	232	TL,HL	1.00	0.539	0.759	0.686	PPV	2.474	0.448
		TL,HL Result		***0.575***	***0.748***	***0.746***	***PPV***	***2.960***	***0.653***
		Grand Average		***0.506***	***0.665***	***0.678***	***TPR***	***6.248***	***0.642***

Tab. G.14. Table lists the top 3 results for every double loss combination using the performance-based continuous merge strategy. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

No.	R-time(m)	Loss	Selection %	IoU	PPV	TPR	PPV vs. TPR	pbm_alpha	pbm_l
1	113	CE,DL,BL	1.00	0.588	0.739	0.742	TPR	9.854	0.202
2	43	CE,DL,BL	0.32	0.411	0.495	0.532	TPR	4.815	0.909
3	67	CE,DL,BL	0.64	0.310	0.639	0.652	TPR	5.479	0.360
		CE,DL,BL Result		***0.436***	***0.625***	***0.642***	***TPR***	***6.716***	***0.490***
4	149	CE,DL,HL	1.00	0.667	0.809	0.830	TPR	5.745	0.860
5	50	CE,DL,HL	0.32	0.608	0.755	0.795	TPR	5.766	0.592
6	150	CE,DL,HL	1.00	0.594	0.801	0.712	PPV	9.568	0.427
		CE,DL,HL Result		***0.623***	***0.788***	***0.779***	***PPV***	***7.026***	***0.626***
7	41	CE,TL,BL	0.32	0.519	0.719	0.733	TPR	6.290	0.862
8	74	CE,TL,BL	0.64	0.507	0.662	0.683	TPR	8.155	0.990
9	37	CE,TL,BL	0.32	0.406	0.633	0.693	TPR	9.814	0.527
		CE,TL,BL Result		***0.477***	***0.671***	***0.703***	***TPR***	***8.086***	***0.793***
10	145	CE,TL,HL	1.00	0.633	0.818	0.760	PPV	2.834	0.952
11	151	CE,TL,HL	1.00	0.627	0.771	0.807	TPR	9.665	0.875
12	151	CE,TL,HL	1.00	0.613	0.784	0.773	PPV	9.529	0.990
		CE,TL,HL Result		***0.624***	***0.791***	***0.780***	***PPV***	***7.342***	***0.939***
13	73	FL,DL,BL	0.64	0.632	0.831	0.758	PPV	3.168	0.948
14	71	FL,DL,BL	0.64	0.588	0.755	0.776	TPR	7.923	0.767
15	76	FL,DL,BL	0.64	0.581	0.765	0.768	TPR	6.878	0.612
		FL,DL,BL Result		***0.600***	***0.784***	***0.767***	***PPV***	***5.989***	***0.776***
16	51	FL,DL,HL	0.32	0.620	0.786	0.769	PPV	6.774	0.934
17	94	FL,DL,HL	0.64	0.601	0.812	0.745	PPV	4.047	0.093
18	147	FL,DL,HL	1.00	0.596	0.739	0.793	TPR	9.887	0.190
		FL,DL,HL Result		***0.606***	***0.779***	***0.769***	***PPV***	***6.903***	***0.406***
19	43	FL,TL,BL	0.32	0.570	0.739	0.728	PPV	8.957	0.666
20	38	FL,TL,BL	0.32	0.530	0.757	0.694	PPV	7.897	0.731
21	69	FL,TL,BL	0.64	0.462	0.686	0.700	TPR	6.306	0.323
		FL,TL,BL Result		***0.521***	***0.727***	***0.707***	***PPV***	***7.720***	***0.573***
22	94	FL,TL,HL	0.64	0.609	0.814	0.733	PPV	9.214	0.249
23	99	FL,TL,HL	0.64	0.567	0.749	0.693	PPV	6.783	0.668
24	96	FL,TL,HL	0.64	0.520	0.638	0.630	PPV	9.134	0.077
		FL,TL,HL Result		***0.566***	***0.734***	***0.685***	***PPV***	***8.377***	***0.332***
		Grand Average		***0.557***	***0.737***	***0.729***	***PPV***	***7.270***	***0.617***

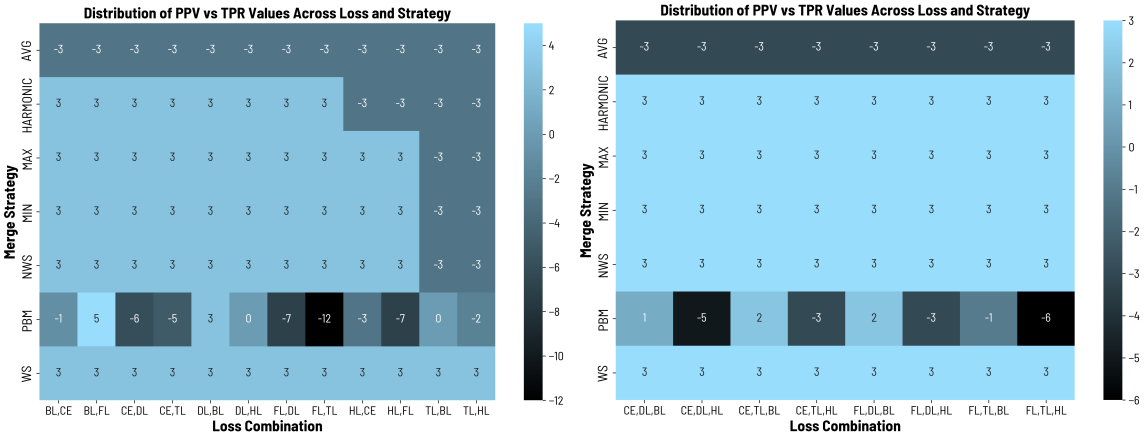
Tab. G.15. Table lists the top 3 results for every triple loss combination using the performance-based continuous merge strategy. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

G.2.4. Specific

No.	R-time(m)	Loss	Selection %	Strategy	IoU	IoU 0	IoU 1	PPV	TPR	PPV vs. TPR
1	299	CE,TL	1.00	NWS	0.660	0.790	0.529	0.799	0.840	TPR
2	167	CE,TL	1.00	NWS	0.659	0.790	0.529	0.787	0.855	TPR
3	302	CE,TL	1.00	NWS	0.652	0.798	0.507	0.810	0.814	TPR
4	283	CE,TL	1.00	NWS	0.651	0.805	0.497	0.800	0.807	PPV
5	285	CE,TL	1.00	NWS	0.629	0.821	0.437	0.769	0.766	PPV
6	294	CE,TL	1.00	NWS	0.629	0.768	0.490	0.768	0.831	TPR
7	185	CE,TL	0.64	NWS	0.671	0.812	0.529	0.822	0.814	PPV
8	166	CE,TL	0.64	NWS	0.670	0.802	0.539	0.796	0.850	TPR
9	174	CE,TL	0.64	NWS	0.665	0.817	0.513	0.827	0.809	PPV
10	186	CE,TL	0.64	NWS	0.661	0.821	0.501	0.816	0.798	PPV
11	197	CE,TL	0.64	NWS	0.655	0.811	0.499	0.793	0.824	TPR
12	202	CE,TL	0.64	NWS	0.648	0.817	0.479	0.819	0.771	PPV
13	106	CE,TL	0.32	NWS	0.657	0.830	0.483	0.842	0.774	PPV
14	109	CE,TL	0.32	NWS	0.648	0.804	0.492	0.800	0.817	TPR
15	102	CE,TL	0.32	NWS	0.629	0.805	0.453	0.780	0.804	TPR
16	56	CE,TL	0.32	NWS	0.619	0.778	0.459	0.792	0.787	PPV
17	103	CE,TL	0.32	NWS	0.609	0.740	0.479	0.789	0.776	PPV
18	115	CE,TL	0.32	NWS	0.591	0.731	0.451	0.777	0.784	TPR
				Grand Average	***0.645***	***0.797***	***0.493***	***0.799***	***0.807***	***TPR***

Tab. G.16. Table lists six models for each selection percentage trained repetitively on the loss combination CE, TL using the normalized weighted sum (NWS) strategy for merging. These results serve as part of a final comparison against baseline results providing an identical iteration number. The columns $IoU_0, \ldots, IoU_1$ represent the two classes, Background and Foreground. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

G.2.5. PPV vs. TPR



(a) Distribution of PPV vs. TPR for double losses

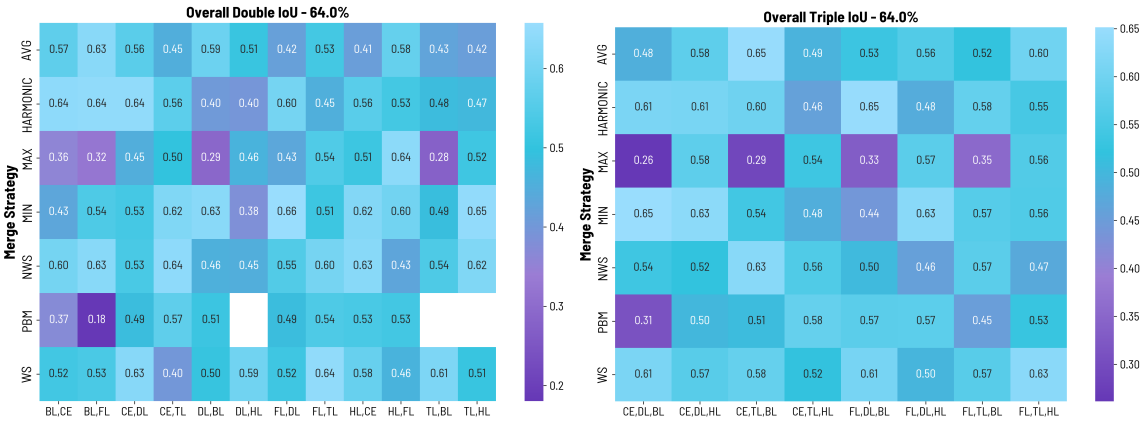
(b) Distribution of PPV vs. TPR for triple losses

Fig. G.3. Heatmaps visualizing the relative prevalence as a difference map of PPV and TPR across various loss combinations and merge strategies. Each cell represents a unique combination. Positive values indicate a predominance of TPR, while negative values signify a predominance of PPV.

G.2.6. Ablation study

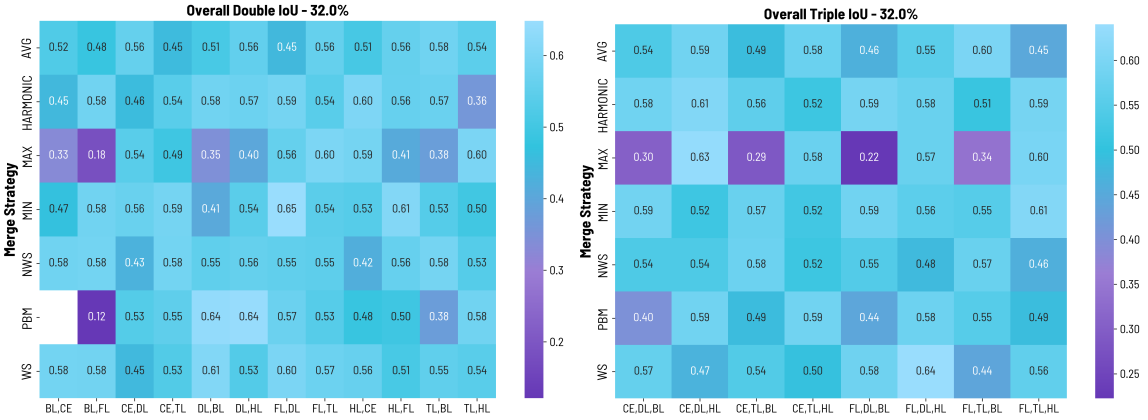
(a) Double Loss Combination for 100 % Data Utilization

(b) Triple Loss Combination for 100 % Data Utilization



(c) Double Loss Combination for 64 % Data Utilization

(d) Triple Loss Combination for 64 % Data Utilization



(e) Double Loss Combination for 32 % Data Utilization

(f) Triple Loss Combination for 32 % Data Utilization

Fig. G.4. Overall results generated from different dataset sizes. The left column (a,c,e) illustrates results from double loss combinations, while the right column (b,d,f) depicts results from triple loss combinations.

G.3. Idrid

G.3.1. Baseline

No.	R-time(m)	Loss type	Loss	Iou	IoU 0	IoU 1	IoU 2	IoU 3	IoU 4	PPV	TPR	PPV vs. TPR
1	37	DB	CE	0.367	0.961	0.002	0.234	0.002	0.634	0.520	0.435	PPV
2	44	DB	CE	0.364	0.954	0.002	0.290	0.011	0.561	0.565	0.429	PPV
3	40	DB	CE	0.348	0.958	0.042	0.289	0.000	0.449	0.525	0.408	PPV
			CE Result	***0.359***	***0.958***	***0.015***	***0.271***	***0.004***	***0.548***	***0.537***	***0.424***	**PPV**
4	35	DB	FL	0.361	0.949	0.003	0.289	0.027	0.539	0.602	0.431	PPV
5	42	DB	FL	0.301	0.940	0.002	0.258	0.023	0.282	0.522	0.374	PPV
6	35	DB	FL	0.300	0.948	0.003	0.255	0.041	0.253	0.580	0.381	PPV
			FL Result	***0.321***	***0.946***	***0.002***	***0.267***	***0.031***	***0.358***	***0.568***	***0.396***	**PPV**
7	36	RB	FL	0.343	0.896	0.103	0.276	0.115	0.324	0.591	0.446	PPV
8	44	RB	FL	0.282	0.941	0.000	0.000	0.000	0.470	0.342	0.336	PPV
9	45	RB	FL	0.276	0.940	0.017	0.240	0.053	0.131	0.501	0.358	PPV
			FL Result	***0.300***	***0.926***	***0.040***	***0.172***	***0.056***	***0.308***	***0.478***	***0.380***	**PPV**
10	27	RB	TL	0.302	0.969	0.000	0.000	0.000	0.539	0.352	0.332	PPV
11	22	RB	TL	0.228	0.952	0.000	0.000	0.000	0.188	0.262	0.269	TPR
12	26	RB	TL	0.225	0.949	0.000	0.000	0.000	0.177	0.248	0.273	TPR
			TL Result	***0.252***	***0.956***	***0.000***	***0.000***	***0.000***	***0.301***	***0.287***	***0.292***	**TPR**
13	91	BB	HL_1	0.283	0.940	0.000	0.000	0.000	0.477	0.343	0.340	PPV
14	89	BB	HL_1	0.270	0.948	0.000	0.000	0.000	0.402	0.342	0.319	PPV
15	95	BB	HL_1	0.255	0.951	0.000	0.000	0.000	0.326	0.302	0.301	PPV
			HL Result	***0.270***	***0.946***	***0.000***	***0.000***	***0.000***	***0.402***	***0.329***	***0.320***	**PPV**
16	64	BB	BL	0.075	0.349	0.007	0.007	0.001	0.011	0.199	0.168	PPV
17	65	BB	BL	0.038	0.171	0.012	0.006	0.001	0.000	0.187	0.177	PPV
18	69	BB	BL	0.024	0.089	0.007	0.013	0.001	0.009	0.202	0.216	TPR
			BL Result	***0.046***	***0.203***	***0.009***	***0.009***	***0.001***	***0.007***	***0.196***	***0.187***	**PPV**
			Grand Average	***0.258***	***0.823***	***0.011***	***0.120***	***0.015***	***0.321***	***0.399***	***0.333***	**PPV**

Tab. G.17. The table presents the average values of various metrics for the top 3 baseline models corresponding to each loss function. The columns $IoU_0, \ldots, IoU_4$ represent the five classes, Background, Haemorrhages, Hard Exudates, Microaneurys and Optic Disc. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

G.3.2. Discrete

Loss Combination

No.	R-time(m)	Loss combination	Strategy	IoU	IoU 0	IoU 1	IoU 2	IoU 3	Iou 4	PPV	TPR	PPV vs. TPR
1	189	BL,CE	AVG	0.352	0.936	0.002	0.309	0.000	0.514	0.492	0.426	PPV
2	83	BL,CE	NWS	0.330	0.935	0.032	0.279	0.000	0.402	0.539	0.401	PPV
3	38	BL,CE	MAX	0.298	0.933	0.002	0.265	0.000	0.291	0.441	0.380	PPV
		BL,CE Result		***0.327***	***0.935***	***0.012***	***0.284***	***0.000***	***0.402***	***0.491***	***0.402***	**PPV**
4	129	BL,FL	AVG	0.358	0.951	0.169	0.342	0.076	0.252	0.665	0.426	PPV
5	116	BL,FL	WS	0.329	0.950	0.001	0.274	0.006	0.413	0.475	0.408	PPV
6	38	BL,FL	NWS	0.328	0.920	0.012	0.280	0.016	0.410	0.588	0.420	PPV
		BL,FL Result		***0.338***	***0.941***	***0.061***	***0.299***	***0.033***	***0.358***	***0.576***	***0.418***	**PPV**
7	16	CE,DL	AVG	0.397	0.936	0.205	0.312	0.163	0.369	0.636	0.509	PPV
8	33	CE,DL	AVG	0.390	0.948	0.121	0.315	0.134	0.433	0.697	0.468	PPV
9	16	CE,DL	HARMONIC	0.347	0.936	0.051	0.304	0.054	0.390	0.645	0.427	PPV
		CE,DL Result		***0.378***	***0.940***	***0.126***	***0.310***	***0.117***	***0.397***	***0.659***	***0.468***	**PPV**
10	24	CE,TL	MAX	0.414	0.916	0.216	0.368	0.196	0.372	0.637	0.534	PPV
11	29	CE,TL	WS	0.394	0.948	0.226	0.286	0.196	0.315	0.626	0.534	PPV
12	29	CE,TL	AVG	0.380	0.940	0.112	0.292	0.151	0.403	0.597	0.495	PPV
		CE,TL Result		***0.396***	***0.934***	***0.185***	***0.315***	***0.181***	***0.363***	***0.620***	***0.521***	**PPV**
13	60	DL,BL	AVG	0.380	0.946	0.132	0.284	0.146	0.391	0.580	0.474	PPV
14	37	DL,BL	AVG	0.378	0.940	0.086	0.320	0.145	0.401	0.633	0.479	PPV
15	69	DL,BL	WS	0.370	0.964	0.096	0.284	0.155	0.352	0.676	0.453	PPV
		DL,BL Result		***0.376***	***0.950***	***0.105***	***0.296***	***0.148***	***0.381***	***0.630***	***0.469***	**PPV**
16	110	DL,HL_1	NWS	0.305	0.930	0.060	0.262	0.090	0.181	0.537	0.410	PPV
17	103	DL,HL_1	MAX	0.258	0.910	0.002	0.231	0.000	0.145	0.354	0.354	TPR
18	101	DL,HL_1	HARMONIC	0.257	0.961	0.000	0.000	0.000	0.322	0.277	0.297	TPR
		DL,HL_1 Result		***0.273***	***0.933***	***0.021***	***0.164***	***0.030***	***0.216***	***0.389***	***0.354***	**PPV**
19	17	FL,DL	AVG	0.362	0.936	0.122	0.245	0.117	0.390	0.663	0.472	PPV
20	37	FL,DL	NWS	0.351	0.962	0.080	0.257	0.158	0.298	0.649	0.446	PPV
21	17	FL,DL	MAX	0.349	0.940	0.165	0.247	0.165	0.228	0.637	0.458	PPV
		FL,DL Result		***0.354***	***0.946***	***0.123***	***0.249***	***0.147***	***0.305***	***0.649***	***0.459***	**PPV**
22	16	FL,TL	MAX	0.377	0.919	0.216	0.270	0.160	0.318	0.626	0.519	PPV
23	16	FL,TL	AVG	0.372	0.968	0.188	0.275	0.168	0.260	0.641	0.488	PPV
24	16	FL,TL	MIN	0.351	0.948	0.053	0.308	0.069	0.377	0.640	0.426	PPV
		FL,TL Result		***0.367***	***0.945***	***0.152***	***0.284***	***0.133***	***0.318***	***0.636***	***0.478***	**PPV**
25	100	HL_1,CE	MIN	0.332	0.931	0.004	0.250	0.000	0.473	0.485	0.407	PPV
26	109	HL_1,CE	NWS	0.307	0.955	0.001	0.194	0.000	0.387	0.410	0.372	PPV
27	85	HL_1,CE	HARMONIC	0.286	0.948	0.003	0.171	0.000	0.307	0.394	0.366	PPV
		HL_1,CE Result		***0.308***	***0.945***	***0.003***	***0.205***	***0.000***	***0.389***	***0.430***	***0.382***	**PPV**
28	94	HL_1,FL	MIN	0.342	0.923	0.093	0.316	0.063	0.317	0.617	0.430	PPV
29	89	HL_1,FL	HARMONIC	0.327	0.939	0.000	0.228	0.000	0.469	0.433	0.433	TPR
30	98	HL_1,FL	NWS	0.319	0.940	0.003	0.212	0.000	0.441	0.449	0.390	PPV
		HL_1,FL Result		***0.330***	***0.934***	***0.032***	***0.252***	***0.021***	***0.409***	***0.500***	***0.418***	**PPV**
31	82	TL,BL	AVG	0.369	0.927	0.199	0.339	0.134	0.246	0.601	0.468	PPV
32	83	TL,BL	HARMONIC	0.300	0.936	0.067	0.178	0.000	0.316	0.477	0.433	PPV
33	36	TL,BL	AVG	0.277	0.961	0.000	0.216	0.000	0.207	0.418	0.373	PPV
		TL,BL Result		***0.315***	***0.941***	***0.089***	***0.244***	***0.045***	***0.256***	***0.499***	***0.425***	**PPV**
34	94	TL,HL_1	NWS	0.275	0.960	0.000	0.000	0.000	0.417	0.328	0.316	PPV
35	104	TL,HL_1	WS	0.240	0.951	0.000	0.000	0.000	0.250	0.319	0.279	PPV

No.	R-time(m)	Loss combination	Strategy	IoU	IoU 0	IoU 1	IoU 2	IoU 3	Iou 4	PPV	TPR	PPV vs. TPR
36	94	TL,HL_1	MAX	0.238	0.946	0.000	0.000	0.000	0.245	0.316	0.285	PPV
		TL,HL Result		***0.251***	***0.952***	***0.000***	***0.000***	***0.000***	***0.304***	***0.321***	***0.294***	***PPV***
		Grand Average		***0.334***	***0.941***	***0.076***	***0.242***	***0.071***	***0.342***	***0.533***	***0.424***	**PPV**

Tab. G.18. Table lists the top 3 results for every double loss combination. The column »*Strategy*« indicates which type of loss merging strategy has been used. The columns $IoU_0, \ldots, IoU_4$ represent the five classes, Background, Haemorrhages, Hard Exudates, Microaneurys and Optic Disc. The column titled »*PPV vs. TPR*«" illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

No.	R-time(m)	Loss combination	Strategy	IoU	IoU 0	IoU 1	IoU 2	IoU 3	Iou 4	PPV	TPR	PPV vs. TPR
1	39	CE,DL,BL	AVG	0.387	0.935	0.069	0.306	0.136	0.488	0.612	0.498	PPV
2	47	CE,DL,BL	AVG	0.366	0.956	0.131	0.265	0.159	0.318	0.594	0.482	PPV
3	39	CE,DL,BL	NWS	0.336	0.936	0.154	0.236	0.150	0.205	0.641	0.462	PPV
		CE,DL,BL Result		***0.363***	***0.942***	***0.118***	***0.269***	***0.148***	***0.337***	***0.616***	***0.481***	***PPV***
4	115	CE,DL,HL_1	NWS	0.326	0.965	0.096	0.241	0.111	0.217	0.605	0.412	PPV
5	105	CE,DL,HL_1	AVG	0.312	0.955	0.001	0.234	0.000	0.371	0.453	0.389	PPV
6	111	CE,DL,HL_1	WS	0.304	0.942	0.019	0.257	0.110	0.193	0.522	0.398	PPV
		CE,DL,HL_1 Result		***0.314***	***0.954***	***0.039***	***0.244***	***0.074***	***0.260***	***0.527***	***0.400***	***PPV***
7	38	CE,TL,BL	WS	0.367	0.885	0.111	0.304	0.126	0.406	0.604	0.482	PPV
8	94	CE,TL,BL	NWS	0.359	0.955	0.135	0.278	0.176	0.250	0.622	0.476	PPV
9	86	CE,TL,BL	AVG	0.354	0.938	0.151	0.264	0.164	0.252	0.603	0.481	PPV
		CE,TL,BL Result		***0.360***	***0.926***	***0.132***	***0.282***	***0.155***	***0.303***	***0.610***	***0.480***	***PPV***
10	111	CE,TL,HL_1	NWS	0.311	0.927	0.040	0.244	0.145	0.198	0.522	0.438	PPV
11	107	CE,TL,HL_1	WS	0.300	0.907	0.036	0.224	0.106	0.226	0.519	0.419	PPV
12	99	CE,TL,HL_1	HARMONIC	0.292	0.949	0.000	0.072	0.000	0.440	0.420	0.350	PPV
		CE,TL,HL_1 Result		***0.301***	***0.928***	***0.025***	***0.180***	***0.084***	***0.288***	***0.487***	***0.402***	***PPV***
13	66	FL,DL,BL	AVG	0.411	0.948	0.223	0.325	0.185	0.375	0.667	0.520	PPV
14	75	FL,DL,BL	WS	0.368	0.936	0.107	0.291	0.135	0.370	0.617	0.475	PPV
15	67	FL,DL,BL	NWS	0.363	0.935	0.054	0.281	0.121	0.421	0.587	0.459	PPV
		FL,DL,BL Result		***0.381***	***0.940***	***0.128***	***0.299***	***0.147***	***0.388***	***0.624***	***0.485***	***PPV***
16	61	FL,DL,HL_1	MIN	0.332	0.946	0.002	0.281	0.008	0.424	0.567	0.404	PPV
17	120	FL,DL,HL_1	WS	0.320	0.930	0.100	0.284	0.138	0.150	0.513	0.429	PPV
18	167	FL,DL,HL_1	HARMONIC	0.318	0.940	0.009	0.249	0.008	0.386	0.545	0.404	PPV
		FL,DL,HL_1 Result		***0.324***	***0.938***	***0.037***	***0.272***	***0.051***	***0.320***	***0.542***	***0.412***	***PPV***
19	52	FL,TL,BL	AVG	0.351	0.935	0.233	0.257	0.188	0.140	0.586	0.489	PPV
20	40	FL,TL,BL	NWS	0.348	0.947	0.153	0.211	0.135	0.293	0.569	0.481	PPV
21	62	FL,TL,BL	NWS	0.335	0.946	0.100	0.277	0.169	0.183	0.611	0.454	PPV
		FL,TL,BL Result		***0.344***	***0.943***	***0.162***	***0.248***	***0.164***	***0.205***	***0.589***	***0.475***	***PPV***
22	61	FL,TL,HL_1	MIN	0.334	0.944	0.002	0.276	0.030	0.417	0.554	0.422	PPV
23	115	FL,TL,HL_1	NWS	0.327	0.941	0.091	0.271	0.133	0.196	0.566	0.436	PPV
24	62	FL,TL,HL_1	HARMONIC	0.314	0.941	0.015	0.285	0.008	0.319	0.521	0.377	PPV
		FL,TL,HL_1 Result		**0.325**	**0.942**	**0.036**	**0.277**	**0.057**	**0.311**	**0.547**	**0.412**	**PPV**
		Grand Average		***0.339***	***0.939***	***0.085***	***0.259***	***0.110***	***0.302***	***0.568***	***0.443***	***PPV***

Tab. G.19. Table lists the top 3 results for every triple loss combination. The column "Strategy" indicates which type of loss merging strategy has been used. The columns $IoU_0, \ldots, IoU_4$ represent the five classes, Background, Haemorrhages, Hard Exudates, Microaneurys and Optic Disc. The column titled »*PPV vs. TPR*«" illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

Merge Strategy

No.	R-time(m)	Loss	Strategy	IoU	IoU 0	IoU 1	IoU 2	IoU 3	IoU 4	PPV	TPR	PPV vs. TPR
1	16	CE,DL	AVG	0.397	0.936	0.205	0.312	0.163	0.369	0.636	0.509	PPV
2	33	CE,DL	AVG	0.390	0.948	0.121	0.315	0.134	0.433	0.697	0.468	PPV
3	60	DL,BL	AVG	0.380	0.946	0.132	0.284	0.146	0.391	0.580	0.474	PPV
			AVG Result	**0.389**	**0.943**	**0.153**	**0.303**	**0.148**	**0.398**	**0.637**	**0.483**	**PPV**
4	16	CE,DL	HARMONIC	0.347	0.936	0.051	0.304	0.054	0.390	0.645	0.427	PPV

No.	R-time(m)	Loss	Strategy	IoU	IoU 0	IoU 1	IoU 2	IoU 3	IoU 4	PPV	TPR	PPV vs. TPR
5	89	HL_1,FL	HARMONIC	0.327	0.939	0.000	0.228	0.000	0.469	0.433	0.433	TPR
6	18	FL,DL	HARMONIC	0.313	0.917	0.016	0.252	0.051	0.331	0.532	0.408	PPV
			HARMONIC Result	***0.329***	***0.931***	***0.022***	***0.261***	***0.035***	***0.397***	***0.537***	***0.423***	***PPV***
7	24	CE,TL	MAX	0.414	0.916	0.216	0.368	0.196	0.372	0.637	0.534	PPV
8	16	FL,TL	MAX	0.377	0.919	0.216	0.270	0.160	0.318	0.626	0.519	PPV
9	17	FL,DL	MAX	0.349	0.940	0.165	0.247	0.165	0.228	0.637	0.458	PPV
			MAX Result	***0.380***	***0.925***	***0.199***	***0.295***	***0.174***	***0.306***	***0.633***	***0.504***	***PPV***
10	16	FL,TL	MIN	0.351	0.948	0.053	0.308	0.069	0.377	0.640	0.426	PPV
11	94	HL_1,FL	MIN	0.342	0.923	0.093	0.316	0.063	0.317	0.617	0.430	PPV
12	100	HL,CE	MIN	0.332	0.931	0.004	0.250	0.000	0.473	0.485	0.407	PPV
			MIN Result	***0.342***	***0.934***	***0.050***	***0.291***	***0.044***	***0.389***	***0.581***	***0.421***	***PPV***
13	37	FL,DL	NWS	0.351	0.962	0.080	0.257	0.158	0.298	0.649	0.446	PPV
14	35	FL,TL	NWS	0.349	0.947	0.093	0.252	0.150	0.302	0.582	0.462	PPV
15	18	FL,DL	NWS	0.347	0.916	0.086	0.320	0.135	0.277	0.617	0.448	PPV
			NWS Result	**0.349**	**0.942**	**0.086**	**0.276**	**0.148**	**0.292**	**0.616**	**0.452**	**PPV**
16	29	CE,TL	WS	0.394	0.948	0.226	0.286	0.196	0.315	0.626	0.534	PPV
17	35	FL,TL	WS	0.381	0.933	0.124	0.303	0.180	0.366	0.609	0.505	PPV
18	69	DL,BL	WS	0.370	0.964	0.096	0.284	0.155	0.352	0.676	0.453	PPV
			WS Result	***0.382***	***0.948***	***0.149***	***0.291***	***0.177***	***0.344***	***0.637***	***0.497***	***PPV***
			Grand Average	***0.362***	***0.937***	***0.110***	***0.286***	***0.121***	***0.354***	***0.607***	***0.463***	***PPV***

Tab. G.20. Table contains 18 trained models representing the top 3 double loss combinations for every discrete merge strategy. The columns $IoU_0, \ldots, IoU_4$ represent the five classes, Background, Haemorrhages, Hard Exudates, Microaneurys and Optic Disc. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

No.	R-time(m)	Loss	Strategy	IoU	IoU 0	IoU 1	IoU 2	IoU 3	IoU 4	PPV	TPR	PPV vs. TPR
1	66	FL,DL,BL	AVG	0.411	0.948	0.223	0.325	0.185	0.375	0.667	0.520	PPV
2	39	CE,DL,BL	AVG	0.387	0.935	0.069	0.306	0.136	0.488	0.612	0.498	PPV
3	47	CE,DL,BL	AVG	0.366	0.956	0.131	0.265	0.159	0.318	0.594	0.482	PPV
			AVG Result	***0.388***	***0.947***	***0.141***	***0.299***	***0.160***	***0.394***	***0.625***	***0.500***	***PPV***
4	40	FL,TL,BL	HARMONIC	0.320	0.928	0.003	0.215	0.000	0.455	0.463	0.408	PPV
5	167	FL,DL,HL_1	HARMONIC	0.318	0.940	0.009	0.249	0.008	0.386	0.545	0.404	PPV
6	39	CE,DL,BL	HARMONIC	0.318	0.924	0.005	0.132	0.000	0.527	0.454	0.397	PPV
			HARMONIC Result	***0.319***	***0.931***	***0.006***	***0.199***	***0.003***	***0.456***	***0.487***	***0.403***	***PPV***
7	36	CE,TL,BL	MAX	0.324	0.866	0.056	0.247	0.066	0.385	0.490	0.451	PPV
8	45	CE,DL,BL	MAX	0.296	0.943	0.075	0.229	0.072	0.162	0.515	0.412	PPV
9	108	FL,DL,HL_1	MAX	0.288	0.928	0.001	0.264	0.000	0.247	0.406	0.379	PPV
			MAX Result	***0.303***	***0.912***	***0.044***	***0.247***	***0.046***	***0.265***	***0.470***	***0.414***	***PPV***
10	61	FL,TL,HL	MIN	0.334	0.944	0.002	0.276	0.030	0.417	0.554	0.422	PPV
11	61	FL,DL,HL	MIN	0.332	0.946	0.002	0.281	0.008	0.424	0.567	0.404	PPV
12	84	FL,DL,HL_1	MIN	0.304	0.955	0.002	0.240	0.035	0.288	0.498	0.392	PPV
			MIN Result	***0.323***	***0.949***	***0.002***	***0.266***	***0.024***	***0.376***	***0.540***	***0.406***	***PPV***
13	67	FL,DL,BL	NWS	0.363	0.935	0.054	0.281	0.121	0.421	0.587	0.459	PPV
14	94	CE,TL,BL	NWS	0.359	0.955	0.135	0.278	0.176	0.250	0.622	0.476	PPV
15	39	CE,TL,BL	NWS	0.349	0.944	0.199	0.236	0.175	0.189	0.631	0.487	PPV
			NWS Result	***0.357***	***0.945***	***0.129***	***0.265***	***0.157***	***0.287***	***0.613***	***0.474***	***PPV***
16	75	FL,DL,BL	WS	0.368	0.936	0.107	0.291	0.135	0.370	0.617	0.475	PPV
17	38	CE,TL,BL	WS	0.367	0.885	0.111	0.304	0.126	0.406	0.604	0.482	PPV
18	108	CE,TL,BL	WS	0.352	0.952	0.141	0.252	0.123	0.294	0.554	0.459	PPV
			WS Result	***0.362***	***0.925***	***0.119***	***0.282***	***0.128***	***0.357***	***0.592***	***0.472***	***PPV***
			Grand Average	***0.342***	***0.934***	***0.074***	***0.259***	***0.086***	***0.356***	***0.554***	***0.445***	***PPV***

Tab. G.21. Table contains 18 trained models representing the top 3 triple loss combinations for every discrete merge strategy. The columns $IoU_0, \ldots, IoU_4$ represent the five classes, Background, Haemorrhages, Hard Exudates, Microaneurys and Optic Disc. The column titled »*PPV vs. TPR*«" illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

G.3.3. Continuous

No.	R-time(m)	Loss	IoU	IoU 0	IoU 1	IoU 2	IoU 3	IoU 4	PPV	TPR	PPV vs. TPR	pbm_alpha	pbm_I
1	68	BL,CE	0.318	0.939	0.002	0.211	0.000	0.438	0.471	0.391	PPV	2.000	1.000
2	34	BL,CE	0.313	0.943	0.001	0.223	0.000	0.397	0.489	0.373	PPV	2.000	1.000
3	35	BL,CE	0.302	0.924	0.020	0.223	0.000	0.342	0.443	0.377	PPV	3.000	1.000
		BL,CE Result	***0.311***	***0.935***	***0.008***	***0.219***	***0.000***	***0.392***	***0.468***	***0.380***	***PPV***	***2.333***	***1.000***
4	26	BL,FL	0.339	0.933	0.052	0.294	0.028	0.390	0.563	0.417	PPV	3.000	0.832
5	23	BL,FL	0.307	0.946	0.001	0.107	0.000	0.480	0.451	0.363	PPV	4.000	0.999
6	23	BL,FL	0.296	0.963	0.001	0.049	0.000	0.467	0.400	0.328	PPV	4.000	0.000
		BL,FL Result	***0.314***	***0.947***	***0.018***	***0.150***	***0.009***	***0.446***	***0.472***	***0.369***	***PPV***	***3.667***	***0.610***
7	13	CE,DL	0.384	0.927	0.087	0.336	0.119	0.453	0.668	0.474	PPV	1.000	0.000
8	17	CE,DL	0.354	0.935	0.092	0.294	0.106	0.342	0.602	0.443	PPV	3.000	1.000
9	34	CE,DL	0.351	0.928	0.127	0.311	0.135	0.253	0.625	0.455	PPV	2.000	1.000
		CE,DL Result	***0.363***	***0.930***	***0.102***	***0.314***	***0.120***	***0.349***	***0.632***	***0.458***	***PPV***	***2.000***	***0.667***
10	17	CE,TL	0.402	0.962	0.066	0.320	0.096	0.567	0.647	0.468	PPV	3.000	1.000
11	12	CE,TL	0.366	0.921	0.072	0.319	0.061	0.458	0.582	0.474	PPV	1.000	0.000
12	40	CE,TL	0.357	0.926	0.164	0.303	0.166	0.227	0.585	0.478	PPV	3.861	0.859
		CE,TL Result	***0.375***	***0.936***	***0.101***	***0.314***	***0.107***	***0.417***	***0.605***	***0.473***	***PPV***	***2.620***	***0.620***
13	33	DL,BL	0.338	0.919	0.021	0.258	0.006	0.486	0.503	0.438	PPV	1.000	1.000
14	89	DL,BL	0.320	0.952	0.126	0.269	0.129	0.123	0.550	0.403	PPV	2.538	0.974
15	35	DL,BL	0.308	0.933	0.012	0.231	0.050	0.315	0.522	0.413	PPV	3.000	1.000
		DL,BL Result	***0.322***	***0.934***	***0.053***	***0.253***	***0.062***	***0.308***	***0.525***	***0.418***	***PPV***	***2.179***	***0.991***
16	106	DL,HL_1	0.277	0.923	0.000	0.000	0.000	0.464	0.327	0.345	TPR	6.342	0.711
17	103	DL,HL_1	0.269	0.916	0.000	0.000	0.000	0.426	0.309	0.355	TPR	7.381	0.615
18	145	DL,HL_1	0.218	0.799	0.000	0.045	0.000	0.245	0.357	0.365	TPR	5.547	0.713
		DL,HL_1 Result	***0.255***	***0.879***	***0.000***	***0.015***	***0.000***	***0.378***	***0.331***	***0.355***	***TPR***	***6.423***	***0.680***
19	14	FL,DL	0.400	0.943	0.155	0.358	0.084	0.458	0.650	0.480	PPV	3.000	0.000
20	18	FL,DL	0.376	0.900	0.151	0.347	0.046	0.436	0.570	0.519	PPV	5.000	1.000
21	15	FL,DL	0.347	0.924	0.011	0.304	0.024	0.473	0.602	0.441	PPV	1.000	1.000
		FL,DL Result	***0.374***	***0.922***	***0.106***	***0.336***	***0.051***	***0.456***	***0.607***	***0.480***	***PPV***	***3.000***	***0.667***
22	40	FL,TL	0.389	0.893	0.174	0.363	0.138	0.375	0.570	0.519	PPV	7.674	0.960
23	34	FL,TL	0.388	0.920	0.211	0.352	0.143	0.316	0.558	0.507	PPV	2.000	1.000
24	17	FL,TL	0.383	0.929	0.166	0.286	0.190	0.342	0.581	0.533	PPV	2.000	0.000
		FL,TL Result	***0.387***	***0.914***	***0.184***	***0.334***	***0.157***	***0.344***	***0.570***	***0.520***	***PPV***	***3.891***	***0.653***
25	52	HL_1,CE	0.278	0.941	0.002	0.089	0.000	0.356	0.402	0.344	PPV	4.000	1.000
26	54	HL_1,CE	0.276	0.909	0.002	0.128	0.000	0.343	0.352	0.368	TPR	2.000	1.000
27	109	HL_1,CE	0.268	0.961	0.000	0.000	0.000	0.379	0.348	0.305	PPV	8.712	0.911
		HL_1,CE Result	***0.274***	***0.937***	***0.001***	***0.072***	***0.000***	***0.359***	***0.367***	***0.339***	***PPV***	***4.904***	***0.970***
28	56	HL_1,FL	0.345	0.957	0.001	0.308	0.000	0.462	0.462	0.430	PPV	2.000	1.000
29	54	HL_1,FL	0.322	0.920	0.001	0.309	0.005	0.375	0.542	0.400	PPV	1.000	1.000
30	59	HL_1,FL	0.304	0.931	0.002	0.246	0.008	0.333	0.463	0.395	PPV	2.000	1.000
		HL_1,FL Result	***0.324***	***0.936***	***0.001***	***0.288***	***0.004***	***0.390***	***0.489***	***0.408***	***PPV***	***1.667***	***1.000***
31	36	TL,BL	0.313	0.920	0.049	0.266	0.000	0.329	0.479	0.424	PPV	4.000	1.000
32	69	TL,BL	0.297	0.914	0.039	0.265	0.000	0.269	0.450	0.410	PPV	2.000	1.000
33	33	TL,BL	0.281	0.917	0.044	0.228	0.000	0.216	0.440	0.395	PPV	1.000	1.000
		TL,BL Result	***0.297***	***0.917***	***0.044***	***0.253***	***0.000***	***0.271***	***0.456***	***0.410***	***PPV***	***2.333***	***1.000***
34	99	TL,HL_1	0.266	0.920	0.000	0.004	0.000	0.405	0.366	0.326	PPV	6.495	0.851
35	82	TL,HL_1	0.250	0.963	0.000	0.000	0.000	0.287	0.337	0.273	PPV	6.291	0.187
36	50	TL,HL_1	0.243	0.933	0.000	0.000	0.000	0.283	0.284	0.299	TPR	3.000	1.000
		TL,HL_1 Result	***0.253***	***0.938***	***0.000***	***0.001***	***0.000***	***0.325***	***0.329***	***0.300***	***PPV***	***5.262***	***0.679***
		Grand Average	***0.321***	***0.927***	***0.051***	***0.212***	***0.043***	***0.370***	***0.488***	***0.409***	***PPV***	***3.357***	***0.795***

Tab. G.22. Table lists the top 3 results for every double loss combination using the performance-based continuous merge strategy. The columns $IoU_0, \ldots, IoU_4$ represent the five classes, Background, Haemorrhages, Hard Exudates, Microaneurys and Optic Disc. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

No.	R-time(m)	Loss	IoU	IoU 0	IoU 1	IoU 2	IoU 3	IoU 4	PPV	TPR	PPV vs. TPR	pbm_alpha	pbm_l
1	36	CE,DL,BL	0.350	0.947	0.002	0.293	0.042	0.464	0.539	0.438	PPV	3.000	1.000
2	25	CE,DL,BL	0.322	0.957	0.000	0.007	0.000	0.646	0.362	0.346	PPV	5.000	0.000
3	39	CE,DL,BL	0.320	0.932	0.013	0.264	0.044	0.350	0.561	0.407	PPV	2.000	1.000
		CE,DL,BL Result	***0.331***	***0.945***	***0.005***	***0.188***	***0.029***	***0.487***	***0.488***	***0.397***	***PPV***	***3.333***	***0.667***
4	66	CE,DL,HL_1	0.267	0.959	0.000	0.000	0.000	0.378	0.352	0.306	PPV	2.000	1.000
5	49	CE,DL,HL_1	0.248	0.960	0.000	0.000	0.000	0.281	0.342	0.278	PPV	1.000	1.000
6	52	CE,DL,HL_1	0.233	0.953	0.000	0.000	0.000	0.210	0.331	0.267	PPV	5.000	1.000
		CE,DL,HL_1 Result	***0.249***	***0.957***	***0.000***	***0.000***	***0.000***	***0.289***	***0.342***	***0.284***	***PPV***	***2.667***	***1.000***
7	36	CE,TL,BL	0.348	0.937	0.130	0.307	0.115	0.249	0.602	0.432	PPV	4.000	1.000
8	36	CE,TL,BL	0.331	0.917	0.024	0.300	0.052	0.364	0.533	0.428	PPV	5.000	1.000
9	36	CE,TL,BL	0.312	0.958	0.001	0.172	0.000	0.430	0.414	0.363	PPV	3.000	1.000
		CE,TL,BL Result	***0.330***	***0.937***	***0.052***	***0.260***	***0.056***	***0.348***	***0.516***	***0.408***	***PPV***	***4.000***	***1.000***
10	52	CE,TL,HL_1	0.282	0.946	0.000	0.105	0.000	0.357	0.361	0.360	PPV	3.000	1.000
11	114	CE,TL,HL_1	0.275	0.923	0.000	0.084	0.000	0.367	0.401	0.356	PPV	9.228	0.700
12	89	CE,TL,HL_1	0.258	0.939	0.000	0.081	0.000	0.270	0.404	0.327	PPV	7.199	0.922
		CE,TL,HL_1 Result	***0.272***	***0.936***	***0.000***	***0.090***	***0.000***	***0.331***	***0.388***	***0.348***	***PPV***	***6.476***	***0.874***
13	34	FL,DL,BL	0.377	0.934	0.046	0.324	0.088	0.493	0.629	0.485	PPV	1.000	1.000
14	69	FL,DL,BL	0.338	0.932	0.144	0.282	0.078	0.254	0.586	0.429	PPV	2.000	1.000
15	76	FL,DL,BL	0.317	0.932	0.004	0.174	0.000	0.473	0.432	0.411	PPV	6.470	0.654
		FL,DL,BL Result	***0.344***	***0.932***	***0.065***	***0.260***	***0.055***	***0.407***	***0.549***	***0.442***	***PPV***	***3.157***	***0.885***
16	85	FL,DL,HL_1	0.327	0.946	0.000	0.229	0.000	0.459	0.447	0.407	PPV	9.634	0.853
17	103	FL,DL,HL_1	0.298	0.941	0.000	0.231	0.000	0.318	0.452	0.387	PPV	6.156	0.680
18	78	FL,DL,HL_1	0.274	0.863	0.000	0.068	0.000	0.440	0.350	0.393	TPR	8.711	0.066
		FL,DL,HL_1 Result	***0.300***	***0.917***	***0.000***	***0.176***	***0.000***	***0.406***	***0.416***	***0.396***	***PPV***	***8.167***	***0.533***
19	72	FL,TL,BL	0.402	0.931	0.179	0.329	0.121	0.448	0.643	0.515	PPV	5.887	0.830
20	25	FL,TL,BL	0.360	0.959	0.095	0.271	0.000	0.477	0.499	0.457	PPV	5.000	0.744
21	37	FL,TL,BL	0.360	0.941	0.022	0.310	0.057	0.468	0.572	0.440	PPV	5.000	1.000
		FL,TL,BL Result	***0.374***	***0.943***	***0.099***	***0.303***	***0.059***	***0.464***	***0.571***	***0.471***	***PPV***	***5.296***	***0.858***
22	40	FL,TL,HL_1	0.285	0.952	0.000	0.106	0.000	0.368	0.408	0.342	PPV	1.000	0.000
23	113	FL,TL,HL_1	0.274	0.904	0.000	0.113	0.000	0.354	0.352	0.381	TPR	4.456	0.791
24	72	FL,TL,HL_1	0.261	0.963	0.000	0.000	0.000	0.344	0.300	0.301	TPR	2.000	1.000
		FL,TL,HL_1 Result	***0.274***	***0.940***	***0.000***	***0.073***	***0.000***	***0.355***	***0.353***	***0.341***	***PPV***	***2.485***	***0.597***
		Grand Average	***0.309***	***0.938***	***0.028***	***0.169***	***0.025***	***0.386***	***0.453***	***0.386***	***PPV***	***4.448***	***0.802***

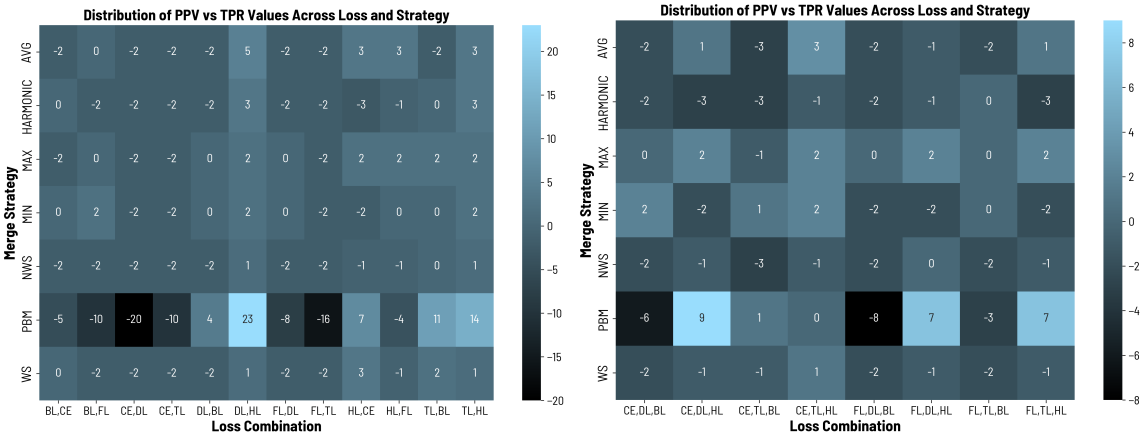
Tab. G.23. Table lists the top 3 results for every triple loss combination using the performance-based continuous merge strategy. The columns $IoU_0, \ldots, IoU_4$ represent the five classes, Background, Haemorrhages, Hard Exudates, Microaneurys and Optic Disc. The column titled »*PPV vs. TPR*« illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

G.3.4. Specific

No.	R-time(m)	Loss	Strategy	IoU	IoU 0	IoU 1	IoU 2	IoU 3	IoU 4	PPV	TPR	PPV vs. TPR
1	51	CE,DL	AVG	0.398	0.964	0.083	0.323	0.097	0.521	0.646	0.462	PPV
2	39	CE,DL	AVG	0.378	0.955	0.015	0.297	0.092	0.532	0.568	0.458	PPV
3	38	CE,DL	AVG	0.377	0.964	0.071	0.326	0.105	0.417	0.707	0.429	PPV
4	38	CE,DL	AVG	0.376	0.969	0.038	0.276	0.113	0.487	0.662	0.440	PPV
5	52	CE,DL	AVG	0.376	0.956	0.178	0.249	0.189	0.307	0.647	0.460	PPV
6	47	CE,DL	AVG	0.374	0.948	0.034	0.274	0.059	0.553	0.539	0.474	PPV
			Grand Average	***0.380***	***0.959***	***0.070***	***0.291***	***0.109***	***0.470***	***0.628***	***0.454***	***PPV***

Tab. G.24. Table lists six models trained repetitively on the loss combination CE, DL using the arithmetic averaging (AVG) strategy for merging. These results serve as part of a final comparison against baseline results providing an identical iteration number. The columns $IoU_0, \ldots, IoU_4$ represent the five classes, Background, Haemorrhages, Hard Exudates, Microaneurys and Optic Disc. The column titled »*PPV vs. TPR*«" illustrates the trade-off between a high Positive Predictive Value (PPV) and low True Positive Rate (TPR), or vice versa, for each model.

G.3.5. PPV vs. TPR



(a) Distribution of PPV vs. TPR for double losses

(b) Distribution of PPV vs. TPR for triple losses

Fig. G.5. Heatmaps visualizing the relative prevalence as a difference map of PPV and TPR across various loss combinations and merge strategies. Each cell represents a unique combination. Positive values indicate a predominance of TPR, while negative values signify a predominance of PPV.

H. Bibliography

[1] *12th IEEE International Symposium on Biomedical Imaging, ISBI 2015, Brooklyn, NY, USA, April 16-19, 2015*. IEEE, 2015. ISBN: 978-1-4799-2374-8. URL: https://ieeexplore.ieee.org/xpl/conhome/7150573/proceeding.

[2] *2D Semantic Label. - Vaihingen*. [Online; accessed 14. Mar. 2023]. Mar. 2023. URL: https://www.isprs.org/education/benchmarks/UrbanSemLab/2d-sem-label-vaihingen.aspx.

[3] Hervé Abdi and Lynne J. Williams. "Principal component analysis". In: *WIREs Computational Statistics* 2.4 (2010), pp. 433–459. DOI: https://doi.org/10.1002/wics.101.

[4] Sharat Agarwal et al. "Contextual Diversity for Active Learning". In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi et al. Cham: Springer International Publishing, 2020, pp. 137–153. ISBN: 978-3-030-58517-4. DOI: 10.48550/arXiv.2008.05723.

[5] Jiwoon Ahn and Suha Kwak. "Learning Pixel-level Semantic Affinity with Image-level Supervision for Weakly Supervised Semantic Segmentation". In: *CoRR* abs/1803.10464 (2018). DOI: 10.48550/arXiv.1803.10464.

[6] Neziha Akalin and Amy Loutfi. "Reinforcement Learning Approaches in Social Robotics". In: *Sensors* 21.4 (2021). ISSN: 1424-8220. DOI: 10.3390/s21041292.

[7] Hussain Alibrahim and Simone A. Ludwig. "Hyperparameter Optimization: Comparing Genetic Algorithm against Grid Search and Bayesian Optimization". In: *2021 IEEE Congress on Evolutionary Computation (CEC)*. 2021, pp. 1551–1559. DOI: 10.1109/CEC45853.2021.9504761.

[8] Hussein Almuallim, Shigeo Kaneda, and Yasuhiro Akiba. "3 - Development and Applications of Decision Trees". In: *Expert Systems*. Ed. by Cornelius T. Leondes. Burlington: Academic Press, 2002, pp. 53–77. ISBN: 978-0-12-443880-4. DOI: https://doi.org/10.1016/B978-012443880-4/50047-8.

[9] Omar Alonso. "Challenges with Label Quality for Supervised Learning". In: *J. Data and Information Quality* 6.1 (Mar. 2015). ISSN: 1936-1955. DOI: 10.1145/2724721.

[10] Ethem Alpaydin. *Introduction to machine learning*. MIT press, 2020.

[11] Marcin Andrychowicz et al. "Hindsight Experience Replay". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/453fadbd8a1a3af50a9df4df899537b5-Paper.pdf.

[12] Saeid Asgari Taghanaki et al. "Deep semantic segmentation of natural and medical images: a review". In: *Artificial Intelligence Review* 54.1 (Jan. 2021), pp. 137–178. ISSN: 1573-7462. DOI: 10.1007/s10462-020-09854-1.

[13] Mariette Awad and Rahul Khanna. "Support Vector Machines for Classification". In: *Efficient Learning Machines: Theories, Concepts, and Applications for Engineers and System Designers*. Berkeley, CA: Apress, 2015, pp. 39–66. ISBN: 978-1-4302-5990-9. DOI: 10.1007/978-1-4302-5990-9_3.

[14] Noor H. Awad, Neeratyoy Mallik, and Frank Hutter. "DEHB: Evolutionary Hyberband for Scalable, Robust and Efficient Hyperparameter Optimization". In: *CoRR* abs/2105.09821 (2021). DOI: 10.48550/arXiv.2105.09821.

[15] Vijay Badrinarayanan, Ignas Budvytis, and Roberto Cipolla. "Semi-Supervised Video Segmentation Using Tree Structured Graphical Models". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 35.11 (2013), pp. 2751–2764. DOI: 10.1109/TPAMI.2013.54.

[16] Spyridon Bakas et al. *Identifying the Best Machine Learning Algorithms for Brain Tumor Segmentation, Progression Assessment, and Overall Survival Prediction in the BRATS Challenge*. 2018. DOI: 10.48550/ARXIV.1811.02629.

[17] Spyridon Bakas et al. "Identifying the Best Machine Learning Algorithms for Brain Tumor Segmentation, Progression Assessment, and Overall Survival Prediction in the BRATS Challenge". In: *CoRR* abs/1811.02629 (2018). DOI: 10.48550/arXiv.1811.02629.

[18] Jesús Balado et al. "Road Environment Semantic Segmentation with Deep Learning from MLS Point Cloud Data". In: *Sensors* 19.16 (2019). ISSN: 1424-8220. DOI: 10.3390/s19163466.

[19] Suresh Balakrishnama and Aravind Ganapathiraju. "Linear discriminant analysis-a brief tutorial". In: *Institute for Signal and information Processing* 18.1998 (1998), pp. 1–8. URL: https://www.researchgate.net/publication/240093048_Linear_Discriminant_Analysis-A_Brief_Tutorial.

[20] Bjorn Barz and Joachim Denzler. "Deep Learning on Small Datasets without Pre-Training using Cosine Loss". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. Mar. 2020. DOI: 10.48550/arXiv.1901.09054.

[21] Yoshua Bengio, Yann Lecun, and Geoffrey Hinton. "Deep Learning for AI". In: *Commun. ACM* 64.7 (June 2021), pp. 58–65. ISSN: 0001-0782. DOI: 10.1145/3448250.

[22] James Bergstra and Yoshua Bengio. "Random Search for Hyper-Parameter Optimization". In: *J. Mach. Learn. Res.* 13.null (Feb. 2012), pp. 281–305. ISSN: 1532-4435. DOI: 10.5555/2188385.2188395.

[23] Petra Bevandic et al. *Simultaneous Semantic Segmentation and Outlier Detection in Presence of Domain Shift*. 2019. DOI: 10.48550/arXiv.1908.01098.

[24] Dinesh Bhuriya et al. "Stock market predication using a linear regression". In: *2017 international conference of electronics, communication and aerospace technology (ICECA)*. Vol. 2. IEEE. 2017, pp. 510–513. DOI: 10.1109/ICECA.2017.8212716.

[25] Nils Bjorck et al. "Understanding Batch Normalization". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018. DOI: 10.48550/arXiv.1806.02375.

[26] Hermann Blum et al. "Fishyscapes: A benchmark for safe semantic segmentation in autonomous driving". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision Workshops*. 2019, pp. 0–0. DOI: 10.1109/ICCVW.2019.00294.

[27] Daniel Bolya et al. "Yolact: Real-time instance segmentation". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 9157–9166. DOI: 10.48550/arXiv.1904.02689.

[28] Edwin G Boring. "A new ambiguous figure." In: *The American Journal of Psychology* (1930). DOI: 10.2307/1415447.

[29] Léon Bottou. "Stochastic Gradient Descent Tricks". In: *Neural Networks: Tricks of the Trade: Second Edition*. Ed. by Grégoire Montavon, Geneviève B. Orr, and Klaus-Robert Müller. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 421–436. ISBN: 978-3-642-35289-8. DOI: 10.1007/978-3-642-35289-8_25.

[30] Craig Boutilier, Kevin Regan, and Paolo Viappiani. "Online Feature Elicitation in Interactive Optimization". In: *Proceedings of the 26th Annual International Conference on Machine Learning*. ICML '09. Montreal, Quebec, Canada: Association for Computing Machinery, 2009, pp. 73–80. ISBN: 9781605585161. DOI: 10.1145/1553374.1553384.

[31] Yuri Boykov et al. "An Integral Solution to Surface Evolution PDEs Via Geo-cuts". In: *Computer Vision – ECCV 2006*. Ed. by Aleš Leonardis, Horst Bischof, and Axel Pinz. Berlin, Heidelberg: Springer Berlin Heidelberg, 2006, pp. 409–422. ISBN: 978-3-540-33837-6. DOI: 10.1007/11744078_32.

[32] Gary Bradski. "The openCV library." In: *Dr. Dobb's Journal: Software Tools for the Professional Programmer* 25.11 (2000). [Online; accessed 2. Jul. 2023], pp. 120–123. URL: https://www.scirp.org/(S(351jmbntvnsjt1aadkposzje))/reference/ReferencesPapers.aspx?ReferenceID=1692176.

[33] Leo Breiman. "Bagging predictors". In: *Machine Learning* 24.2 (Aug. 1996), pp. 123–140. ISSN: 1573-0565. DOI: 10.1007/BF00058655.

[34] Jerrin Bright. "Instance Segmentation with Custom Datasets in Python - Value ML". In: *Value ML* (Nov. 2020). URL: https://valueml.com/instance-segmentation-with-custom-datasets-in-python.

[35] Gabriel J. Brostow, Julien Fauqueur, and Roberto Cipolla. "Semantic object classes in video: A high-definition ground truth database". In: *Pattern Recognition Letters* 30.2 (2009). Video-based Object and Event Analysis, pp. 88–97. ISSN: 0167-8655. DOI: https://doi.org/10.1016/j.patrec.2008.04.005.

[36] Mateusz Buda, Atsuto Maki, and Maciej A. Mazurowski. "A systematic study of the class imbalance problem in convolutional neural networks". In: *CoRR* abs/1710.05381 (2017). DOI: 10.1016/j.neunet.2018.07.011.

[37] Samuel Budd, Emma C. Robinson, and Bernhard Kainz. "A Survey on Active Learning and Human-in-the-Loop Deep Learning for Medical Image Analysis". In: *CoRR* abs/1910.02923 (2019). DOI: 10.1016/j.media.2021.102062.

[38] Ignas Budvytis et al. "Large Scale Labelled Video Data Augmentation for Semantic Segmentation in Driving Scenarios". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV) Workshops*. Oct. 2017. DOI: 10.1109/ICCVW.2017.36.

[39] Míriam Bellver Bueno et al. "Hierarchical object detection with deep reinforcement learning". In: *Deep Learning for Image Processing Applications* 31.164 (2017), p. 3. DOI: 10.48550/arXiv.1611.03718.

[40] Lucian Busoniu et al. *Reinforcement learning and dynamic programming using function approximators*. CRC press, 2017. DOI: 10.1201/9781439821091.

[41] Yutong Cai and Yong Wang. *MA-Unet: An improved version of Unet based on multi-scale and attention mechanism for medical image segmentation*. 2020. DOI: 10.48550/ARXIV.2012.10952.

[42] Tom Cane and James Ferryman. "Evaluating deep semantic segmentation networks for object detection in maritime surveillance". In: *2018 15th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*. 2018, pp. 1–6. DOI: 10.1109/AVSS.2018.8639077.

[43] Aaron Carass et al. "Evaluating White Matter Lesion Segmentations with Refined Sørensen-Dice Analysis". en. In: *Sci Rep* 10.1 (May 2020), p. 8242. DOI: 10.1038/s41598-020-64803-w.

[44] Noe Casas. "Deep Deterministic Policy Gradient for Urban Traffic Light Control". In: *CoRR* abs/1703.09035 (2017). DOI: 10.48550/arXiv.1703.09035.

[45] Robin Chan et al. "Application of Decision Rules for Handling Class Imbalance in Semantic Segmentation". In: *CoRR* abs/1901.08394 (2019). DOI: 10.48550/arXiv.1901.08394.

[46] Angel X. Chang et al. "Matterport3D: Learning from RGB-D Data in Indoor Environments". In: *CoRR* abs/1709.06158 (2017). DOI: 10.48550/arXiv.1709.06158.

[47] Bike Chen, Chen Gong, and Jian Yang. "Importance-Aware Semantic Segmentation for Autonomous Vehicles". In: *IEEE Transactions on Intelligent Transportation Systems* 20.1 (2019), pp. 137–148. DOI: 10.1109/TITS.2018.2801309.

[48] Cheng Chen et al. "Synergistic Image and Feature Adaptation: Towards Cross-Modality Domain Adaptation for Medical Image Segmentation". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 33.01 (July 2019), pp. 865–872. DOI: 10.1609/aaai.v33i01.3301865.

[49] Dong Chen et al. "Supervised Transformer Network for Efficient Face Detection". In: *Computer Vision – ECCV 2016*. Ed. by Bastian Leibe et al. Cham: Springer International Publishing, 2016, pp. 122–138. ISBN: 978-3-319-46454-1. DOI: 10.48550/arXiv.1607.05477.

[50] Hongming Chen et al. "The rise of deep learning in drug discovery". In: *Drug Discovery Today* 23.6 (2018), pp. 1241–1250. ISSN: 1359-6446. DOI: 10.1016/j.drudis.2018.01.039.

[51] Liang-Chieh Chen et al. "DeepLab: Semantic Image Segmentation with Deep Convolutional Nets, Atrous Convolution, and Fully Connected CRFs". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40.4 (2018), pp. 834–848. DOI: 10.1109/TPAMI.2017.2699184.

[52] Liang-Chieh Chen et al. "Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation". In: *CoRR* abs/1802.02611 (2018). DOI: 10.48550/arXiv.1802.02611.

[53] Tianping Chen and Hong Chen. "Universal approximation to nonlinear operators by neural networks with arbitrary activation functions and its application to dynamical systems". In: *IEEE Transactions on Neural Networks* 6.4 (1995), pp. 911–917. DOI: 10.1109/72.392253.

[54] Yang Chen et al. "Discriminative feature representation: an effective postprocessing solution to low dose CT imaging*". In: *Physics in Medicine and Biology* 62.6 (Feb. 2017), p. 2103. DOI: 10.1088/1361-6560/aa5c24.

[55] Wonwoo Cho, Jeonghoon Park, and Jaegul Choo. "Training Auxiliary Prototypical Classifiers for Explainable Anomaly Detection in Medical Image Segmentation". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*. Jan. 2023, pp. 2624–2633. DOI: 10.1109/WACV56688.2023.00265.

[56] Jan K Chorowski et al. "Attention-Based Models for Speech Recognition". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc., 2015. DOI: 10.48550/arXiv.1506.07503.

[57] Diya Chudasama et al. "Image segmentation using morphological operations". In: *International Journal of Computer Applications* 117.18 (2015). DOI: 10.5120/20654-3197.

[58] Noel C. F. Codella et al. "Skin Lesion Analysis Toward Melanoma Detection: A Challenge at the 2017 International Symposium on Biomedical Imaging (ISBI), Hosted by the International Skin Imaging Collaboration (ISIC)". In: *CoRR* abs/1710.05006 (2017). DOI: 10.48550/arXiv.1605.01397.

[59] Tim F. Cootes et al. "Robust and Accurate Shape Model Fitting Using Random Forest Regression Voting". In: *Computer Vision – ECCV 2012*. Ed. by Andrew Fitzgibbon et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2012, pp. 278–291. ISBN: 978-3-642-33786-4. DOI: 10.1007/978-3-642-33786-4_21.

[60] Marius Cordts et al. "The cityscapes dataset for semantic urban scene understanding". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 3213–3223. DOI: 10.48550/arXiv.1604.01685.

[61] Corinna Cortes and Vladimir Vapnik. "Support-vector networks". In: *Machine Learning* 20.3 (Sept. 1995), pp. 273–297. ISSN: 1573-0565. DOI: 10.1007/BF00994018.

[62] Adele Cutler, D. Richard Cutler, and John R. Stevens. "Random Forests". In: *Ensemble Machine Learning: Methods and Applications*. Ed. by Cha Zhang and Yunqian Ma. Boston, MA: Springer US, 2012, pp. 157–175. ISBN: 978-1-4419-9326-7. DOI: 10.1007/978-1-4419-9326-7_5.

[63] Angela Dai et al. "ScanNet: Richly-annotated 3D Reconstructions of Indoor Scenes". In: *CoRR* abs/1702.04405 (2017). DOI: 10.48550/arXiv.1512.04412.

[64] Arno De Caigny, Kristof Coussement, and Koen W. De Bock. "A new hybrid classification algorithm for customer churn prediction based on logistic regression and decision trees". In: *European Journal of Operational Research* 269.2 (2018), pp. 760–772. ISSN: 0377-2217. DOI: 10.1016/j.ejor.2018.02.009.

[65] Li Deng and Dong Yu. "Deep Learning: Methods and Applications". In: *Foundations and Trends® in Signal Processing* 7.3–4 (2014), pp. 197–387. ISSN: 1932-8346. DOI: 10.1561/2000000039.

[66] Yao Deng, Huawei Liang, and Zhiyan Yi. "An Improved Approach for Object Proposals Generation". In: *Electronics* 10.7 (2021). ISSN: 2079-9292. DOI: 10.3390/electronics10070794.

[67] Yue Deng et al. "Deep Direct Reinforcement Learning for Financial Signal Representation and Trading". In: *IEEE Transactions on Neural Networks and Learning Systems* 28.3 (2017), pp. 653–664. DOI: 10.1109/TNNLS.2016.2522401.

[68] Aashish Dhawan, Pankaj Bodani, and Vishal Garg. "Post Processing of Image Segmentation using Conditional Random Fields". In: *2019 6th International Conference on Computing for Sustainable Global Development (INDIACom)*. 2019, pp. 729–734.

[69] Chris Donahue, Julian J. McAuley, and Miller S. Puckette. "Synthesizing Audio with Generative Adversarial Networks". In: *CoRR* abs/1802.04208 (2018). DOI: 10.48550/arXiv.1802.04208.

[70] Xibin Dong et al. "A survey on ensemble learning". In: *Frontiers of Computer Science* 14.2 (Apr. 2020), pp. 241–258. ISSN: 2095-2236. DOI: 10.1007/s11704-019-8208-z.

[71] Pinar Donmez, Jaime G. Carbonell, and Paul N. Bennett. "Dual Strategy Active Learning". In: *Machine Learning: ECML 2007*. Ed. by Joost N. Kok et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 2007, pp. 116–127. ISBN: 978-3-540-74958-5. DOI: 10.1007/978-3-540-74958-5_14.

[72] Getao Du et al. "Medical Image Segmentation based on U-Net: A Review". In: *Journal of Imaging Science and Technology* 64 (Mar. 2020). DOI: 10.2352/J.ImagingSci.Technol.2020.64.2.020508.

[73] Shiv Ram Dubey, Satish Kumar Singh, and Bidyut Baran Chaudhuri. "A Comprehensive Survey and Performance Analysis of Activation Functions in Deep Learning". In: *CoRR* abs/2109.14545 (2021). DOI: 10.48550/arXiv.2109.14545.

[74] Zach Eaton-Rosen et al. "Improving Data Augmentation for Medical Image Segmentation". In: *OpenReview* (June 2018). URL: https://openreview.net/forum?id=rkBBChjiG.

[75] Andreas Eitel et al. "Multimodal deep learning for robust RGB-D object recognition". In: *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2015, pp. 681–687. DOI: 10.1109/IROS.2015.7353446.

[76] Elizar Elizar et al. "A Review on Multiscale-Deep-Learning Applications". en. In: *Sensors (Basel)* 22.19 (Sept. 2022). DOI: 10.3390/s22197384.

[77] Andreas Ess et al. "Segmentation-Based Urban Traffic Scene Understanding." In: *BMVC*. Vol. 1. Citeseer. 2009, p. 2. DOI: 10.5244/C.23.84.

[78] M. Everingham et al. *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results*. http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html. URL: http://host.robots.ox.ac.uk/pascal/VOC/voc2012.

[79] Mark Everingham et al. "The Pascal Visual Object Classes Challenge: A Retrospective". In: *International Journal of Computer Vision* 111.1 (Jan. 2015), pp. 98–136. ISSN: 1573-1405. DOI: 10.1007/s11263-014-0733-5.

[80] Jianqing Fan et al. "A Theoretical Analysis of Deep Q-Learning". In: *Proceedings of the 2nd Conference on Learning for Dynamics and Control*. Ed. by Alexandre M. Bayen et al. Vol. 120. Proceedings of Machine Learning Research. PMLR, June 2020, pp. 486–489. URL: https://proceedings.mlr.press/v120/yang20a.html.

[81] Di Feng et al. "Deep Multi-Modal Object Detection and Semantic Segmentation for Autonomous Driving: Datasets, Methods, and Challenges". In: *IEEE Transactions on Intelligent Transportation Systems* 22.3 (2021), pp. 1341–1360. DOI: 10.1109/TITS.2020.2972974.

[82] Silvia Ferrari and Francisco Cribari-Neto. "Beta Regression for Modelling Rates and Proportions". In: *Journal of Applied Statistics* 31.7 (2004), pp. 799–815. DOI: 10.1080/0266476042000214501.

[83] Thibault Févry and Jason Phang. "Unsupervised Sentence Compression using Denoising Auto-Encoders". In: *CoRR* abs/1809.02669 (2018). DOI: 10.48550/arXiv.1809.02669.

[84] Santosh K Gaikwad, Bharti W Gawali, and Pravin Yannawar. "A review on speech recognition technique". In: *International Journal of Computer Applications* 10.3 (2010), pp. 16–24. DOI: 10.5120/1462-1976.

[85] Yarin Gal and Zoubin Ghahramani. "Dropout as a Bayesian Approximation: Representing Model Uncertainty in Deep Learning". In: *Proceedings of The 33rd International Conference on Machine Learning*. Ed. by Maria Florina Balcan and Kilian Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, New York, USA: PMLR, June 2016, pp. 1050–1059. URL: https://proceedings.mlr.press/v48/gal16.html.

[86] Bolin Gao and Lacra Pavel. *On the Properties of the Softmax Function with Application in Game Theory and Reinforcement Learning*. 2017. DOI: 10.48550/ARXIV.1704.00805.

[87] Hongyang Gao et al. "Pixel Transposed Convolutional Networks". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 42.5 (2020), pp. 1218–1227. DOI: 10.1109/TPAMI.2019.2893965.

[88] Alberto Garcia-Garcia et al. "A survey on deep learning techniques for image and video semantic segmentation". In: *Applied Soft Computing* 70 (2018), pp. 41–65. ISSN: 1568-4946. DOI: 10.1016/j.asoc.2018.05.018.

[89] Weifeng Ge, Sibei Yang, and Yizhou Yu. "Multi-Evidence Filtering and Fusion for Multi-Label Classification, Object Detection and Semantic Segmentation Based on Weakly Supervised Learning". In: *CoRR* abs/1802.09129 (2018). DOI: 10.48550/arXiv.1802.09129.

[90] Sina Ghassemi et al. "Learning and Adapting Robust Features for Satellite Image Segmentation on Heterogeneous Data Sets". In: *IEEE Transactions on Geoscience and Remote Sensing* 57.9 (2019), pp. 6517–6529. DOI: 10.1109/TGRS.2019.2906689.

[91] Amir Gholami et al. "A Novel Domain Adaptation Framework for Medical Image Segmentation". In: *CoRR* abs/1810.05732 (2018). DOI: 10.48550/arXiv.1810.05732.

[92] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016. URL: https://mitpress.mit.edu/9780262035613/deep-learning.

[93] Omer Gottesman et al. *Evaluating Reinforcement Learning Algorithms in Observational Health Settings*. 2018. DOI: 10.48550/ARXIV.1805.12298.

[94] L. Grady. "Random Walks for Image Segmentation". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 28.11 (2006), pp. 1768–1783. DOI: 10.1109/TPAMI.2006.233.

[95] Shane Griffith et al. "Policy Shaping: Integrating Human Feedback with Reinforcement Learning". In: *Advances in Neural Information Processing Systems*. Ed. by C.J. Burges et al. Vol. 26. Curran Associates, Inc., 2013. URL: https://proceedings.neurips.cc/paper_files/paper/2013/file/e034fb6b66aacc1d48f445ddfb08da98-Paper.pdf.

[96] Wenzhong Guo, Jinyu Cai, and Shiping Wang. "Unsupervised discriminative feature representation via adversarial auto-encoder". In: *Applied Intelligence* 50.4 (Apr. 2020), pp. 1155–1171. ISSN: 1573-7497. DOI: 10.1007/s10489-019-01581-7.

[97] John Hadden et al. "Churn prediction: Does technology matter?" In: *International Journal of Industrial and Manufacturing Engineering* 2.4 (2008), pp. 524–536. URL: https://www.researchgate.net/publication/283992554_Churn_Prediction_Does_Technology_Matter.

[98] Abdul Mueed Hafiz and Ghulam Mohiuddin Bhat. "A survey on instance segmentation: state of the art". In: *International Journal of Multimedia Information Retrieval* 9.3 (Sept. 2020), pp. 171–189. ISSN: 2192-662X. DOI: 10.1007/s13735-020-00195-x.

[99] John Michael Hammersley and David Christopher Handscomb. *Monte Carlo Methods*. London, England, UK: Methuen, 1964. ISBN: 978-0-41652340-9. URL: https://books.google.de/books/about/Monte_Carlo_Methods.html?id=Kk4OAAAAQAAJ&redir_esc=y.

[100] David R. Hardoon, Sandor Szedmak, and John Shawe-Taylor. "Canonical Correlation Analysis: An Overview with Application to Learning Methods". In: *Neural Computation* 16.12 (2004), pp. 2639–2664. DOI: 10.1162/0899766042321814.

[101] Hado van Hasselt, Arthur Guez, and David Silver. "Deep Reinforcement Learning with Double Q-Learning". In: *AAAI* 30.1 (Mar. 2016). ISSN: 2374-3468. DOI: 10.1609/aaai.v30i1.10295.

[102] Hado van Hasselt, Arthur Guez, and David Silver. "Deep Reinforcement Learning with Double Q-Learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 30.1 (Mar. 2016). DOI: 10.1609/aaai.v30i1.10295.

[103] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning*. New York, NY, USA: Springer. ISBN: 978-0-387-84858-7. URL: https://link.springer.com/book/10.1007/978-0-387-84858-7.

[104] Timm Haucke and Volker Steinhage. "Exploiting Depth Information for Wildlife Monitoring". In: *CoRR* abs/2102.05607 (2021). DOI: 10.48550/arXiv.2102.05607.

[105] Zeeshan Hayder, Xuming He, and Mathieu Salzmann. "Shape-aware Instance Segmentation". In: *CoRR* abs/1612.03129 (2016). DOI: 10.48550/arXiv.1612.03129.

[106] Marti A. Hearst. "Trends & Controversies: Support Vector Machines". In: *IEEE Intell. Syst.* 13 (1998), pp. 18–28. URL: https://www.semanticscholar.org/paper/Trends-%26-Controversies%3A-Support-Vector-Machines-Hearst/455d9a4ff96561d543acbcb2aa81d6cd8fcd20df.

[107] Johannes Heinrich and David Silver. "Deep Reinforcement Learning from Self-Play in Imperfect-Information Games". In: *CoRR* abs/1603.01121 (2016). DOI: 10.48550/arXiv.1603.01121.

[108] Philip Henson et al. "Anomaly detection to predict relapse risk in schizophrenia". In: *Translational Psychiatry* 11.1 (Jan. 2021), p. 28. ISSN: 2158-3188. DOI: 10.1038/s41398-020-01123-7.

[109] Samer Hijazi, Rishi Kumar, Chris Rowen, et al. "Using convolutional neural networks for image recognition". In: *Cadence Design Systems Inc.: San Jose, CA, USA* 9 (2015). URL: https://www.semanticscholar.org/paper/Using-Convolutional-Neural-Networks-for-Image-By-Hijazi-Kumar/bbf7b5bdc39f9b8849c639c11f4726e36915a0da.

[110] Derek Hoiem, Santosh K Divvala, and James H Hays. "Pascal VOC 2008 challenge". In: *World Literature Today* 24 (2009). URL: http://host.robots.ox.ac.uk/pascal/VOC/voc2008/index.html.

[111] *Home page for the book, "Bayesian Data Analysis"*. [Online; accessed 2. Jul. 2023]. July 2020. URL: http://www.stat.columbia.edu/~gelman/book.

[112] K D Hopper et al. "Analysis of interobserver and intraobserver variability in CT tumor measurements." In: *American Journal of Roentgenology* 167.4 (1996). PMID: 8819370, pp. 851–854. DOI: 10.2214/ajr.167.4.8819370.

[113] Roger A Horn. "The hadamard product". In: *Proc. Symp. Appl. Math*. Vol. 40. 1990, pp. 87–169. URL: https://scholar.google.com/citations?view_op=view_citation&hl=en&user=tLr2dXsAAAAJ&citation_for_view=tLr2dXsAAAAJ:qjMakFHDy7sC.

[114] Xiaoling Hu et al. "Topology-Aware Segmentation Using Discrete Morse Theory". In: *CoRR* abs/2103.09992 (2021). DOI: 10.48550/arXiv.2103.09992.

[115] Yuh-Jong Hu and Shang-Jen Lin. "Deep Reinforcement Learning for Optimizing Finance Portfolio Management". In: *2019 Amity International Conference on Artificial Intelligence (AICAI)*. 2019, pp. 14–20. DOI: 10.1109/AICAI.2019.8701368.

[116] Lun Huang et al. "Attention on attention for image captioning". In: *Proceedings of the IEEE/CVF international conference on computer vision*. 2019, pp. 4634–4643. DOI: 10.48550/arXiv.1908.06954.

[117] M.Q. Huang, J. Ninić, and Q.B. Zhang. "BIM, machine learning and computer vision techniques in underground construction: Current status and future perspectives". In: *Tunnelling and Underground Space Technology* 108 (2021), p. 103677. ISSN: 0886-7798. DOI: https://doi.org/10.1016/j.tust.2020.103677.

[118] J. D. Hunter. "Matplotlib: A 2D Graphics Environment". In: *Computing in Science & Engineering* 9.3 (May 2007), pp. 90–95. ISSN: 1558-366X. DOI: 10.1109/MCSE.2007.55.

[119] Sergey Ioffe and Christian Szegedy. "Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift". In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 448–456. URL: https://proceedings.mlr.press/v37/ioffe15.html.

[120] Shruti Jadon. "A survey of loss functions for semantic segmentation". In: *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*. IEEE, Oct. 2020. DOI: 10.1109/cibcb48159.2020.9277638.

[121] Jingkai Jia and Wenlin Wang. "Review of reinforcement learning research". In: *2020 35th Youth Academic Annual Conference of Chinese Association of Automation (YAC)*. 2020, pp. 186–191. DOI: 10.1109/YAC51587.2020.9337653.

[122] Zhengyao Jiang, Dixing Xu, and Jinjun Liang. *A Deep Reinforcement Learning Framework for the Financial Portfolio Management Problem*. 2017. DOI: 10.48550/ARXIV.1706.10059.

[123] M. I. Jordan and T. M. Mitchell. "Machine learning: Trends, perspectives, and prospects". In: *Science* 349.6245 (2015), pp. 255–260. DOI: 10.1126/science.aaa8415.

[124] Shirin Joshi, Sulabh Kumra, and Ferat Sahin. "Robotic Grasping using Deep Reinforcement Learning". In: *2020 IEEE 16th International Conference on Automation Science and Engineering (CASE)*. 2020, pp. 1461–1466. DOI: 10.1109/CASE48305.2020.9216986.

[125] Young-Kee Jung, Kyu-Won Lee, and Yo-Sung Ho. "Content-based event retrieval using semantic scene interpretation for automated traffic surveillance". In: *IEEE Transactions on Intelligent Transportation Systems* 2.3 (2001), pp. 151–163. DOI: 10.1109/6979.954548.

[126] L. P. Kaelbling, M. L. Littman, and A. W. Moore. "Reinforcement Learning: A Survey". In: *J. Artif. Intell. Res.* 4 (May 1996), pp. 237–285. ISSN: 1076-9757. DOI: 10.1613/jair.301.

[127] Gaetano Kanizsa. "Subjective contours". In: *Scientific American* 234.4 (1976), pp. 48–53. URL: https://www.jstor.org/stable/24950327.

[128] Hao-Cheng Kao, Kai-Fu Tang, and Edward Chang. "Context-Aware Symptom Checking for Disease Diagnosis Using Hierarchical Reinforcement Learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 32.1 (Apr. 2018). DOI: 10.1609/aaai.v32i1.11902.

[129] Davood Karimi and Septimiu E. Salcudean. "Reducing the Hausdorff Distance in Medical Image Segmentation With Convolutional Neural Networks". In: *IEEE Transactions on Medical Imaging* 39.2 (2020), pp. 499–513. DOI: 10.1109/TMI.2019.2930068.

[130] Guolin Ke et al. "LightGBM: A Highly Efficient Gradient Boosting Decision Tree". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/6449f44a102fde848669bdd9eb6b76fa-Paper.pdf.

[131] Seho Kee, Enrique del Castillo, and George Runger. "Query-by-committee improvement with diversity and density in batch active learning". In: *Information Sciences* 454-455 (2018), pp. 401–418. ISSN: 0020-0255. DOI: 10.1016/j.ins.2018.05.014.

[132] Daniel Keim, Huamin Qu, and Kwan-Liu Ma. "Big-Data Visualization". In: *IEEE Computer Graphics and Applications* 33.4 (2013), pp. 20–21. DOI: 10.1109/MCG.2013.54.

[133] Hoel Kervadec et al. "Boundary loss for highly unbalanced segmentation". In: *Medical Image Analysis* 67 (Jan. 2021), p. 101851. DOI: 10.1016/j.media.2020.101851.

[134] Khalil Khan, Massimo Mauro, and Riccardo Leonardi. "Multi-class semantic segmentation of faces". In: *2015 IEEE International Conference on Image Processing (ICIP)*. 2015, pp. 827–831. DOI: 10.1109/ICIP.2015.7350915.

[135] Muhammad Zubair Khan et al. "Deep Neural Architectures for Medical Image Semantic Segmentation: Review". In: *IEEE Access* 9 (2021), pp. 83002–83024. DOI: 10.1109/ACCESS.2021.3086530.

[136] Carl Kingsford and Steven L. Salzberg. "What are decision trees?" In: *Nature Biotechnology* 26.9 (Sept. 2008), pp. 1011–1013. ISSN: 1546-1696. DOI: 10.1038/nbt0908-1011.

[137] Tae-young Ko and Seung-ho Lee. "Novel Method of Semantic Segmentation Applicable to Augmented Reality". In: *Sensors* 20.6 (2020). ISSN: 1424-8220. DOI: 10.3390/s20061737.

[138] Deyvid Kochanov, Fatemeh Karimi Nejadasl, and Olaf Booij. *KPRNet: Improving projection-based LiDAR semantic segmentation*. 2020. DOI: 10.48550/ARXIV.2007.12668.

[139] Florian Kofler et al. *blob loss: instance imbalance aware loss functions for semantic segmentation*. 2022. DOI: 10.48550/ARXIV.2205.08209.

[140] Martin Kolarik, Radim Burget, and Kamil Riha. "Upsampling Algorithms for Autoencoder Segmentation Neural Networks: A Comparison Study". In: *2019 11th International Congress on Ultra Modern Telecommunications and Control Systems and Workshops (ICUMT)*. 2019, pp. 1–5. DOI: 10.1109/ICUMT48472.2019.8970918.

[141] Nikolaos Kouiroukidis and Georgios Evangelidis. "The Effects of Dimensionality Curse in High Dimensional kNN Search". In: *2011 15th Panhellenic Conference on Informatics*. 2011, pp. 41–45. DOI: 10.1109/PCI.2011.45.

[142] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. "CIFAR-10 (Canadian Institute for Advanced Research)". In: (). URL: http://www.cs.toronto.edu/~kriz/cifar.html.

[143] Guillaume Lample and Devendra Singh Chaplot. "Playing FPS Games with Deep Reinforcement Learning". In: *Proceedings of the AAAI Conference on Artificial Intelligence* 31.1 (Feb. 2017). DOI: 10.1609/aaai.v31i1.10827.

[144] Agostina J. Larrazabal, Cesar Martinez, and Enzo Ferrante. *Anatomical Priors for Image Segmentation via Post-Processing with Denoising Autoencoders*. 2019. DOI: 10.48550/ARXIV.1906.02343.

[145] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. "Deep learning". In: *Nature* 521.7553 (May 2015), pp. 436–444. ISSN: 1476-4687. DOI: 10.1038/nature14539.

[146] Johannes Lederer. "Activation Functions in Artificial Neural Networks: A Systematic Overview". In: *CoRR* abs/2101.09957 (2021). DOI: 10.48550/arXiv.2101.09957.

[147] Zhaoqi Leng et al. "PolyLoss: A Polynomial Expansion Perspective of Classification Loss Functions". In: *International Conference on Learning Representations*. 2022. URL: https://openreview.net/forum?id=gSdSJoenupI.

[148] Gaël Letarte et al. "Importance of Self-Attention for Sentiment Analysis". In: *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*. Brussels, Belgium: Association for Computational Linguistics, Nov. 2018, pp. 267–275. DOI: 10.18653/v1/W18-5429.

[149] Steff Lewis. "Regression analysis". In: *Practical neurology* 7.4 (2007), pp. 259–264. DOI: 10.1136/jnnp.2007.120055.

[150] Hao Li et al. "Visualizing the Loss Landscape of Neural Nets". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc., 2018. DOI: 10.48550/arXiv.1806.02375.

[151] Lisha Li et al. "Efficient Hyperparameter Optimization and Infinitely Many Armed Bandits". In: *CoRR* abs/1603.06560 (2016). DOI: 10.48550/arXiv.1603.06560.

[152] Lisha Li et al. "Hyperband: A Novel Bandit-Based Approach to Hyperparameter Optimization". In: (2016). DOI: 10.48550/ARXIV.1603.06560.

[153] Shuailin Li, Chuyu Zhang, and Xuming He. "Shape-aware Semi-supervised 3D Semantic Segmentation for Medical Images". In: *CoRR* abs/2007.10732 (2020). DOI: 10.1007/978-3-030-59710-8_54.

[154] Tian Li et al. "Tilted Empirical Risk Minimization". In: *CoRR* abs/2007.01162 (2020). DOI: 10.48550/arXiv.2007.01162.

[155] Yuxi Li. "Deep Reinforcement Learning: An Overview". In: *CoRR* abs/1701.07274 (2017). DOI: 10.48550/arXiv.1701.07274.

[156] *Lightning in 15 minutes — PyTorch Lightning 2.0.1 documentation*. [Online; accessed 6. Apr. 2023]. Mar. 2023. URL: https://lightning.ai/docs/pytorch/stable/starter/introduction.html.

[157] Tsung-Yi Lin et al. "Focal loss for dense object detection". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2980–2988. DOI: 10.48550/arXiv.1708.02002.

[158] Gang Liu and Jiabao Guo. "Bidirectional LSTM with attention mechanism and convolutional layer for text classification". In: *Neurocomputing* 337 (2019), pp. 325–338. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2019.01.078.

[159] Lucia Liu et al. "Robot Navigation in Crowded Environments Using Deep Reinforcement Learning". In: *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. 2020, pp. 5671–5677. DOI: 10.1109/IROS45743.2020.9341540.

[160] Xiao-Yang Liu et al. *FinRL: A Deep Reinforcement Learning Library for Automated Stock Trading in Quantitative Finance*. 2020. DOI: 10.48550/ARXIV.2011.09607.

[161] Ying Liu et al. "Deep Reinforcement Learning for Dynamic Treatment Regimes on Medical Registry Data". In: *2017 IEEE International Conference on Healthcare Informatics (ICHI)*. 2017, pp. 380–385. DOI: 10.1109/ICHI.2017.45.

[162] Jonathan Long, Evan Shelhamer, and Trevor Darrell. "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 3431–3440. DOI: 10.48550/arXiv.1411.4038.

[163] Linyuan Lü et al. "Recommender systems". In: *Physics Reports* 519.1 (2012). Recommender Systems, pp. 1–49. ISSN: 0370-1573. DOI: 10.1016/j.physrep.2012.02.006.

[164] Jun Ma. *Segmentation Loss Odyssey*. 2020. DOI: 10.48550/ARXIV.2005.13449.

[165] Rui Ma, Pin Tao, and Huiyun Tang. "Optimizing Data Augmentation for Semantic Segmentation on Small-Scale Dataset". In: *Proceedings of the 2nd International Conference on Control and Computer Vision*. ICCCV '19. Jeju, Republic of Korea: Association for Computing Machinery, 2019, pp. 77–81. ISBN: 9781450363228. DOI: 10.1145/3341016.3341020.

[166] Amin Madani et al. "Artificial Intelligence for Intraoperative Guidance: Using Semantic Segmentation to Identify Surgical Anatomy During Laparoscopic Cholecystectomy". In: *Annals of Surgery* Publish Ahead of Print (Nov. 2020). DOI: 10.1097/SLA.0000000000004594.

[167] Alireza Makhzani et al. *Adversarial Autoencoders*. 2015. DOI: 10.48550/ARXIV.1511.05644.

[168] Edward C. Malthouse and Robert C. Blattberg. "Can we predict customer lifetime value?" In: *Journal of Interactive Marketing* 19.1 (2005), pp. 2–16. DOI: https://doi.org/10.1002/dir.20027.

[169] Maja J. Matarić. "Reinforcement Learning in the Multi-Robot Domain". In: *Robot Colonies*. Ed. by Ronald C. Arkin and George A. Bekey. Boston, MA: Springer US, 1997, pp. 73–83. ISBN: 978-1-4757-6451-2. DOI: 10.1007/978-1-4757-6451-2_4.

[170] Dastan Maulud and Adnan M Abdulazeez. "A review on linear regression comprehensive in machine learning". In: *Journal of Applied Science and Technology Trends* 1.4 (2020), pp. 140–147. DOI: 10.38094/jastt1457.

[171] Christoph Mayer and Radu Timofte. "Adversarial sampling for active learning". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*. 2020, pp. 3071–3079. DOI: 10.48550/arXiv.1808.06671.

[172] Michael McCloskey and Neal J. Cohen. "Catastrophic Interference in Connectionist Networks: The Sequential Learning Problem". In: ed. by Gordon H. Bower. Vol. 24. Psychology of Learning and Motivation. Academic Press, 1989, pp. 109–165. DOI: 10.1016/S0079-7421(08)60536-8.

[173] John McCormac et al. "SceneNet RGB-D: Can 5M Synthetic Images Beat Generic ImageNet Pre-training on Indoor Segmentation?" In: (2017). DOI: 10.1109/ICCV.2017.292.

[174] Anthony McGregor et al. "Flow Clustering Using Machine Learning Techniques". In: *Passive and Active Network Measurement*. Ed. by Chadi Barakat and Ian Pratt. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004, pp. 205–214. ISBN: 978-3-540-24668-8. DOI: 10.1007/978-3-540-24668-8_21.

[175] Toshanlal Meenpal, Ashutosh Balakrishnan, and Amit Verma. "Facial Mask Detection using Semantic Segmentation". In: *2019 4th International Conference on Computing, Communications and Security (ICCCS)*. 2019, pp. 1–5. DOI: 10.1109/CCCS.2019.8888092.

[176] Ninareh Mehrabi et al. "A Survey on Bias and Fairness in Machine Learning". In: *ACM Comput. Surv.* 54.6 (July 2021). ISSN: 0360-0300. DOI: 10.1145/3457607.

[177] Bjoern H Menze et al. "The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)". en. In: *IEEE Trans Med Imaging* 34.10 (Dec. 2014), pp. 1993–2024. DOI: 10.1109/TMI.2014.2377694.

[178] Andres Milioto, Philipp Lottes, and Cyrill Stachniss. "Real-Time Semantic Segmentation of Crop and Weed for Precision Agriculture Robots Leveraging Background Knowledge in CNNs". In: *2018 IEEE International Conference on Robotics and Automation (ICRA)*. 2018, pp. 2229–2235. DOI: 10.1109/ICRA.2018.8460962.

[179] Andres Milioto and Cyrill Stachniss. "Bonnet: An Open-Source Training and Deployment Framework for Semantic Segmentation in Robotics using CNNs". In: *2019 International Conference on Robotics and Automation (ICRA)*. 2019, pp. 7094–7100. DOI: 10.1109/ICRA.2019.8793510.

[180] Lichao Mou, Yuansheng Hua, and Xiao Xiang Zhu. "Relation Matters: Relational Context-Aware Fully Convolutional Network for Semantic Segmentation of High-Resolution Aerial Images". In: *IEEE Transactions on Geoscience and Remote Sensing* 58.11 (2020), pp. 7557–7569. DOI: 10.1109/TGRS.2020.2979552.

[181] S. Naoum and I. K. Tsanis. "Orographic Precipitation Modeling with Multiple Linear Regression". In: *Journal of Hydrologic Engineering* 9.2 (2004), pp. 79–102. DOI: 10.1061/(ASCE)1084-0699(2004)9:2(79).

[182] Hong-Wei Ng et al. "Deep Learning for Emotion Recognition on Small Datasets Using Transfer Learning". In: *Proceedings of the 2015 ACM on International Conference on Multimodal Interaction*. ICMI '15. Seattle, Washington, USA: Association for Computing Machinery, 2015, pp. 443–449. ISBN: 9781450339124. DOI: 10.1145/2818346.2830593.

[183] Hai Nguyen and Hung La. "Review of Deep Reinforcement Learning for Robot Manipulation". In: *2019 Third IEEE International Conference on Robotic Computing (IRC)*. 2019, pp. 590–595. DOI: 10.1109/IRC.2019.00120.

[184] Ishan Nigam, Chen Huang, and Deva Ramanan. "Ensemble Knowledge Transfer for Semantic Segmentation". In: *2018 IEEE Winter Conference on Applications of Computer Vision (WACV)*. 2018, pp. 1499–1508. DOI: 10.1109/WACV.2018.00168.

[185] Shuteng Niu et al. "Distant Domain Transfer Learning for Medical Imaging". In: *IEEE Journal of Biomedical and Health Informatics* 25.10 (2021), pp. 3784–3793. DOI: 10.1109/JBHI.2021.3051470.

[186] Zhaoyang Niu, Guoqiang Zhong, and Hui Yu. "A review on the attention mechanism of deep learning". In: *Neurocomputing* 452 (2021), pp. 48–62. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2021.03.091.

[187] Curtis G. Northcutt, Anish Athalye, and Jonas Mueller. "Pervasive Label Errors in Test Sets Destabilize Machine Learning Benchmarks". In: (2021). DOI: 10.48550/ARXIV.2103.14749.

[188] Chigozie Nwankpa et al. "Activation Functions: Comparison of trends in Practice and Research for Deep Learning". In: *CoRR* abs/1811.03378 (2018). DOI: 10.48550/arXiv.1811.03378.

[189] Markus Oberweger, Paul Wohlhart, and Vincent Lepetit. "Hands Deep in Deep Learning for Hand Pose Estimation". In: *CoRR* abs/1502.06807 (2015). DOI: 10.48550/arXiv.1502.06807.

[190] *Object recognition by computer | Guide books*. Cambridge, MA, USA: MIT Press. ISBN: 978-026207130. URL: https://dl.acm.org/doi/abs/10.5555/102900.

[191] E. Osuna, R. Freund, and F. Girosit. "Training support vector machines: an application to face detection". In: *Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition*. 1997, pp. 130–136. DOI: 10.1109/CVPR.1997.609310.

[192] Lucas Paletta and Axel Pinz. "Active object recognition by view integration and reinforcement learning". In: *Robotics and Autonomous Systems* 31.1 (2000), pp. 71–86. ISSN: 0921-8890. DOI: 10.1016/S0921-8890(99)00079-2.

[193] Maria Papadomanolaki, Maria Vakalopoulou, and Konstantinos Karantzalos. "A Deep Multitask Learning Framework Coupling Semantic Segmentation and Fully Convolutional LSTM Networks for Urban Change Detection". In: *IEEE Transactions on Geoscience and Remote Sensing* 59.9 (2021), pp. 7651–7668. DOI: 10.1109/TGRS.2021.3055584.

[194] Rajul Parikh et al. "Understanding and using sensitivity, specificity and predictive values". en. In: *Indian J Ophthalmol* 56.1 (Jan. 2008), pp. 45–50. DOI: 10.4103/0301-4738.37595.

[195] Dinesh D Patil and Sonal G Deore. "Medical image segmentation: a review". In: *International Journal of Computer Science and Mobile Computing* 2.1 (2013), pp. 22–27. URL: https://www.semanticscholar.org/paper/Medical-Image-Segmentation%3A-A-Review-Patil-Deore/ae68948d2f0d2b4e9eb107f6af1d67066c0c6fdf.

[196] C. Perlich et al. "Machine learning for targeted display advertising: transfer learning in action". In: *Machine Learning* 95.1 (Apr. 2014), pp. 103–127. ISSN: 1573-0565. DOI: 10.1007/s10994-013-5375-2.

[197] Derek A. Pisner and David M. Schnyer. "Chapter 6 - Support vector machine". In: *Machine Learning*. Ed. by Andrea Mechelli and Sandra Vieira. Academic Press, 2020, pp. 101–121. ISBN: 978-0-12-815739-8. DOI: 10.1016/B978-0-12-815739-8.00006-7.

[198] Derek A. Pisner and David M. Schnyer. "Chapter 6 - Support vector machine". In: *Machine Learning*. Ed. by Andrea Mechelli and Sandra Vieira. Academic Press, 2020, pp. 101–121. ISBN: 978-0-12-815739-8. DOI: 10.1016/B978-0-12-815739-8.00006-7.

[199] Prasanna Porwal et al. *Indian Diabetic Retinopathy Image Dataset (IDRiD)*. 2018. DOI: 10.21227/H25W98.

[200] David M. W. Powers. *Evaluation: from precision, recall and F-measure to ROC, informedness, markedness and correlation*. 2020. DOI: 10.48550/arXiv.2010.16061.

[201] Sivaramakrishnan Rajaraman et al. "A Systematic Evaluation of Ensemble Learning Methods for Fine-Grained Semantic Segmentation of Tuberculosis-Consistent Lesions in Chest Radiographs". en. In: *Bioengineering (Basel)* 9.9 (Aug. 2022). DOI: 10.3390/bioengineering9090413.

[202] WP Ramadhan, STMT Astri Novianty, and STMT Casi Setianingsih. "Sentiment analysis using multinomial logistic regression". In: *2017 International Conference on Control, Electronics, Renewable Energy and Communications (ICCREC)*. IEEE. 2017, pp. 46–49. DOI: 10.1109/ICCEREC.2017.8226700.

[203] Christopher J. Rapson et al. "Reducing the Pain: A Novel Tool for Efficient Ground-Truth Labelling in Images". In: *2018 International Conference on Image and Vision Computing New Zealand (IVCNZ)*. 2018, pp. 1–9. DOI: 10.1109/IVCNZ.2018.8634750.

[204] Michiel van der Ree and Marco Wiering. "Reinforcement learning in the game of Othello: Learning against a fixed opponent and learning from self-play". In: *2013 IEEE Symposium on Adaptive Dynamic Programming and Reinforcement Learning (ADPRL)*. 2013, pp. 108–115. DOI: 10.1109/ADPRL.2013.6614996.

[205] Shaoqing Ren et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *Advances in Neural Information Processing Systems*. Ed. by C. Cortes et al. Vol. 28. Curran Associates, Inc., 2015. URL: https://proceedings.neurips.cc/paper/2015/file/14bfa6bb14875e45bba028a21ed38046-Paper.pdf.

[206] Mina Rezaei et al. "A Conditional Adversarial Network for Semantic Segmentation of Brain Tumor". In: *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*. Ed. by Alessandro Crimi et al. Cham: Springer International Publishing, 2018, pp. 241–252. ISBN: 978-3-319-75238-9. DOI: 10.48550/arXiv.1708.05227.

[207] Florian Richter, Ryan K. Orosco, and Michael C. Yip. "Open-Sourced Reinforcement Learning Environments for Surgical Robotics". In: *CoRR* abs/1903.02090 (2019). DOI: 10.48550/arXiv.1903.02090.

[208] R Tyrrell Rockafellar and Roger J-B Wets. *Variational analysis*. Vol. 317. Springer Science & Business Media, 2009. URL: https://link.springer.com/book/10.1007/978-3-642-02431-3.

[209] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. "U-Net: Convolutional Networks for Biomedical Image Segmentation". In: *CoRR* abs/1505.04597 (2015). DOI: 10.48550/arXiv.1505.04597.

[210] Frank Rosenblatt. "The perceptron: a probabilistic model for information storage and organization in the brain." In: *Psychological review* 65.6 (1958), p. 386. DOI: 10.1037/h0042519.

[211] Holger R. Roth et al. "Going to Extremes: Weakly Supervised Medical Image Segmentation". In: *Machine Learning and Knowledge Extraction* 3.2 (2021), pp. 507–524. ISSN: 2504-4990. DOI: 10.3390/make3020026.

[212] Sebastian Ruder. "An overview of gradient descent optimization algorithms". In: *CoRR* abs/1609.04747 (2016). DOI: 10.48550/arXiv.1609.04747.

[213] Stuart Russell and Peter Norvig. *Artificial Intelligence: A Modern Approach*. 3rd ed. Prentice Hall, 2010. URL: https://aima.cs.berkeley.edu.

[214] Christoph Sager, Christian Janiesch, and Patrick Zschech. "A survey of image labelling for computer vision applications". In: *Journal of Business Analytics* 4.2 (2021), pp. 91–110. DOI: 10.1080/2573234X.2021.1908861.

[215] Nimrod Sagie, Hayit Greenspan, and Jacob Goldberger. "Transfer Learning via Parameter Regularization for Medical Image Segmentation". In: *2021 29th European Signal Processing Conference (EUSIPCO)*. 2021, pp. 985–989. DOI: 10.23919/EUSIPCO54536.2021.9616331.

[216] F. Sahba, H.R. Tizhoosh, and M.M.A. Salama. "A Reinforcement Learning Framework for Medical Image Segmentation". In: *The 2006 IEEE International Joint Conference on Neural Network Proceedings*. 2006, pp. 511–517. DOI: 10.1109/IJCNN.2006.246725.

[217] Y. Sahin and E. Duman. "Detecting credit card fraud by ANN and logistic regression". In: *2011 International Symposium on Innovations in Intelligent Systems and Applications*. 2011, pp. 315–319. DOI: 10.1109/INISTA.2011.5946108.

[218] Seyed Sadegh Mohseni Salehi, Deniz Erdogmus, and Ali Gholipour. "Tversky loss function for image segmentation using 3D fully convolutional deep networks". In: *CoRR* abs/1706.05721 (2017). DOI: 10.48550/arXiv.1706.05721.

[219] Lars Schmarje et al. "A Survey on Semi-, Self- and Unsupervised Learning for Image Classification". In: *IEEE Access* 9 (2021), pp. 82146–82168. DOI: 10.1109/ACCESS.2021.3084358.

[220] John Schulman et al. "Trust Region Policy Optimization". In: *Proceedings of the 32nd International Conference on Machine Learning*. Ed. by Francis Bach and David Blei. Vol. 37. Proceedings of Machine Learning Research. Lille, France: PMLR, July 2015, pp. 1889–1897. URL: https://proceedings.mlr.press/v37/schulman15.html.

[221] John Schulman et al. "Proximal Policy Optimization Algorithms". In: *CoRR* abs/1707.06347 (2017). DOI: 10.48550/arXiv.1707.06347.

[222] Mark Schutera et al. "Methods for the frugal labeler: Multi-class semantic segmentation on heterogeneous labels". In: *PLOS ONE* 17.2 (Feb. 2022), pp. 1–14. DOI: 10.1371/journal.pone.0263656.

[223] Burr Settles. "Active Learning Literature Survey". In: *University of Wisconsin-Madison Department of Computer Sciences* (2009). URL: https://minds.wisconsin.edu/handle/1793/60660.

[224] Subarna Shakya and Smys Smys. "Big data analytics for improved risk management and customer segregation in banking applications". In: *Journal of ISMAC* 3.3 (2021), pp. 235–249. DOI: 10.36548/jismac.2021.3.005.

[225] Kun Shao et al. "A Survey of Deep Reinforcement Learning in Video Games". In: *CoRR* abs/1912.10944 (2019). DOI: 10.48550/arXiv.1912.10944.

[226] Linda G Shapiro, George C Stockman, et al. *Computer vision*. Vol. 3. Prentice Hall New Jersey, 2001. URL: https://books.google.de/books/about/Computer_Vision.html?id=INYCnwEACAAJ&redir_esc=y.

[227] Manali Sharma and Mustafa Bilgic. "Evidence-based uncertainty sampling for active learning". In: *Data Mining and Knowledge Discovery* 31.1 (Jan. 2017), pp. 164–202. ISSN: 1573-756X. DOI: 10.1007/s10618-016-0460-3.

[228] Siddharth Sharma, Simone Sharma, and Anidhya Athaiya. "ACTIVATION FUNCTIONS IN NEURAL NETWORKS". In: *International Journal of Engineering Applied Sciences and Technology* 04 (May 2020), pp. 310–316. DOI: 10.33564/IJEAST.2020.v04i12.054.

[229] Miaogen Shen et al. "Precipitation impacts on vegetation spring phenology on the Tibetan Plateau". In: *Global Change Biology* 21.10 (2015), pp. 3647–3656. DOI: 10.1111/gcb.12961.

[230] Hengcan Shi et al. *ProposalCLIP: Unsupervised Open-Category Object Proposal Generation via Exploiting CLIP Cues*. 2022. DOI: 10.48550/ARXIV.2201.06696.

[231] Connor Shorten and Taghi M. Khoshgoftaar. "A survey on Image Data Augmentation for Deep Learning". In: *Journal of Big Data* 6.1 (July 2019), p. 60. ISSN: 2196-1115. DOI: 10.1186/s40537-019-0197-0.

[232] GT Shrivakshan and Chandramouli Chandrasekar. "A comparison of various edge detection techniques used in image processing". In: *International Journal of Computer Science Issues (IJCSI)* 9.5 (2012), p. 269. URL: https://www.researchgate.net/publication/303142762_A_Comparison_of_various_Edge_Detection_Techniques_used_in_Image_Processing.

[233] Saman Siadati. "What is unsupervised Learning". In: (Aug. 2018). DOI: 10.13140/RG.2.2.33325.10720.

[234] Mennatullah Siam et al. "A comparative study of real-time semantic segmentation for autonomous driving". In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*. 2018, pp. 587–597. DOI: 10.1109/CVPRW.2018.00101.

[235] Yawar Siddiqui, Julien Valentin, and Matthias Nießner. *ViewAL: Active Learning with Viewpoint Entropy for Semantic Segmentation*. 2019. DOI: 10.48550/ARXIV.1911.11789.

[236] David Silver. "Reinforcement Learning and Simulation-Based Search in Computer Go". In: *ERA* (2009). DOI: 10.7939/R39D8T.

[237] David Silver et al. "A general reinforcement learning algorithm that masters chess, shogi, and Go through self-play". In: *Science* 362.6419 (2018), pp. 1140–1144. DOI: 10.1126/science.aar6404.

[238] Ashish Sinha and Jose Dolz. "Multi-Scale Self-Guided Attention for Medical Image Segmentation". In: *IEEE Journal of Biomedical and Health Informatics* 25.1 (2021), pp. 121–130. DOI: 10.1109/JBHI.2020.2986926.

[239] Shuran Song, Samuel P Lichtenberg, and Jianxiong Xiao. "Sun rgb-d: A rgb-d scene understanding benchmark suite". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2015, pp. 567–576. URL: https://rgbd.cs.princeton.edu.

[240] C. O. S. Sorzano, J. Vargas, and A. Pascual Montano. *A survey of dimensionality reduction techniques*. 2014. DOI: 10.48550/ARXIV.1403.2877.

[241] Ireneous N. Soyiri and Daniel D. Reidpath. "An overview of health forecasting". In: *Environmental Health and Preventive Medicine* 18.1 (Jan. 2013), pp. 1–9. ISSN: 1347-4715. DOI: 10.1007/s12199-012-0294-6.

[242] Sinisa Stekovic, Friedrich Fraundorfer, and Vincent Lepetit. *Casting Geometric Constraints in Semantic Segmentation as Semi-Supervised Learning*. 2019. DOI: 10.48550/ARXIV.1904.12534.

[243] Alan W. Stitt et al. "The progress in understanding and treatment of diabetic retinopathy". In: *Progress in Retinal and Eye Research* 51 (2016), pp. 156–186. ISSN: 1350-9462. DOI: 10.1016/j.preteyeres.2015.08.001.

[244] Lars St»hle and Svante Wold. "Analysis of variance (ANOVA)". In: *Chemometrics and Intelligent Laboratory Systems* 6.4 (1989), pp. 259–272. ISSN: 0169-7439. DOI: https://doi.org/10.1016/0169-7439(89)80095-4.

[245] Xiaogang Su, Xin Yan, and Chih-Ling Tsai. "Linear regression". In: *Wiley Interdisciplinary Reviews: Computational Statistics* 4.3 (2012), pp. 275–294. DOI: 10.1002/wics.1198.

[246] Shashank Subramanian, Amir Gholami, and George Biros. "Simulation of glioblastoma growth using a 3D multispecies tumor model with mass effect". In: *Journal of Mathematical Biology* 79.3 (Aug. 2019), pp. 941–967. ISSN: 1432-1416. DOI: 10.1007/s00285-019-01383-y.

[247] Carole H. Sudre et al. "Generalised Dice Overlap as a Deep Learning Loss Function for Highly Unbalanced Segmentations". In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Springer International Publishing, 2017, pp. 240–248. DOI: 10.1007/978-3-319-67558-9_28.

[248] Attila Szabó, Hadi Jamali Rad, and Siva-Datta Mannava. "Tilted Cross Entropy (TCE): Promoting Fairness in Semantic Segmentation". In: *CoRR* abs/2103.14051 (2021). DOI: 10.48550/arXiv.2103.14051.

[249] Richard Szeliski. *Computer vision: algorithms and applications*. Springer Nature, 2022.

[250] Leonardo Tanzi et al. "Real-time deep learning semantic segmentation during intra-operative surgery for 3D augmented reality assistance". In: *International Journal of Computer Assisted Radiology and Surgery* 16.9 (Sept. 2021), pp. 1435–1445. ISSN: 1861-6429. DOI: 10.1007/s11548-021-02432-y.

[251] Luke Taylor and Geoff Nitschke. "Improving Deep Learning with Generic Data Augmentation". In: *2018 IEEE Symposium Series on Computational Intelligence (SSCI)*. 2018, pp. 1542–1547. DOI: 10.1109/SSCI.2018.8628742.

[252] Sebastian Taylor. *Decision Tree*. [Online; accessed 1. Mar. 2023]. Nov. 2022. URL: https://corporatefinanceinstitute.com/resources/data-science/decision-tree.

[253] Firas H. Thanoon. "Robust Regression by Least Absolute Deviations Method". In: *International Journal of Statistics and Applications* 5.3 (2015), pp. 109–112. ISSN: 2168-5215. DOI: 10.5923/j.statistics.20150503.02.

[254] Nugzar Todua, Petre Babilua, and Teona Dochviri. "On the multiple linear regression in marketing research". In: *Bull. Georg. Natl. Acad. Sci* 7.3 (2013), pp. 135–139. URL: https://www.researchgate.net/publication/260361577_On_the_Multiple_Linear_Regression_in_Marketing_Research.

[255] Tatiana Tommasi et al. "A Deeper Look at Dataset Bias". In: *Domain Adaptation in Computer Vision Applications*. Ed. by Gabriela Csurka. Cham: Springer International Publishing, 2017, pp. 37–55. ISBN: 978-3-319-58347-1. DOI: 10.1007/978-3-319-58347-1_2.

[256] S. TONG. "Support vector machine active learning with applications to text classification". In: *Proc. Seventeenth International Conference on Machine Learning, 2000* (2000). URL: https://cir.nii.ac.jp/crid/1574231874012815360.

[257] GODFRIED TOUSSAINT. "GEOMETRIC PROXIMITY GRAPHS FOR IMPROVING NEAREST NEIGHBOR METHODS IN INSTANCE-BASED LEARNING AND DATA MINING". In: *International Journal of Computational Geometry & Applications* 15.02 (2005), pp. 101–150. DOI: 10.1142/S0218195905001622.

[258] Michael Treml et al. "Speeding up Semantic Segmentation for Autonomous Driving". In: *OpenReview* (Oct. 2016). URL: https://openreview.net/forum?id=S1uHiFyyg.

[259] Christine L. Tsien et al. "Using Classification Tree and Logistic Regression Methods to Diagnose Myocardial Infarction". In: *MEDINFO '98*. IOS Press, 1998, pp. 493–497. DOI: 10.3233/978-1-60750-896-0-493.

[260] Volodymyr Turchenko, Eric Chalmers, and Artur Luczak. "A Deep Convolutional Auto-Encoder with Pooling - Unpooling Layers in Caffe". In: *CoRR* abs/1701.04949 (2017). DOI: 10.48550/arXiv.1701.04949.

[261] Amos Tversky. "Features of similarity." In: *Psychological review* 84.4 (1977), p. 327. DOI: 10.1037/0033-295X.84.4.327.

[262] Eric Tzeng et al. *Deep Domain Confusion: Maximizing for Domain Invariance*. 2014. DOI: 10.48550/ARXIV.1412.3474.

[263] Shahadat Uddin et al. "Comparative performance analysis of K-nearest neighbour (KNN) algorithm and its different variants for disease prediction". In: *Scientific Reports* 12.1 (Apr. 2022), p. 6256. ISSN: 2045-2322. DOI: 10.1038/s41598-022-10358-x.

[264] Irem Ulku and Erdem Akagündüz. "A Survey on Deep Learning-based Architectures for Semantic Segmentation on 2D Images". In: *Applied Artificial Intelligence* 36.1 (2022), p. 2032924. DOI: 10.1080/08839514.2022.2032924.

[265] A. Vamseekrishna et al. "Prediction of Temperature and Humidity Using IoT and Machine Learning Algorithm". In: *International Conference on Intelligent and Smart Computing in Data Analytics*. Ed. by Siddhartha Bhattacharyya et al. Singapore: Springer Singapore, 2021, pp. 271–279. ISBN: 978-981-33-6176-8. DOI: 10.1007/978-981-33-6176-8_30.

[266] D. Van De Ville et al. "Noise reduction by fuzzy image filtering". In: *IEEE Transactions on Fuzzy Systems* 11.4 (2003), pp. 429–436. DOI: 10.1109/TFUZZ.2003.814830.

[267] Adam Van Etten, Dave Lindenbaum, and Todd M. Bacastow. *SpaceNet: A Remote Sensing Dataset and Challenge Series*. 2018. DOI: 10.48550/ARXIV.1807.01232.

[268] V. Vapnik. "Principles of Risk Minimization for Learning Theory". In: *Advances in Neural Information Processing Systems*. Ed. by J. Moody, S. Hanson, and R.P. Lippmann. Vol. 4. Morgan-Kaufmann, 1991. URL: https://proceedings.neurips.cc/paper/1991/file/ff4d5fbbafdf976cfdc032e3bde78de5-Paper.pdf.

[269] Ashish Vaswani et al. "Attention is All you Need". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. DOI: 10.48550/arXiv.1706.03762.

[270] Jorge R. Vergara and Pablo A. Estévez. "A review of feature selection methods based on mutual information". In: *Neural Computing and Applications* 24.1 (Jan. 2014), pp. 175–186. ISSN: 1433-3058. DOI: 10.1007/s00521-013-1368-0.

[271] Vijay Verma and Rajesh Kumar Aggarwal. "A comparative analysis of similarity measures akin to the Jaccard index in collaborative recommendations: empirical and theoretical perspective". In: *Social Network Analysis and Mining* 10.1 (June 2020), p. 43. ISSN: 1869-5469. DOI: 10.1007/s13278-020-00660-9.

[272] *Visual perception is complicated because our cognitive systems are not adept at analyzing visual stimuli.* [Online; accessed 7. Mar. 2023]. Mar. 2023. URL: https://brainmass.com/psychology/perception/what-is-a-visual-ambiguity-223578.

[273] Joshua T. Vogelstein et al. "Discovery of Brainwide Neural-Behavioral Maps via Multiscale Unsupervised Structure Learning". In: *Science* 344.6182 (2014), pp. 386–392. DOI: 10.1126/science.1250298.

[274] Kentaro Wada, Kei Okada, and Masayuki Inaba. "Joint Learning of Instance and Semantic Segmentation for Robotic Pick-and-Place with Heavy Occlusions in Clutter". In: *2019 International Conference on Robotics and Automation (ICRA)*. 2019, pp. 9558–9564. DOI: 10.1109/ICRA.2019.8793783.

[275] Ji Wan et al. "Deep Learning for Content-Based Image Retrieval: A Comprehensive Study". In: *Proceedings of the 22nd ACM International Conference on Multimedia*. MM '14. Orlando, Florida, USA: Association for Computing Machinery, 2014, pp. 157–166. ISBN: 9781450330633. DOI: 10.1145/2647868.2654948.

[276] Jia Wan, Nikil Senthil Kumar, and Antoni B. Chan. "Fine-Grained Crowd Counting". In: *IEEE Transactions on Image Processing* 30 (2021), pp. 2114–2126. DOI: 10.1109/TIP.2021.3049938.

[277] Gaoang Wang et al. "Uncertainty sampling based active learning with diversity constraint by sparse selection". In: *2017 IEEE 19th International Workshop on Multimedia Signal Processing (MMSP)*. 2017, pp. 1–6. DOI: 10.1109/MMSP.2017.8122269.

[278] Luyang Wang et al. "Skip-Connection Convolutional Neural Network for Still Image Crowd Counting". In: *Applied Intelligence* 48.10 (Oct. 2018), pp. 3360–3371. ISSN: 0924-669X. DOI: 10.1007/s10489-018-1150-1.

[279] Mei Wang and Weihong Deng. "Deep visual domain adaptation: A survey". In: *Neurocomputing* 312 (2018), pp. 135–153. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2018.05.083.

[280] Pei Wang and Albert C.S. Chung. "Relax and focus on brain tumor segmentation". In: *Medical Image Analysis* 75 (2022), p. 102259. ISSN: 1361-8415. DOI: 10.1016/j.media.2021.102259.

[281] Rui Wang, Dhruv Mahajan, and Vignesh Ramanathan. *What leads to generalization of object proposals?* 2020. DOI: 10.48550/ARXIV.2008.05700.

[282] Shanshan Wang et al. "Deep learning for fast MR imaging: A review for learning reconstruction from incomplete k-space data". In: *Biomedical Signal Processing and Control* 68 (2021), p. 102579. ISSN: 1746-8094. DOI: 10.1016/j.bspc.2021.102579.

[283] Zifeng Wang et al. "Deep semantic segmentation for visual understanding on construction sites". In: *Computer-Aided Civil and Infrastructure Engineering* 37.2 (2022), pp. 145–162. DOI: 10.1111/mice.12701.

[284] Ziyu Wang et al. "Dueling Network Architectures for Deep Reinforcement Learning". In: *Proceedings of The 33rd International Conference on Machine Learning*. Ed. by Maria Florina Balcan and Kilian Q. Weinberger. Vol. 48. Proceedings of Machine Learning Research. New York, New York, USA: PMLR, June 2016, pp. 1995–2003. URL: https://proceedings.mlr.press/v48/wangf16.html.

[285] Gordon Wetzstein et al. "Tensor displays: compressive light field synthesis using multilayer displays with directional backlighting". In: (2012). DOI: 10.1145/2185520.2335431.

[286] Martin J. Willemink et al. "Preparing Medical Imaging Data for Machine Learning". In: *Radiology* 295.1 (2020). PMID: 32068507, pp. 4–15. DOI: 10.1148/radiol.2020192224.

[287] David H. Wolpert. "Stacked generalization". In: *Neural Networks* 5.2 (1992), pp. 241–259. ISSN: 0893-6080. DOI: 10.1016/S0893-6080(05)80023-1.

[288] Jia Wu et al. "Hyperparameter Optimization for Machine Learning Models Based on Bayesian Optimizationb". In: *Journal of Electronic Science and Technology* 17.1 (2019), pp. 26–40. ISSN: 1674-862X. DOI: 10.11989/JEST.1674-862X.80904120.

[289] Yanli Wu et al. "Application of alternating decision tree with AdaBoost and bagging ensembles for landslide susceptibility mapping". In: *CATENA* 187 (2020), p. 104396. ISSN: 0341-8162. DOI: 10.1016/j.catena.2019.104396.

[290] Yi Wu et al. "Sampling Strategies for Active Learning in Personal Photo Retrieval". In: *2006 IEEE International Conference on Multimedia and Expo*. 2006, pp. 529–532. DOI: 10.1109/ICME.2006.262442.

[291] Radhika Yalavarthi and M. Shashi. "Atmospheric Temperature Prediction using Support Vector Machines". In: *International Journal of Computer Theory and Engineering* 1 (Jan. 2009), pp. 55–58. DOI: 10.7763/IJCTE.2009.V1.9.

[292] Michael Yeung et al. "Unified Focal loss: Generalising Dice and cross entropy-based losses to handle class imbalanced medical image segmentation". In: *Computerized Medical Imaging and Graphics* 95 (2022), p. 102026. ISSN: 0895-6111. DOI: 10.1016/j.compmedimag.2021.102026.

[293] Xin Yi, Ekta Walia, and Paul Babyn. "Generative adversarial network in medical imaging: A review". In: *Medical Image Analysis* 58 (2019), p. 101552. ISSN: 1361-8415. DOI: 10.1016/j.media.2019.101552.

[294] Youngjin Yoon et al. "Learning a deep convolutional network for light-field image super-resolution". In: *Proceedings of the IEEE international conference on computer vision workshops*. 2015, pp. 24–32. DOI: 10.1109/ICCVW.2015.17.

[295] Hongshan Yu et al. "Methods and datasets on semantic segmentation: A review". In: *Neurocomputing* 304 (2018), pp. 82–103. ISSN: 0925-2312. DOI: 10.1016/j.neucom.2018.03.037.

[296] Litao Yu et al. "Distribution-Aware Margin Calibration for Semantic Segmentation in Images". In: *International Journal of Computer Vision* 130.1 (Jan. 2022), pp. 95–110. ISSN: 1573-1405. DOI: 10.1007/s11263-021-01533-0.

[297] Sihyun Yu et al. *Generating Videos with Dynamics-aware Implicit Generative Adversarial Networks*. 2022. DOI: 10.48550/ARXIV.2202.10571.

[298] Jianlong Yuan et al. *A Simple Baseline for Semi-supervised Semantic Segmentation with Strong Data Augmentation*. 2021. DOI: 10.48550/ARXIV.2104.07256.

[299] Sangdoo Yun et al. "Action-decision networks for visual tracking with deep reinforcement learning". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 2711–2720. DOI: 10.1109/CVPR.2017.148.

[300] Matthew D. Zeiler. "ADADELTA: An Adaptive Learning Rate Method". In: *CoRR* abs/1212.5701 (2012). DOI: 10.48550/arXiv.1212.5701.

[301] Hang Zhang et al. *Context Encoding for Semantic Segmentation*. 2018. DOI: 10.48550/ARXIV.1803.08904.

[302] Hongyi Zhang et al. "mixup: Beyond Empirical Risk Minimization". In: *CoRR* abs/1710.09412 (2017). DOI: 10.48550/arXiv.1710.09412.

[303] Huanle Zhang et al. "Slimmer: Accelerating 3D Semantic Segmentation for Mobile Augmented Reality". In: *2020 IEEE 17th International Conference on Mobile Ad Hoc and Sensor Systems (MASS)*. 2020, pp. 603–612. DOI: 10.1109/MASS50613.2020.00079.

[304] Jing Zhang et al. "Multi-Scale Context Aggregation for Semantic Segmentation of Remote Sensing Images". In: *Remote Sensing* 12.4 (2020). ISSN: 2072-4292. DOI: 10.3390/rs12040701.

[305] Ying Zhang and Chen Ling. "A strategy to apply machine learning to small datasets in materials science". In: *npj Computational Materials* 4.1 (May 2018), p. 25. ISSN: 2057-3960. DOI: 10.1038/s41524-018-0081-z.

[306] Zhongheng Zhang. "Introduction to machine learning: k-nearest neighbors". en. In: *Ann Transl Med* 4.11 (June 2016), p. 218. DOI: 10.21037/atm.2016.03.37.

[307] Qi Zhao et al. "Semantic Segmentation With Attention Mechanism for Remote Sensing Images". In: *IEEE Transactions on Geoscience and Remote Sensing* 60 (2022), pp. 1–13. DOI: 10.1109/TGRS.2021.3085889.

[308] Rongjian Zhao et al. "Rethinking Dice Loss for Medical Image Segmentation". In: *2020 IEEE International Conference on Data Mining (ICDM)*. 2020, pp. 851–860. DOI: 10.1109/ICDM50108.2020.00094.

[309] Zhong-Qiu Zhao et al. "Object Detection with Deep Learning: A Review". In: *CoRR* abs/1807.05511 (2018). DOI: 10.48550/arXiv.1807.05511.

[310] Zhong-Qiu Zhao et al. "Object Detection With Deep Learning: A Review". In: *IEEE Transactions on Neural Networks and Learning Systems* 30.11 (2019), pp. 3212–3232. DOI: 10.1109/TNNLS.2018.2876865.

[311] Bolei Zhou et al. "Learning deep features for discriminative localization". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2016, pp. 2921–2929. DOI: 10.48550/arXiv.1512.04150.

[312] Bolei Zhou et al. "Scene parsing through ade20k dataset". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 633–641. DOI: 10.1109/CVPR.2017.544.

[313] Wei Zhou et al. "Automated Evaluation of Semantic Segmentation Robustness for Autonomous Driving". In: *IEEE Transactions on Intelligent Transportation Systems* 21.5 (2020), pp. 1951–1963. DOI: 10.1109/TITS.2019.2909066.

[314] Yizhou Zhou et al. "Context-Reinforced Semantic Segmentation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. June 2019. DOI: 10.1109/CVPR.2019.00417.

[315] Zhenpeng Zhou et al. "Optimization of Molecules via Deep Reinforcement Learning". In: *CoRR* abs/1810.08678 (2018). DOI: 10.1038/s41598-019-47148-x.

[316] Zhi-Hua Zhou. "A brief introduction to weakly supervised learning". In: *National Science Review* 5.1 (Aug. 2017), pp. 44–53. ISSN: 2095-5138. DOI: 10.1093/nsr/nwx106.

[317] Fuzhen Zhuang et al. "A Comprehensive Survey on Transfer Learning". In: *CoRR* abs/1911.02685 (2019). DOI: 10.48550/arXiv.1911.02685.

[318] James Zou et al. "A primer on deep learning in genomics". In: *Nature Genetics* 51.1 (Jan. 2019), pp. 12–18. ISSN: 1546-1718. DOI: 10.1038/s41588-018-0295-5.

[319] Shaofeng Zou, Tengyu Xu, and Yingbin Liang. "Finite-Sample Analysis for SARSA with Linear Function Approximation". In: *Advances in Neural Information Processing Systems*. Ed. by H. Wallach et al. Vol. 32. Curran Associates, Inc., 2019. DOI: 10.48550/arXiv.1902.02234.