Comparative Study of Weakly-Supervised Segmentation Methods Using Image-Level Classification Labels for Blood Accumulation Segmentation.

Master Thesis

Garvit Joshi

Division Translational Surgical Oncology
NCT Dresden

Reviewer:	Prof. Dr.-Ing. Stefanie Speidel
Second reviewer:	Dr. Sebastian Bodenstedt
Advisor:	Rivoir Dominik

Duration: June 14, 2021 – February 21, 2022

NCT | NATIONAL CENTER FOR TUMOR DISEASES PARTNER SITE DRESDEN UNIVERSITY CANCER CENTER UCC

TECHNISCHE UNIVERSITÄT DRESDEN

I hereby declare that I have developed and written the enclosed thesis completely by myself, and have not used sources or means without declaration in the text.

Dresden, 21st February , 2022

Garvit Joshi

Digitally signed by Garvit Joshi
Date: 2022.02.21 02:41:01 +01'00'

Garvit Joshi
Dresden 21.02.2022

Abstract

In this work, we discuss the segmentation of blood accumulations in surgical images. There are scenarios where blood can get accumulated in the body during surgery. The idea is to identify these accumulations to stop bleeding or remove the blood. The traditional method of pixel-level labelling is time-consuming and potentially ambiguous. Hence, Weakly-supervised learning using image-level labels is an exciting method to study as this allows us to collect and label data quickly, allowing for faster construction of models.
We evaluate various neural networks on how well they are suited for our use-case scenario and argue over the impact of using Hard negatives (images that contains regions where there is some blood, but there is no accumulation) during training to boost the model's performance. There are five methods selected for evaluation in this work. Three methods are evaluated for the Weakly-supervised approach (Grad-CAM, SEAM, Puzzle-CAM). These Three methods utilise Class Activation Maps to generate segmentation masks. Two approaches are used as upper and lower for the performance of Weakly- supervised methods. For upper bound, a Fully-supervised approach is chosen, namely U-Net. For the lower bound, a non-learning based approach is chosen.

We perform qualitative and quantitative analysis and observe that Grad-CAM generates the most satisfactory results compared to other Weakly supervised models. However, the results are not as good compared to the Fully supervised method but are still producing results that can be used for the aforementioned use-case scenario. We also empirically determine that to threshold the heatmaps produced by Weakly-supervised models Otsu's thresholding method is the best option among various thresholding techniques [1]. The inclusion of Hard negative is advantageous for Weakly-supervised methods but reduces the performance of the Fully-supervised method. We do Four-fold cross-validation and observe that models perform better on folds 1 and 2 compared to 3 and 4. Puzzle-CAM does not generate satisfactory results even though it is a state-of-the-art method as of Feb 21st, 2022. SEAM performs satisfactory but warrants further improvements to suit the application more. For future studies, we can use Guided Backpropagation in Grad-CAM and also look at alternative methods that utilise saliency information or uses graph-convolutional neural networks.

Contents

1. Introduction

1.1 Motivation

In surgical image segmentation, Neural Networks [2] are useful tools and have remarkable performance. The main problems that plague these Neural Networks are data availability, evaluation criteria and the availability of labels. The training data is often not available with proper labels. Manual generation of pixel-wise annotations takes a lot of time and resources and often needs an expert knowledge. Human bias always influences this data, and thus image-level labels can overcome these issues partially. However they present a challenge as they do not provide any information about the location of the object and its shape [3].

The Thesis aims to evaluate various methods to perform Weakly-supervised semantic segmentation and determine whether or not the object of interest is present in the image (can also be referred as ROI) using these image-level labels. We do this to localise the blood accumulation in a given image. The information generated from this can be used to assist the surgeon by alerting them about blood accumulation. Accumulation of blood is a significant problem during endoscopic surgeries, and can obscure vital regions. The above problems can culminate into something serious if not resolved in a timely fashion. Thus, assistance from the system can be vital for assisting the surgeon.

Since Weak-supervision using image-level labels does not contain any information about the exact boundaries of the region/object of interest, it faces problems finding the exact boundaries of the region/object.

However, the blood can be removed with the help of the suction tool as we do not need the boundaries of our predicted mask to be precise. Using the tool anywhere on the accumulation site will remove the blood. This reduced complexity of the task allows us to use methods which are not accurate in finding the exact boundaries of the blood accumulation region but can identify and localise the region.

An example of this real-world scenario can be laparoscopic surgery, during the surgery the surgeons might require assistance to monitor the site of active bleeding and achieve haemostasis. When a surgeon tries to achieve primary haemostasis, they might utilise a gauze or suction tool to remove blood, but this may partially affect the surgeon's vision and then it becomes harder to achieve secondary haemostasis. In this scenario, the results generated from the model can assist the surgeon in localising the site of bleeding and give vital time for surgeons to react.

1.2 Goals

The end goal of this work is to generate masks to visualise the predicted regions with blood accumulation. These masks in a real-world scenario can be projected virtually on display used by the surgeon as a superimposed image or on a Virtual Reality headset to assist the surgeons. These masks can also allow an assistance system to localise the bleeding site and remove the blood obscuring the surgeon's vision.

In this work, we evaluate various Weakly-Supervised methods for semantic segmentation using only image-level labels that only indicate the presence or absence of a blood accumulation. The supervision provided to these networks is only Image-Level Labels.

We are also comparing these networks in this work to a Fully-supervised learning and a non-learning based approach. The idea is to use the results from these approaches to estimate the lower and upper bound for the performance of Weakly-supervised methods.

We are considering two datasets here.

- The First dataset contains images with regions where there is active bleeding and as a result there are regions where blood is accumulated and images where there is no bleeding. The dataset may contain images with surgical instruments in them. Henceforth these negative images known as Simple negative.
- Like the First dataset, the Second dataset contains images with regions where blood is accumulated and regions where there is no blood; however, it also contains regions where there is some blood, but there is no accumulation. These additional images have blood spots or sites where a surgical cut was made, or the tissue was cauterised and may contain surgical instruments in them. Henceforth these negative images are known as Hard negative.

We are trying to use the above two datasets to see how the introduction of hard to distinguish images will impact the performance of the models. The premise behind this being if we only have bloody versus non-bloody images, a Weakly supervised model will only learn to detect blood in general. With the inclusion of Hard negatives, the hope is that it will learn to distinguish blood accumulations from regions that only have little blood. In the dataset, we have image-level labels to train the networks and pixel-level labels available to check the module performance during the evaluation phase. For evaluation, four-fold cross-validation (see Section 3.12) is used. We use four-fold cross-validation as this is a good trade-off between training set size and computation time. The evaluation has three variations.

- The model is trained on the dataset without Hard negatives and then evaluated on the dataset without Hard negatives. Henceforth referred to as SNSN.
- The model is trained on the dataset with Hard negatives dataset and then evaluated on the dataset with Hard negatives dataset. Henceforth referred to as HNHN.
- The model is trained on the dataset with Hard negatives dataset and then evaluated on the dataset without Hard negatives. Henceforth referred to as HNSN.

We compare three Weakly-supervised approaches, one Fully-supervised approach and one non-learning computer vision transformations based approach on various parameters to determine the viability of image-level label to identify the blood accumulations.

2. Related Works

Since weakly-supervised semantic segmentation helps reduce the resources, cost, and time required to generate data with pixel-level labels by relying on a weaker form of supervision, this method has been researched a lot in the past few years. There are various approaches to provide weak supervision, like bounding boxes, scribbles and image-level labels. Among these weak annotations for semantic segmentation, image-level class labels have been widely used as they are easier to generate or are already included in large scale datasets. Using image-level labels proposes a significant challenge as the labels do not provide any information about the location of the object and its shape; they only indicate the object's presence [4].
Various approaches have been proposed to cover the supervision gap in the past few years. The majority of these approaches use class activation maps (CAM) to try and localise objects with the help of image classification labels which are further refined and used as masks that are used to visualise the region which was most relevant for the model's prediction. Some of the other approaches uses information generated from saliency maps or affinity matrix as a secondary source of information in an attempt to bridge the gap between supervised and weakly supervised segmentation.

2.0.1 Weakly-supervised Segmentation

The method **grad-cam** [5] is one of the most cited method and is used frequently in visual explanation [6] tasks and various neural networks utilises Grad-CAM as part of their architecture. This paper is evaluated here for its simplicity and ability to be used in a ad-hoc fashion. This method is further discussed in the Section 4.2.

Another one of the most simplistic, frequently cited approaches is followed in the paper **WILDCAT: Weakly Supervised Learning of Deep ConvNets for Image Classification, Pointwise Localization and Segmentation** [7] and it is the precursor to grad-cam [5]. The paper aims to highlight and explains the working of class activation maps and to achieve this goal the paper utilises global-level image labels and aims to improve the weakly supervised semantic segmentation by implementing the following changes to existing deep Weakly Supervised Learning (WSL) models:

1. Using fully convolutional networks to preserve spatial resolution.

2. Incorporating a multi-map WSL transfer layer used to learn multiple localised related to complementary class modalities.

3. Proposing a new pooling method to provide global image predictions and solve the problem of aggregating spatial scores into a global prediction.

It has two steps class-wise pooling, and then a spatial pooling module is used to select relevant regions to support the predictions.

The paper **Constrained-CNN losses for weakly supervised segmentation** [8] was evaluated to understand the impact of using constraints to optimise the loss function. The paper utilises the concept of Constrained CNNs and aims to embed high-order (global) inequality constraints on the network outputs directly in the loss function to guide learning process. It focuses on constrained Lagrangian dual [9] optimisation and introduces a differentiable penalty, which enforces inequality constraints directly in the loss function, avoiding expensive Lagrangian dual iterates and proposal generation.

The paper **AffinityNet: semi-supervised few-shot learning for disease type prediction** [4] utilises one of the most frequently used methods of creating an affinity matrix. This paper was considered as a potential candidate for the thesis and was evaluated as such to determine the viability for the task at hand. The paper aims to solve the issue of segmenting local discriminative parts rather than the entire object area. They propagate such local responses to nearby areas which belong to the same semantic entity. The network predicts semantic affinity between a pair of adjacent image coordinates. A random walk then realises the semantic propagation with the affinities predicted by AffinityNet. The core idea here is that supervision employed to train AffinityNet is given by the initial discriminative part segmentation, which is incomplete as a segmentation annotation but sufficient for learning semantic affinities within small image areas. Thus the entire framework relies only on image-level class labels and does not require extra data or annotations. Image and its CAMs are used to build a neighbourhood graph where each pixel is connected to its neighbours within a certain radius and estimate semantic affinities of pairs connected in the graph through AffinityNet. Sparse activations in CAMs are then diffused by random walk [10] on the graph for each class. The affinities on edges in the graph encourage random walk to propagate the activations to nearby and semantically identical areas and penalise propagation to areas of the other classes. This semantic diffusion revises CAMs significantly so that fine object shapes are recovered. This process trains images to synthesise their segmentation labels by taking the class label associated with the maximum activation of the revised CAMs at each pixel. The generated segmentation labels are used to train a segmentation model for testing. The remaining issue is how to learn AffinityNet without extra data or additional supervision. To this end, the initial CAMs of training images are utilised as sources of supervision. Presumably because CAMs often miss some object parts and exhibit false alarms, they are incomplete as supervision for learning semantic segmentation whose goal is to accurately predict the entire object masks. To generate reliable labels of the local semantic affinities, the areas with relatively low activation scores on the CAMs are processed in a manner that only confident object and background areas remain. A training example is then obtained by sampling a pair of adjacent image coordinates on the confident areas, and its binary label is 1 if its coordinates belong to the same class and 0 otherwise. The overall pipeline:

1. CAMs of training images are computed and utilised to generate semantic affinity labels, which are then used as supervision to train AffinityNet.

2. Then, trained AffinityNet is applied to each training image to compute the semantic

affinity matrix of its neighbourhood graph, which is employed in random walk to revise its CAMs and obtain synthesised segmentation labels.

3. The generated segmentation labels are used to train a semantic segmentation DNN, which is the only network used at test time.

The reason for not including AffinityNet in this work is that the construction of the semantic affinity matrix takes a lot of computational time and resources and requires a lot of parameters to be optimised for the data type, which is not feasible in the time frame of the thesis. Furthermore these challenges makes the method harder to use and less transferable to new data settings.

The paper **Weakly-and Semi-Supervised Learning of a DCNN for Semantic Image Segmentation** [11] builds its model on the DeepLab [12] model for semantic image segmentation. They use a DCNN to predict the label distribution per pixel, followed by a fully-connected (dense) Conditional Random Fields (CRF) to smooth the predictions while preserving image edges.

The paper **Weakly-Supervised Semantic Segmentation Network with Deep Seeded Region Growing** [13] proposes that a Weakly-supervised network generates new labels using the contextual information within an image. Using the image-level labels enables finding small and sparse discriminative regions from the object of interest, termed as "seed cues". The neighbouring pixels of seed cues with similar features (e.g. colour, texture or deep features) could have the same labels as the seed cues. Following this sequence the classical Seeded Region Growing (SRG) method [14] is used to model this process for generating accurate and complete pixel-level labels.

The paper **CIAN: Cross-Image Affinity Net for Weakly Supervised Semantic Segmentation** [15] discusses and argues about the vitality of the cross-image relationship. This paper was chosen and studied to see the affinity net's vitality and supplement the understanding of the affinity net. The paper proposes generating initial seeds from image labels then use them for training the segmentation network. It trains a classification network with a global-pooling layer (see Section 3.14) right before the last classification layer. After training, it removes the global-average pooling layer and directly applies the classification layer to every pixel in the feature map to obtain the score map. A Siamese backbone obtains embedded features for both query and reference images, followed by the cross-image affinity module to augment the features. The CIAN module derives the affinity from the query and the reference features, and then the query retrieves supplementary information from the reference accordingly.

The paper **Weakly-Supervised Image Semantic Segmentation Using Graph Convolutional Networks** [16] is one of the initial papers chosen for this work but was later, after extensive experimentation and evaluation, not included in the final work. The paper aimed to solve the issue that the random walk imposes no regularisation on the quality of the generated complete pseudo labels because of the feed-forward nature of propagating the activation scores of Class Activation Maps. This is a significant problem faced by the other networks evaluated and implemented in the thesis. The paper proposes using Graph Convolutional Neural Network(GCN) based feature propagation (propagating the high-level semantic features of image pixels, then decoding the propagated features into semantic predictions). They proposed formulating the generation of complete pseudo labels as a semi-supervised learning task and then training a two-layer GCN separately for every training image by

back-propagating a Laplacian and an entropy regularisation loss. GCNs are chosen instead of the convolutional neural networks presumably because of the irregular affinity relations among feature samples.

There are two variants proposed in the paper, and the better performing variant is WSGCN-I. It constructs the affinity matrix using the boundary detection network and uses the features that are in the last layer of the boundary detection network and before its 1×1 convolutional layer as the node features.

The second variant WSGCN-P follows AffinityNet [10] in specifying the affinity matrix and uses the semantic features for affinity evaluation as node features. This originates from the general observation that high-level semantic features usually contain less information about spatial details, and WSGCN-I also considers low-level colour/spatial information in regularising label predictions. The network was not chosen in the end as the optimising of the affinity matrix, and GNN required significant time and resources. Also, Graph convolutional networks suffer from many issues while working with Image data. The non-Euclidean characteristic of graphs (e.g., the irregular structure) makes the convolutions and filtering on graphs not as well-defined as on images [17].

From a computational perspective, the eigenvector matrix requires the explicit computation of the eigenvalue decomposition of the graph Laplacian matrix and hence suffers from the $O(n^3)$ time complexity which is impractical for large-scale graphs. Even though the eigenvectors can be pre-computed, the time complexity is still $O(n^2)$, and there are $O(n)$ parameters to be learned in each layer. Besides, these non-parametric filters are not localised in the vertex domain. The deciding factor to not continue further with this paper was that we have two massive datasets and to make an affinity matrix for one image took 10.58 seconds during the experimental phase. This delay can be associated with the computational complexity of the process and the fetch delay of the networked system. Furthermore, training the network and itself proved challenging due to the network size. This also makes the network less transferable to new data settings.

The paper **Self-supervised Scale Equivariant Network for Weakly Supervised Semantic Segmentation** [18] discusses scale equivariant regularisation and its importance in ensuring consistency of Class Activation Map (CAM) from the same input image with different resolutions. This novel scale equivariant regularisation can guide the whole network to learn more accurate class activation. This was the inspiration for SEAM. Similarly to SEAM [19], a randomly selected RGB image and its downsampled copy are fed into the two branch networks, respectively. At the end of the backbone networks,the $C-1$ channels feature maps are achieved, where $C-1$ is the number of categories excluding the background.

The paper **Discriminative Region Suppression for Weakly-Supervised Semantic Segmentation** [20] discusses the problem with using classifiers for semantic segmentation as the localisation maps obtained from the classifier focus only on sparse discriminative object regions; it is challenging to generate high-quality segmentation labels. Discriminative Region Suppression (DRS) suppresses the attention on discriminative regions and spreads it to adjacent non-discriminative regions, generating dense localisation maps. This is done by diffusing the attention into adjacent non-discriminative parts. The max-element extractor extracts K maximum elements from intermediate feature maps. These K maximum elements are the maximum points of each discriminative region and are considered as starting points to be suppressed. The module consists of three components max-element extractor, suppression controller, and suppressor. These components work together to produce dense localisation maps by reducing the attention gap between discriminative regions and

adjacent non-discriminative regions. Using the dense localisation maps obtained from the refinement network, generated pseudo segmentation labels are used as weak-supervision for the semantic segmentation network. This was programmed but not used in the final implementation. The saliency information extraction requires optimisation and is prone to errors. They are generated to be used as background clues and are very sensitive to the thresholding value and, when performing experimentation, did not produce viable results. The module was scrapped after two weeks as adding one more network to extract the information is not resource frugal. This paper was also chosen for its ability to assist with the overactivation problems but was later not included in the final work due to implementation issues, time, and resource requirements.

The paper **Pseudo-mask Matters in Weakly-supervised Semantic Segmentation** [21] highlights various changes and suggestions to improve the pseudo mask generation. The key points discussed here are (i) Coefficient of Variation Smoothing to smooth the CAMs adaptively; (ii) Proportional Pseudo-mask Generation to project the expanded CAMs to pseudo-mask based on a new metric indicating the importance of each class on each location, instead of the scores trained from binary classifiers. (iii) Pretended Under-Fitting strategy to suppress the influence of noise in pseudo-mask; (iv) Cyclic Pseudo-mask to boost the pseudo-masks during fully supervised semantic segmentation (FSSS) training.

The paper **Anti-Adversarially Manipulated Attributions for Weakly and Semi-Supervised Semantic Segmentation** [22] states that often, the most discriminative regions of the image are focused during weakly supervision semantic segmentation and often, these regions are very small. They propose a method to involve other regions as well. These regions do not sufficiently specify the whole region belonging to the target object. The new manipulation method for extending the discriminative regions of a target object works oppositely to Adversarial attack, which finds a small perturbation of an image that pushes it across the decision boundary to change the classification result. It aims to find a perturbation that pushes the manipulated image away from the decision boundary. An image is perturbed along pixel gradients which increases the classification score of the target class. The result is that non-discriminative regions, which are nevertheless relevant to that class, gradually become involved in the classification so that the CAM of the manipulated image identifies more regions of the object. They also introduce regularisation terms that suppress the scores of other classes and limit the attribution scores of the regions that already have high scores. This is done to solve the problem that Ascending the gradient ensures that classification score increases, but the repetitive ascending may cause irrelevant areas, such as parts of the backgrounds or regions of other objects, to be activated together or the attribution scores of some part of the target object to be increased dramatically.

The paper **Mining Cross-Image Semantics for Weakly Supervised Semantic Segmentation** [23] addresses cross-image semantic relations' value for comprehensive object pattern mining. To achieve this, two neural co-attentions are incorporated into the classifier to capture cross-image semantic similarities and differences complimentary. In particular, given a pair of training images, one co-attention enforces the classifier to recognise the common semantics from co-attentive objects, while the other one, called contrastive co-attention, drives the classifier to identify the unshared semantics from the rest, uncommon objects. This helps the classifier discover more object patterns and better ground semantics in image regions.

The paper **Causal Intervention for Weakly-Supervised Semantic Segmentation** [24] was chosen to understand the pseudo labels generated and the problems we face with them from a logical perspective. The paper proposes a structural causal model to analyse the causalities among images, contexts, and class labels. Context Adjustment (CONTA) is introduced here to remove the confounding bias in image-level classification and thus provide better pseudo-masks as ground-truth for the subsequent segmentation model. It gives three reasons: Object Ambiguity, Incomplete Background, and Incomplete Foreground. It further proposes a classifier to remove confounders to generate better CAMs.The paper defines confounders and tries to visualise their impact on the pseudo-labels.

The paper **Railroad is not a Train: Saliency as Pseudo-pixel Supervision for Weakly Supervised Semantic Segmentation** [25] attempts to address the following limitations of CAMs: sparse object coverage, inaccurate object boundaries, and co-occurring pixels from non-target objects and proposes a framework, Explicit Pseudo-pixel Supervision (EPS), which learns from pixel-level feedback by combining two weak supervisions to overcome the aforementioned challenges. The image-level label provides the object identity via the localisation map, and the saliency map from the off-the-shelf saliency detection model offers rich boundaries. We devise a joint training strategy to utilise the complementary relationship between both information fully. This was initially considered as a potential module but later dropped due to it being dependent on salience information.

The paper **Puzzle-CAM: Improved localization via matching partial and full features** [26] aims to solve the issue with under and overactivation of CAMs by shuffling the image cams and then restitching them together to make better CAMs. This paper is chosen to be evaluated in this work despite the computational limitations.This paper is further discussed in Section 4.3

The paper **Self-supervised Equivariant Attention Mechanism for Weakly Supervised Semantic Segmentation** [19] aims to refine the CAMs using self attention mechanism and uses Siamese networks to achieve this. This paper is chosen to be evaluated in this work as this approach provides a good balance between complexity and simplicity of the network architecture.This paper is further discussed in Section 4.1

We further evaluate the paper **Saliency Guided Self-attention Network for Weakly and Semi-supervised Semantic Segmentation** [27] to understand the implementation of the self-attention mechanism and the importance of saliency maps. It integrates class-agnostic saliency maps and class-specific attention cues to enable the self-attention mechanism (see Section 3.2) to work effectively under weak supervision. These two types of aforementioned priors are fused adaptively in SGAN to help generate high-quality seeds. By replacing saliency maps and attention cues with partially labelled segmentation ground-truth, SGAN works effectively for semi-supervised semantic segmentation. It has a saliency Guided Self-attention Network where there is a CNN backbone to learn deep feature representations, a saliency guided self-attention module that propagates attention from small discriminative parts to non-discriminative regions via capturing long-range contextual dependencies and an image classification branch together with a seed segmentation branch to supervise the training of the entire network. A slightly modified VGG-16 is used as the backbone of the network.

2.0.2 Blood Segmentation

The paper **Real-time identification of blood regions for hemostasis support in laparoscopic surgery** [28] proposes a machine learning approach where it uses colour features (i.e., RGB and HSV values, as well as their combinations) and a supervised SVM(support vector machine) classifier to determine if the pixel can be classified as bloody or not. The method first removes dark pixels, and then the image is utilised. Most discriminative features are selected during the training phase, and SVM is then trained with parameter optimisation.

The paper **Segmentation of Bleeding Regions in Wireless Capsule Endoscopy Images an Approach for inside Capsule Video Summarization** [29] proposes a method that has three steps

1. Colour space conversion.
2. Training a neural network for local classification of bleeding regions.
3. Neural Network quantisation.

By considering pixel value in each of the three Red, Green, Blue channels as an index of lookup table, they identify which channels or colour spaces best represent the bleeding region. A multilevel perceptron is used. A patch around each pixel is considered, and these extracted patches from the three colour spaces are combined and fed to the network. The class of central pixel is considered as the output of the network. They further apply binarisation to reduce the complexity of the neural network.

The paper **Active Blood Detection in a High Resolution Capsule Endoscopy using Color Spectrum Transformation** [30] was originally aimed to detect bleeding sites in capsule endoscopy without requiring any training and this is adopted and an implementation is created here in this work as a benchmark. This paper is further discussed in Section 4.5.

Using spatio-temporal hybrid features for detecting bleeding point in laparoscopic surgery [31] The paper utilises the concept of optical flow where they measure the change of pixel in time domain between current and the previous frame and the correlation between adjacent frames in the image sequence to get motion information about the object in two adjacent frame. They use FlowNet to accomplish this and the output of it is fed to the Fast R-CNN model alongside with the image during training.

2.0.3 Fully Supervised Semantic Segmentation

There are various fully supervised semantic segmentation approaches and one of the most frequently cited approaches is **U-Net: Convolutional Networks for Biomedical Image Segmentation** [32] it follows a fully supervised approach to semantic segmentation. Here the contracting network is supplemented with successive layers where we replace pooling operations by upsampling operators. This paper is chosen to be evaluated in this work as a benchmark. This paper is further discussed in Section 4.4

3. Theoretical Background

In this section we look at the theoretical background of various approaches and explain some of the algorithms in detail.

3.1 Convolutional Neural Networks(ConvNet/CNNs)

These are Deep Learning algorithms that take images as input and then assign learnable biases to various image elements to allow for distinction. They require less preprocessing than regular image processing [33, 2, 34, 35]. CNNs are good at getting temporal and spatial dependencies in an image. They can do so because the weights are shared and a reduced number of parameters. Typical architecture can be seen in Figure 3.1

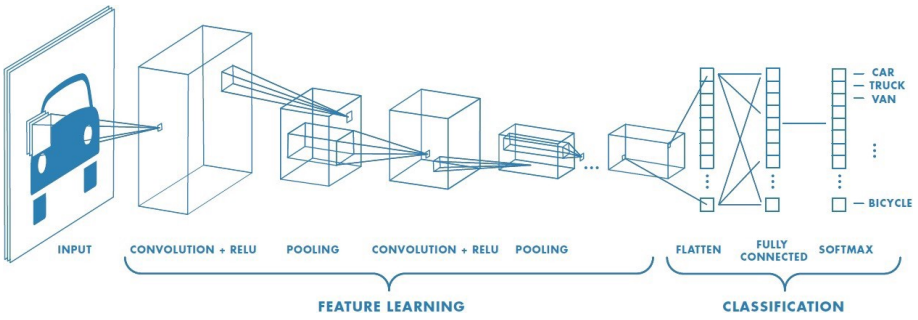

Figure 3.1: Architecture of typical CNN [36]

3.1.1 The Kernel (Convolution Layer)

These are filters that are used to extract information from the images. They are also a matrix that moves over the input images and performs dot products. They have a stride value that corresponds to the number of columns a filter can move. They move from left to right and then drop down to the next row. They do this till the whole image is traversed. The kernel depth is the same as the input image if multiple channels are present, and matrix multiplication is done between the kernels and each channel. Then they are squashed to give one depth channel output (convoluted features). The idea here is to extract high-level features like edges, colour and gradient orientations. We can have multiple convolutional layers and where the first layer generally captures the low-level info, and the subsequent layers generate high-level feature maps [37]. See Figure 3.2 to see typical kernel operation. The results generated are of two types:

1. The convolved feature map has reduced dimensionality.
2. The convolved feature map has increased or similar spatial dimensions.

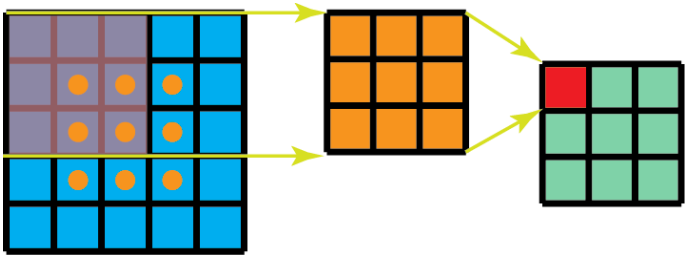

Figure 3.2: Typical kernel operation [38]

We can add padding (blank spaces) to stop the dimensionality reduction (spatial down-sampling).

3.1.2 Image

The input is an RGB image that has three separate channels Red, Green, Blue respectively (see Figure 3.3). These are 2D matrix with numerical values representing pixels. Here the image is from the institutes dataset which contains blank images and has blank normal and blank images blood accumulation. Each image is resized to a resolution of 512×512 or 256×256.

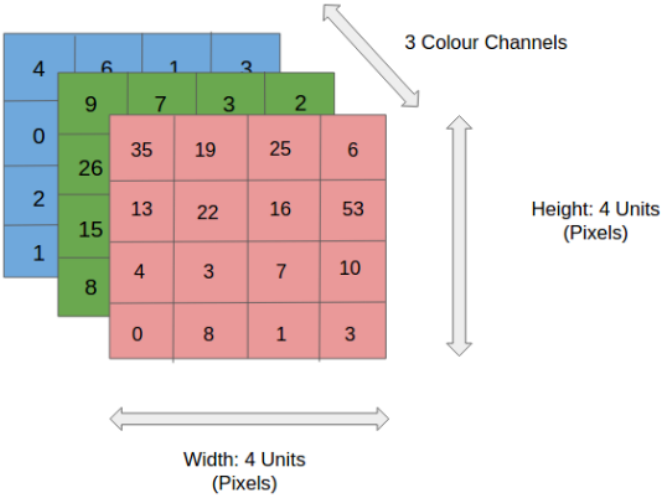

Figure 3.3: Typical 3 channel RGB image[36]

3.1.3 Pooling Layer

These are similar to convolution layers. These layers also reduce the spatial size of the convolved feature. It is useful as it reduces resources consumption and extracts the dominant features. These dominant features are rotationally and positionally invariant. Any aggregation function can be used for pooling (sum,median,random etc.) but max and average pooling are the most commonly used in Deep Learning [37, 2] (see Figure 3.4).

1. **Max Pooling:** return the minimum value from the portions of the image that the kernel covers. This method acts as a noise suppressant as it discards noisy activations. It would retain information if a feature was found.
2. **Average pooling:** return the average of all the values from the portion of the image that the kernel covers. This method only makes dimensionality reduction.

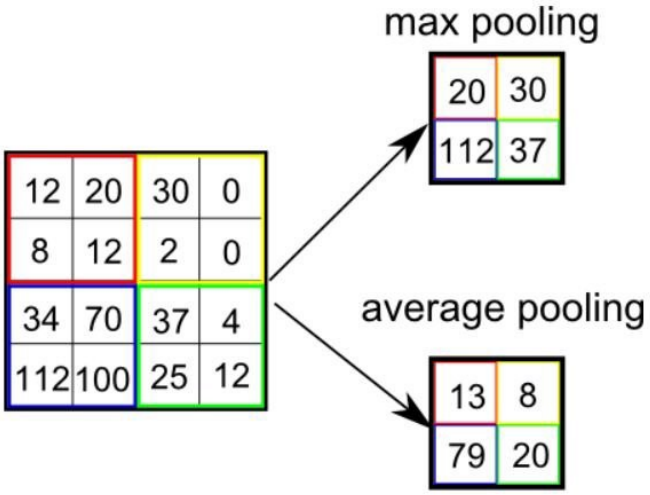

Figure 3.4: Pooling layer working example [36, 39]

The above two layers (convolution and pooling) form the hidden layers of the neural networks. These can be increased or decreased based on the model's complexity or defined by the model used. The kernels can also have various fixed values and size depending on the image type and channels used.

3.1.4 Classification

A Fully-Connected Neural Network is a kind of artificial neural network (NN) where all the neurons/nodes of one layer are connected to all the neurons/nodes in the next layer [35]. A multilevel perceptron consists of Fully-Connected layers and has one input and output layer. It can have one or many hidden layers between input and output layers [35]. A Fully-Connected Neural Network is a kind of artificial neural network (NN) where all the neurons/nodes of a layer are connected to all the neurons/nodes in the next layer [35]. A multilevel perceptron consists of Fully-Connected layers and has one input and output layer. It can have one or many hidden layers between input and output layers [35]. A Fully-Connected layer (belonging to a Fully-Connected NN) can be a computationally simple way to learn non-linear combinations of the high-level features in the output space. After passing through the above layers, the image becomes suitable for our multilevel perceptrons [40]. Here we can apply Backpropagation to every iteration of training. The training is done over a series of epochs, and we perform classification using the Softmax Classification technique.

3.1.4.1 Mathematically

[41]

- Output of j,k$^{\text{th}}$ neuron of a layer for a 1x3x3(Channels x Height x Width) kernel is:

$$a_{l,j,k} = \sigma(b_l + \sum_{m=0}^{2}\sum_{n=0}^{2} w_{l,j,k}.a_{l-1,j+m,k+n}) \tag{3.1}$$

- Softmax: It turns Logits(unnormalised predictions) into probabilities

$$\text{Softmax}(x_i) = \frac{\exp(x_i)}{\sum_j \exp(x_j)} \tag{3.2}$$

- We can use categorical cross-entropy as a cost function.

$$-C_n = -\sum_j y_{n,j} \cdot \log y\,(x_n)_j \tag{3.3}$$

  where

$$y_{n,j}: \text{ j-entry of target output}$$

$$y\,(x_n)_j: \text{ j-entry of output}$$

- Backpropagation can be explained in simplified mathematical equations as:
  - A neuron $X_j^{(l)}$ can take input as:

$$n_j^{(l)} = \sum_{i=1}^{N_{l-1}} a_i^{(l-1)} w_{ij}^{(l)} + b_j^{(l)}, \qquad j = 1, 2, \ldots, N_l. \tag{3.4}$$

  - and the activation a_i of neuron $X_j^{(l)}$ can be defined as :

$$a_j^{(l)} = f^{(l)}(n_j^{(l)}) = f^{(l)}(\sum_{i=1}^{N_{l-1}} a_i^{(l-1)} w_{ij}^{(l)} + b_j^{(l)}). \tag{3.5}$$

– When we consider, mean squared error and supervised learning. Then we assume a set of associations is given, and they have N elements $\mathbf{s}_N^{(q)} : \mathbf{t}_N^{(q)}, q = 1, 2, \ldots, Q$ and the square of the error for these pairs can be defined as:

$$E = \|\mathbf{y}(t) - \mathbf{t}(t)\|^2, \tag{3.6}$$

where $\mathbf{t}(t) = \mathbf{t}_N^{(q)}$

– we have a set of rules that update the weights and biases:

$$w_{ij}^{(l)}(t+1) = w_{ij}^{(l)}(t) - \alpha \frac{\partial E}{\partial w_{ij}^{(l)}(t)} \tag{3.7}$$

$$b_j^{(l)}(t+1) = b_j^{(l)}(t) - \alpha \frac{\partial E}{\partial b_j^{(l)}(t)} \tag{3.8}$$

where $\alpha(> 0)$ is the learning rate.

3.2 Generalised Self-Attention Mechanism

In the attention mechanism, we map a query vector and key-value pair set vector to an output vector [42] (See Figure 3.5) The output is a weighted sum of the values where a

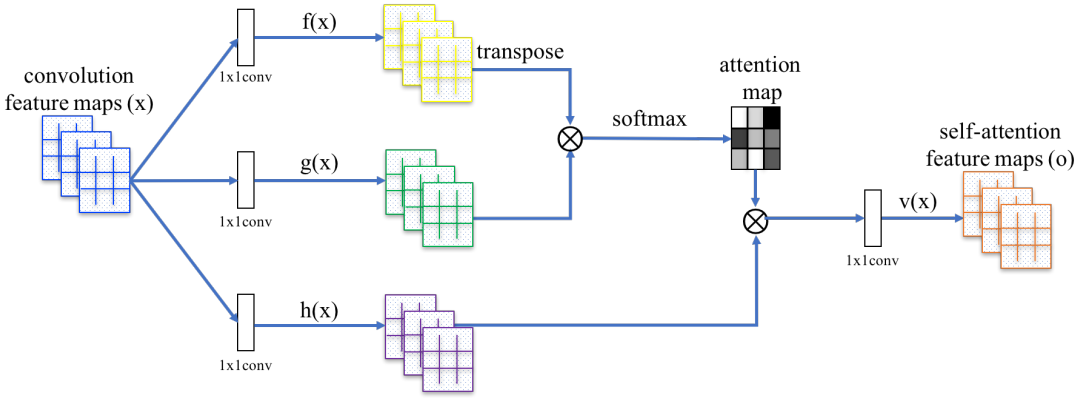

Figure 3.5: Generalised implementation for self attention mechanism[43] Here $f(x), g(x), h(x)$ are query,keys and values.

compatibility function of the query computes the weight assigned to each value with the corresponding key. Self-Attention is an attention mechanism where different positions of a sequence are considered in relation to one another and to get a representation of the same sequence. Attention reweighs specific network features based on some internally supplied weights. In soft attention, these weights are continuous, and in hard attention, these weights are binary [44, 45]. At a position, a non-local operation that can compute a response as a weighted sum of the features at all positions in the input feature maps can be defined as:

$$\mathrm{y}_i = \frac{1}{\mathcal{C}(\mathrm{x}_i)} \sum_{\forall j} f(\mathrm{x}_i, \mathrm{x}_j)\, g(\mathrm{x}_j) + \mathrm{x}_i \tag{3.9}$$

Where i and j are, and f is an Embedded Gaussian function, that is used to compute similarity in an embedding space. We can define f as:

$$f(\mathrm{x}_i, \mathrm{x}_j) = e^{\theta(\mathrm{x}_i)^{\mathrm{T}} \phi(\mathrm{x}_j)} \tag{3.10}$$

in this function $\theta(\mathrm{x}_i)^{\mathrm{T}}\,\phi(\mathrm{x}_j)$ are the embedding. The output signal is then normalised by

$$\mathcal{C}(\mathrm{x}_i) = \sum_{\forall j} f(\mathrm{x}_i, \mathrm{x}_j) \tag{3.11}$$

Function $g(\mathrm{x}_j)$ gives a representation of input signal x_j at each position and all of them are aggregated into position i with the similarity weights given by $f(\mathrm{x}_i, \mathrm{x}_j)$, which calculates the dot-product pixel affinity in an embedding space. [19]

3.3 Saliency Map

In the context of visual processing, Saliency refers to the unique feature of the image like pixels, brightness, resolution of the image etc. Saliency Maps helps in refocusing the attention of the network to the foreground object and ignoring the background object [46]. A saliency map is an image where the brightness of a pixel is directly proportional to its saliency, and generally, the image is grey-scale. Saliency maps can be considered a heat map where hotter regions of the image represent regions that have a significant impact on predicting the object's class. This guides the selection of locations where the attention is diverted based on the spatial distribution of saliency. The Saliency maps can be considered as a topographical representation of the most visually alluring regions of the image. These maps were first proposed in Itti et al.[47] and were first used in the context of deep learning in Simonyan et al.[48].

3.4 Sigmoid

The Sigmoid [49] function is a monotonic non-linear activation function (see Figure 3.6). It has a typical S-shaped curve. It takes real values and maps them between 0 and 1.

$$[49]S(z) = \frac{1}{1+e^{-z}} \tag{3.12}$$

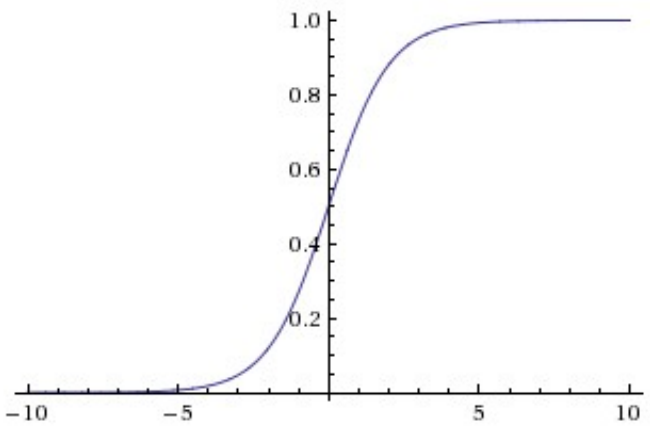

Figure 3.6: sigmoid [39]

3.5 Tanh

Tanh [49] is a hyperbolic tangent activation function similar to Sigmoid but maps real values between -1 and 1 (see Figure 3.7).

$$[49]\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}} \tag{3.13}$$

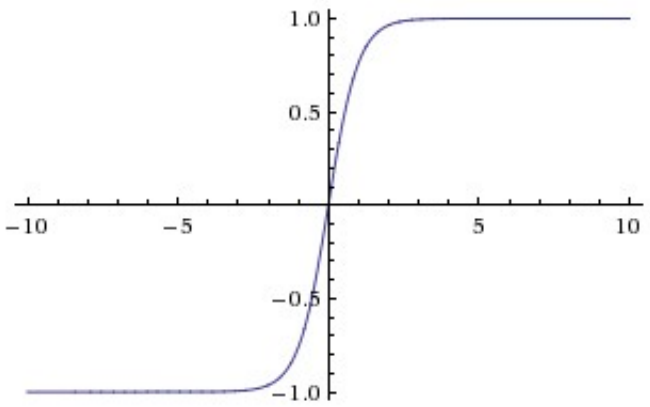

Figure 3.7: tanh[39]

3.6 ReLU

Tanh and Sigmoid functions are non-linear activation functions. They have some limitations; they saturate very quickly, and they are more sensitive to values close to the midpoint of the input, so to solve this, we use ReLU [50]. It is locally linear function, but overall it's non-linear (see Figure 3.8). The main benefits are that computation is easy, and the negative inputs can output true zero values.

$$[49]R(z) = \left\{ \begin{array}{c} z\&z > 0 \\ 0\&z <= 0 \end{array} \right\} \tag{3.14}$$

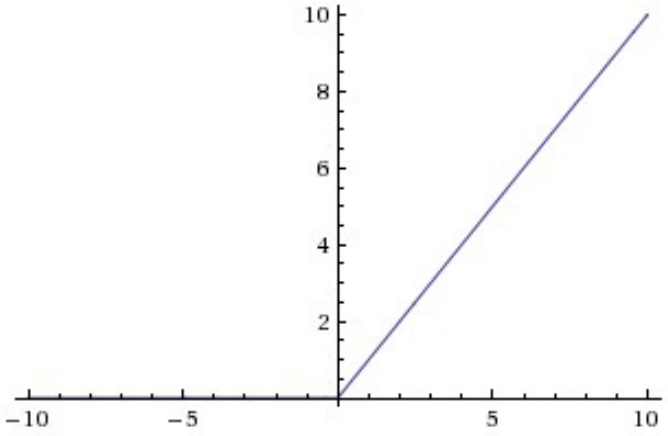

Figure 3.8: ReLU[39]

3.7 Binary Cross-Entropy

Cross-entropy is defined as a measure of the difference between two probability distributions for a given random variable or set of events. It is used as a loss function for binary classification tasks by measuring the difference between the predicted and ground-truth distribution [51].

$$L_{BCE}(y, \hat{y}) = -(y \log(\hat{y}) + (1 - y) \log(1 - \hat{y})) \tag{3.15}$$

Here, $\hat{y}$ is the predicted value by the prediction model.

3.8 Confusion Matrix

It is table that helps in visualising the performance of a algorithm/method. The goal is to maximise True Positive (TP), True Negative (TN) and minimise False Positive (FP), False Negative (FN). The number of FP represents the "fake" bleeding pixels in the prediction mask. FN represents "bleeding pixel" that the model ignores. If the FP is too large, the surgeon will be interrupted with false information. The model ignores some bleeding on too large FN values, but the surgeon is interrupted less, which is a more viable option. There are various evaluation methods that can be based on confusion matrix [52, 53, 54].

$$\texttt{recall} = \frac{TP}{TP + FN} \tag{3.16}$$

$$\texttt{precision} = \frac{TP}{TP + FP} \tag{3.17}$$

$$\texttt{specificity} = \frac{TN}{TN + FP} \tag{3.18}$$

$$\texttt{accuracy} = \frac{TP + TN}{TP + TN + FP + FI} \tag{3.19}$$

$$f_1 = \frac{2 \times \texttt{precision} \times \texttt{recall}}{\texttt{precision} + \texttt{recall}} \tag{3.20}$$

		Real category		
		p	**n**	**total**
Estimation	**p′**	True Positive	False Negative	P′
	n′	False Positive	True Negative	N′
	total	P	N	

(3.21)

3.9 Dice Loss

This is a loss function that is widely used in medical image segmentation as this tends to perform better when there is a imbalance between foreground and background pixels [55, 51]. This loss however does not take into account weather the image is easy or hard to evaluate [31].

$$D = \frac{2 \sum_i^N p_i g_i}{\sum_i^N p_i^2 + \sum_i^N g_i^2} \tag{3.22}$$

or in other terms it can be written as:

$$TP\left(I_m, I_e\right) = \sum_{x,y \in (0,255)} I_m(x,y) \times I_e(x,y) \tag{3.23}$$

$$\text{Positiv}(I) = \sum_{x,y \in (0,255)} I(x,y) \tag{3.24}$$

$$\texttt{Dice}\,\text{Loss}\left(I_m, I_e\right) = 1 - \frac{2 \times TP\left(I_m, I_e\right) + s}{\text{Positive }\left(I_m\right) + \text{ Positive }\left(I_e\right) + s} \tag{3.25}$$

3.10 Multilevel Soft Margin Loss

Creates a criterion that optimises a multi-label one-versus-all loss based on max-entropy, between input xx and target yy of size (N, C) (N,C) [56].

$$\text{loss}(x,y) = -\frac{1}{C} * \sum_i y[i] * \log\left((1+\exp(-x[i]))^{-1}\right) + (1-y[i]) * \log\left(\frac{\exp(-x[i])}{(1+\exp(-x[i]))}\right)$$

where $i \in \{0, \cdots, \text{x.nElement }()-1\}, y[i] \in \{0,1\}$.

3.11 Stochastic Gradient Descent

This is an iterative method based on the concept of stochastic approximation of gradient descent. The gradient is replaced by an estimate calculated on a randomly selected subset of the data. Gradient descent, namely batch gradient descent, computes the gradient of the cost function with respect to the parameters θ for the entire training dataset [35, 57]:

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta) \tag{3.26}$$

Stochastic gradient descent (SGD) in contrast performs a parameter update for *each* training example $x^{(i)}$ and label $y^{(i)}$:

$$\theta = \theta - \eta \cdot \nabla_\theta J(\theta; x^{(i)}; y^{(i)}) \tag{3.27}$$

3.12 K-Fold Validation

It is a re-sampling procedure with a single parameter called k that represents the number of groups the data is to be divided. The original sample is split randomly into k equal-sized subsamples. One of these subsamples is used for validation and others for training. The process is repeated k times, where each of the k subsamples are used exactly once as the validation data [58]. The value for k throughout the thesis is 4, which is a good balance between the data size of the training set and the time required to train the model. Dataset is split into four groups, with an image belonging to one surgery in one group. Images from the same surgery often have high similarities. If we do a random split, then the training and evaluation set will get similar images. Since in this scenario, the module will have seen a similar image in the training phase, this leads to fake validation during the evaluation phase [59, 60].

3.13 Otsu's Thresholding

In thresholding, we separate foreground and background pixels. Otsu's method [61, 62] is a variance-based technique. The idea is to go through all the possible values of thresholds and measure the spread of foreground and background pixels. The threshold is chosen from the region where the spread is minimal. In other words, we iteratively search for a threshold that minimises the weighted sum of variances of background and foreground pixels. The minimal value is chosen. The formula for finding the within-class variance at any threshold t is given by:

$$\sigma^2(t) = \omega_{bg}(t)\sigma^2_{bg}(t) + \omega_{fg}(t)\sigma^2_{fg}(t) \tag{3.28}$$

where $\omega_{bg}(t)$ and $\omega_{fg}(t)$ represents the probability of number of pixels for each class at threshold t and σ^2 represents the variance of colour values.

3.14 Global Average Pooling

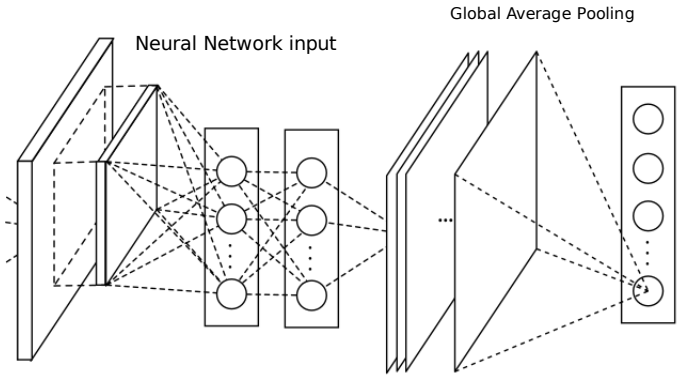

Figure 3.9: Gap Layer connected to the output of a NN layer belonging to a Deep NN [63]

It calculates the average value of every feature map (output of a filter applied to an input image or previous layer). It reduces the dimensions and thus reduces the data. Since there are no parameters to optimise, thus there's no overfitting [63]. Figure 3.9 illustrates GAP layer. Let's assume that we have tensor that looks like (batch,height,width,channel) now for an example, for an input tensor of shape $(10, 10, 10, 32)$ the output will be $(10, 1, 1, 32)$. In the above example, average pooling is applied on the spatial dimensions, but other dimensions remain unaffected.

3.15 CAMs

For a particular category c class activation map (CAM) [64] represents the discriminative regions of the image that the CNN uses to identify that particular category. for a particular category c class activation map represents the discriminative regions of the image that the CNN uses to identify that particular category. Let I be the input image. After the last layer of a CNN, the output is passed through a global average pooling (GAP) layer. If the last layer had $N * i * i$ feature maps, the GAP will take N channels and outputs their spatial average. The channels that have higher activation will have higher signals. Each output category is assigned a weight based on augmenting dense linear layers with Softmax. A dot product of these maps with the feature maps from the previous layer is done to generate class activation maps. Here M_c is the class activation map where spatial element is given by:

$$M_c(x, y) = \sum_k w^c_k f_k(x, y) \tag{3.29}$$

Where $f_k(x, y)$ is activation of feature k in last Convolutional layer at location (x, y) and w_k^c is the weight corresponding to class c for unit k. Essentially, w_k^c indicates the *importance* of F_k for class c. how the above equation is derived is further explained in [64] They can be used to localise objects by image classification labels. However, they are plagued by problems like under activation and over activation. In under activation, parts of the objects are labelled as foreground. In over activation, parts of the background are also labelled as foreground. Moreover, they do not work consistently for Affine transformation augmented images.

3.16 Upsampling

It is an approach that is the opposite of pooling. Here, the abstract image representation is upsampled to the input's spatial dimensions [65]. There are various approaches to this; the most common approaches are:

1. **Nearest Neighbours:** Take input pixel and copy it to nearest neighbours. (see Figure 3.10)
2. **Bi-Linear Interpolation:** The four nearest pixel values of the input pixel are taken, and a weighted average based on the distance of the four nearest cells is performed to smooth the output. (see Figure 3.11)
3. **Max-Unpooling:** The Max-Pooling layer in CNN takes the maximum among all the values in the kernel. To perform max-unpooling, the maximum value index is saved for every max-pooling layer during the encoding step then the saved index is used during the Decoding step. Here the input pixel is mapped to the saved index; every else zero are filled. (see Figure 3.12)
4. **Transposed Convolutions:** These can be used to perform the task of upsampling as well. Ignoring The image channels and taking stride of 1 and no padding. Let us take a $n_h \times n_w$ input tensor and a $k_h \times k_w$ kernel. Sliding the kernel window with stride of 1 for n_w times in each row and n_h times in each column yields a total of $n_h n_w$ intermediate results. Each intermediate result is a $(n_h + k_h - 1) \times (n_w + k_w - 1)$ tensor that are initialised as zeros [66, 66]. (see Figure 3.13) To compute each intermediate tensor, each element in the input tensor is multiplied by the kernel so that the resulting $k_h \times k_w$ tensor replaces a portion in each intermediate tensor. In the end, all the intermediate results are summed over to produce the output.

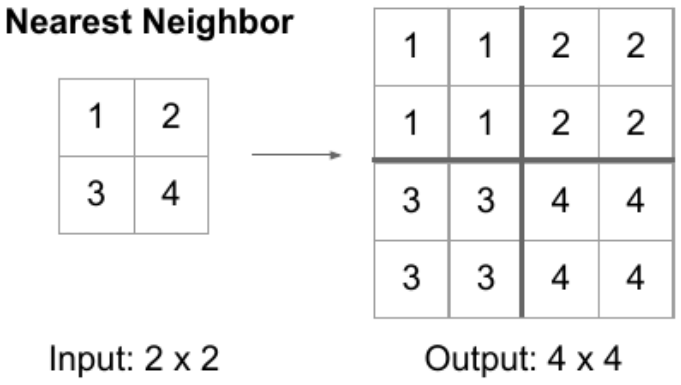

Figure 3.10: Nearest Neighbour upsampling [67]

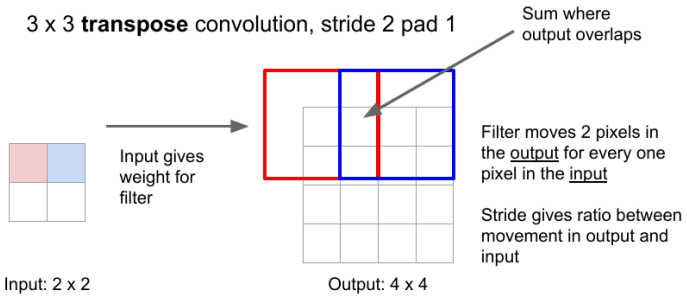

Figure 3.11: Transpose Convolution upsampling [67]

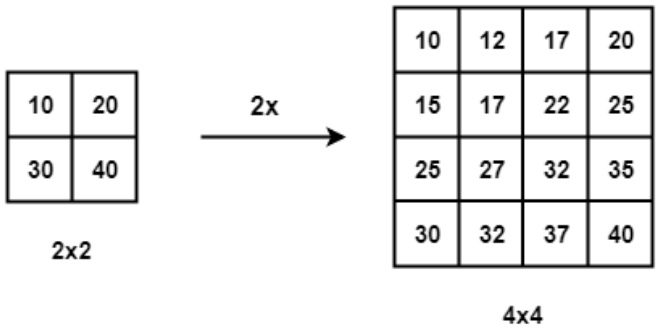

Figure 3.12: Bed of Nails upsampling [68]

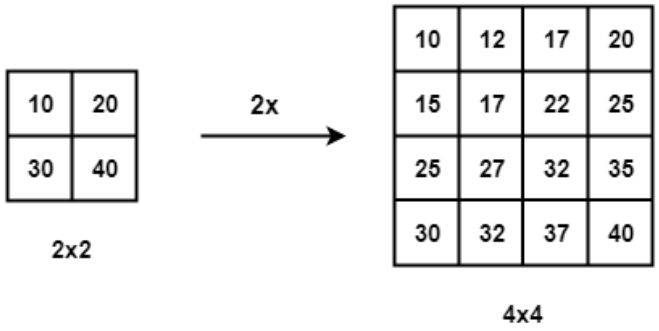

Figure 3.13: Bi-Linear Interpolation [68]

3.17 ResNet

See Figure 3.14 to see a simple ResNet structure. When we keep on stack Convolutional and Pooling layers on a plain convolutional network to increase its depth and performance, it is observed that the network performs worse during training and test. They propose that overfitting is not the issue, and to make a really deep network, they have proposed an architecture. Instead of stacking deep layers on top of each other and having every layer learn some underlying mapping of the desired function, we have blocks where we try and fit a residual mapping instead of direct mapping. The hypothesis behind this is an optimisation problem, and a deep model must be as good as a shallow model. Learned layers from a shallow model can be copied, and additional layers can be set to identity mapping and creating proof by construction of a deep model for this hypothesis [69].

Formally, denoting the desired underlying mapping as $\mathcal{H}(x)$, we let the stacked nonlinear layers fit another mapping of $\mathcal{F}(x) := \mathcal{H}(x) - x$. The original mapping is recast into $\mathcal{F}(x) + x$. We hypothesise that it is easier to optimise the residual mapping than to optimise the original, unreferenced mapping. To the extreme, if an identity mapping were

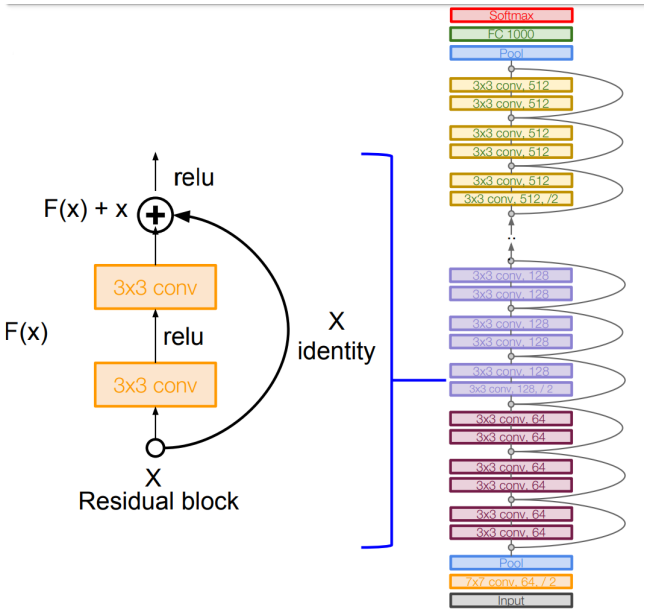

Figure 3.14: **Left** A residual unit **Right** A sample structure of ResNet [39, 69] where Global average pooling layer is used after last convolutional layer

optimal, it would be easier to push the residual to zero than to fit an identity mapping by a stack of nonlinear layers. The formulation of $\mathcal{F}(x) + x$ can be realised by feed-forward neural networks with shortcut connections. In this work we have used ResNet as backbone for various networks in different sizes. Each method follow slightly different adaption of this basic module above [69].

3.18 Weakly Supervised Semantic Segmentation

Technically this problem can be denoted as:
Let's assume that the space of images is $\mathcal{X}$.

Then For any image $X \in \mathcal{X}$, a segmentation mask Y is a collection, $(y_1, \ldots, y_n)$, of semantic labels at n spatial locations.

These semantic labels belong to a set $\mathcal{C} = \mathcal{C}' \cup \{c^{\text{bg}}\}$ of size k, where $\mathcal{C}'$ is a set of all foreground labels and c^{bg} is a background label.

Let's assume that the training data, $\mathcal{D} = \{(X_i, T_i)\}_{i=1}^{N}$, consists of N images, $X_i \in \mathcal{X}$, where each image is weakly annotated by a set, $T_i \subset \mathcal{C}'$, of foreground labels that occur in the image.

The goal is to train a deep convolutional neural network $f(X;\theta)$ which is parameterized by θ, that models the conditional probability of observing any label $c \in \mathcal{C}$ at any location $u \in \{1, 2, \ldots, n\}$, i.e. $f_{u,c}(X;\theta) = p(y_u = c \mid X)$. For the sake of simplicity we can omit the parameters θ in our notation and write $f(X;\theta)$ simply as $f(X)$ [70].

3.19 Normalisation Layers

See Figure 3.15 to see a visual representation of all the operations. Normalisation [71, 72] is a preprocessing technique that we use to standardise data. It is imperative as it does the following important tasks:

- It normalises each feature to maintain the contribution of every feature, as some features have higher numerical value than others. This helps make the network unbiased.

- It reduces Internal Covariate Shift, which can be describes as the change in the distribution of network activations occurring as a result of change in network parameters during training. Our goal is to reduce internal covariate shift to improve the training.
- It makes the Optimisation faster because normalisation doesn't allow weights to explode all over the place and restricts them to a certain range.

3.19.1 Batch-Normalisation

In this method, we normalise activations in a network across the mini-batch of definite size. For each feature mean and variance of that feature in the mini-batch is computed. Then mean is subtracted, and the feature is divided by its mini-batch standard deviation. Algorithm 3.30 shows the Batch Normalisation algorithm [71, 72].

$$\begin{array}{|l|l|}
\text{Input:} & \text{Values of } x \text{ over a mini-batch: } \mathcal{B} = \{x_{1\ldots m}\} \\
 & \text{Parameters to be learned: } \gamma, \beta \\
\text{Output:} & \{y_i = \mathrm{BN}_{\gamma,\beta}(x_i)\} \\
\mu_{\mathcal{B}} & \leftarrow \frac{1}{m}\sum_{i=1}^{m} x_i \\
\sigma^2_{\mathcal{B}} & \leftarrow \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_{\mathcal{B}})^2 \\
\widehat{x}_i & \leftarrow \frac{x_i - \mu_{\mathcal{B}}}{\sqrt{\sigma^2_{\mathcal{B}} + \epsilon}} \quad // \text{ mini-batch mean} \\
y_i & \leftarrow \gamma \widehat{x}_i + \beta \equiv \mathrm{BN}_{\gamma,\beta}(x_i)
\end{array} \tag{3.30}$$

3.19.2 Group-Normalisation

Here we normalise over a group of channels for each training example.

$$\mu_i = \frac{1}{m}\sum_{k \in \mathcal{S}_i} x_k, \quad \sigma_i = \sqrt{\frac{1}{m}\sum_{k \in \mathcal{S}_i}(x_k - \mu_i)^2 + \epsilon} \tag{3.31}$$

$$\mathcal{S}_i = \left\{ k \mid k_N = i_N, \left\lfloor \frac{k_C}{C/G} \right\rfloor = \left\lfloor \frac{i_C}{C/G} \right\rfloor \right\} \tag{3.32}$$

$$\hat{x}_i = \frac{1}{\sigma_i}(x_i - \mu_i) \tag{3.33}$$

$$y_i = \gamma \hat{x}_i + \beta \tag{3.34}$$

Here, x is the feature computed by a layer, and i is an index. In the case of 2D images, $\mathbf{i} = (\mathbf{iN}, \mathbf{iC}, \mathbf{iH}, \mathbf{iW})$ is a 4D vector indexing the features in (N, C, H, W) order, where N is the batch axis, C is the channel axis, and H and W are the spatial height and width axes. $\mathbf{G}$ is the number of groups, which is a pre-defined hyper-parameter. C/G is the number of channels per group. $L\rfloor$ is the floor operation, and " $\lfloor kC/(C/G) \rfloor = \lfloor iC/(C/G) \rfloor$ " means that the indexes i and k are in the same group of channels, assuming each group of channels are stored in a sequential order along the C axis. GN computes μ and σ along the (H, W) axes and along a group of C/G channels [71, 72].

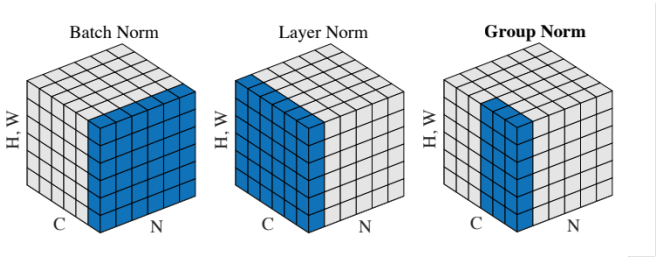

Figure 3.15: Visualisation of various normalisation operations [39]

3.20 IOU

"IOU is the area of overlap between the predicted segmentation and the ground truth divided by the area of union between the predicted segmentation and the ground truth" [73] (see Figure 3.16).

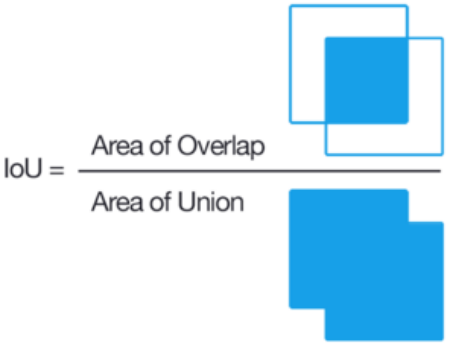

Figure 3.16: iou [73]

3.21 F1-Score

"The Dice Coefficient is 2 * the Area of Overlap divided by the total number of pixels in both images " [73] (see figure 3.17).

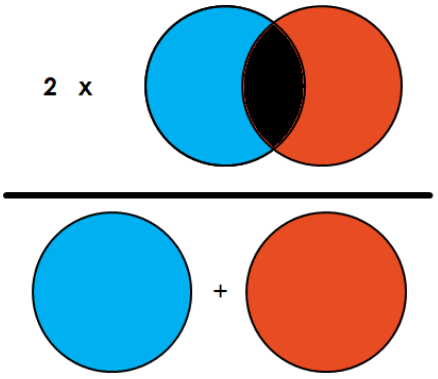

Figure 3.17: f1 [73]

3.22 Nesterov Momentum

This is a different version of momentum update [74], which enjoys greater theoretical convergence guarantees for convex functions. The central idea of Nesterov momentum is that when the current parameter vector is at some position x then by observing the momentum update above, we can infer that the momentum term alone (i.e. ignoring the second term with the gradient) will be nudging the parameter vector by $mu * v$. So if we are going to compute the gradient in that instance, we can assume the approximate future position to be $x + mu * v$ and consider this as a "lookahead" (a point in the vicinity of where we will soon end up). Hence we can computer the gradient at $x + mu * v$ instead of position x [75, 74, 39]. (see Figure 3.18)

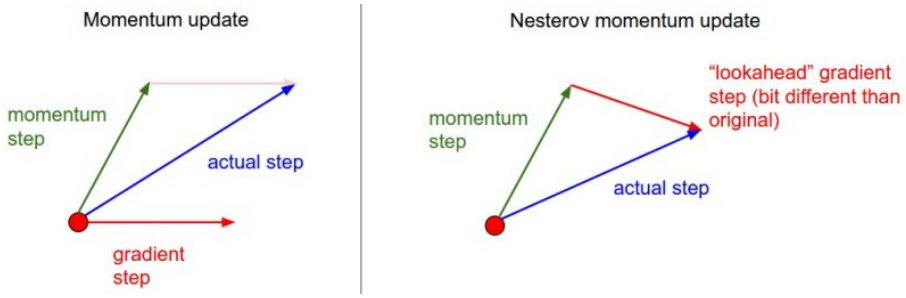

Figure 3.18: nesterov vs standard momentum update.Instead of evaluating gradient at the current position (red circle), we know that our momentum is going towards the tip of the green arrow. With Nesterov momentum we therefore instead evaluate the gradient at this "looked-ahead" position [74].

3.23 Adam Optimiser

"The method computes individual adaptive learning rates for different parameters from estimates of first and second moments of the gradients" [76] Here g_t^2 indicates the element wise square $g_t \odot g_t$., all operations on vectors are element-wise and With β_1^t and β_2^t we denote β_1 and β_2 to the power t [76].

Require:
α : Step size
$\beta_1, \beta_2 \in [0, 1)$: Exponential decay rates for the moment estimates
$f(\theta)$: Stochastic objective function with parameters θ
θ_0 : Initial parameter vector
Ensure:
$m_0 \leftarrow 0$ (Initialize 1^{st} moment vector)
$v_0 \leftarrow 0$ (Initialize 2^{nd} moment vector)
$t \leftarrow 0$ (Initialize timestep)
while θ_t *not converged* **do**
$t \leftarrow t + 1$
$g_t \leftarrow \nabla_\theta f_t(\theta_{t-1})$ (Get gradients w.r.t. stochastic objective at timestep t)
$m_t \leftarrow \beta_1 \cdot m_{t-1} + (1 - \beta_1) \cdot g_t$ (Update biased first moment estimate)
$v_t \leftarrow \beta_2 \cdot v_{t-1} + (1 - \beta_2) \cdot g_t^2$ (Update biased second raw moment estimate)
$\widehat{m}_t \leftarrow m_t / \left(1 - \beta_1^t\right)$ (Compute bias-corrected first moment estimate)
$\widehat{v}_t \leftarrow v_t / \left(1 - \beta_2^t\right)$ (Compute bias-corrected second raw moment estimate)
$\theta_t \leftarrow \theta_{t-1} - \alpha \cdot \widehat{m}_t / \left(\sqrt{\widehat{v}_t} + \epsilon\right)$ (Update parameters)
return θ_t (Resulting parameters)

Algorithm 1: algorithm for Adam [76]

3.24 Downsampling

This aims to reduce the spatial dimension of a image, it can be done by using pooling (see section 3.1.3) or as a result of strided convolution [67]. (see Figure 3.19

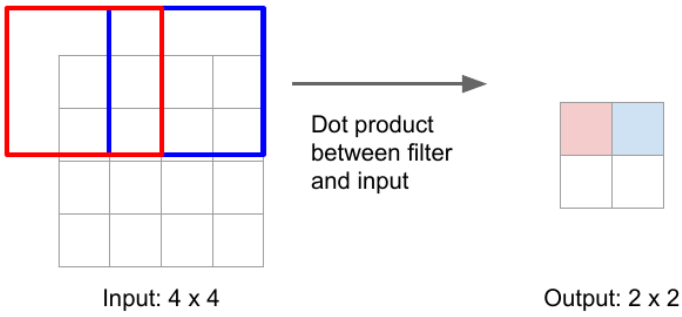

Figure 3.19: Strided covolution resulting downsampling [67]

4. Methods

The main goal of this work is to evaluate the capability of weakly supervised techniques to perform semantic segmentation with only image-level labels (see Section 3.18). There are various approaches discussed in the related work section. Based on further analysis, available resources and experiments, we will look at three implementations used for weakly supervised semantic segmentation and compare them to one fully supervised implementation and one classical no-training based approach. The supervised and non-training approaches act as benchmarks for the three weakly supervised approaches. The weakly supervised approaches utilise various ways to refine CAMs (see Section 3.15) and improve the pseudo masks. Further in the chapter, we discuss two datasets that we have used here and discuss their relevance.

There are multiple thresholding methods [1] but empirically it is determined that Otsu's thresholding (see Section 3.13) generates best results among these methods. So henceforth we will be using Otsu's Thresholding as the main method for dynamic thresholding and will compare it against a fixed threshold value.

4.1 SEAM: Self-supervised Equivariant Attention Mechanism for Weakly Supervised Semantic Segmentation

The paper states that class activation mapping (CAM) generally does not serve well because of the distinction between fully supervised and Weakly-supervised segmentation. This paper bases itself on the observation that equivariance is an implicit constraint in Fully-supervised segmentation, where input images have the same spatial transformation as pixel-level labels when doing data augmentation.

The core idea of the paper is to utilise the above-proposed hypothesis to enforce consistent segmentations between augmented versions of the same image. However, it is proposed that this equivariance constraint is lost when we train CAMs using image-level labels.

Hence, they propose consistency regularisation on predicted CAMs from various transformed images (to give self-supervision for network learning) and use a pixel correlation module (PCM) to exploit context appearance information and refine the prediction of current pixel by its similar neighbours.

The goal of this module is to cover the weaknesses of CAMs. CAMs are generally used in most methods that aim to localise objects with the help of image classification labels.

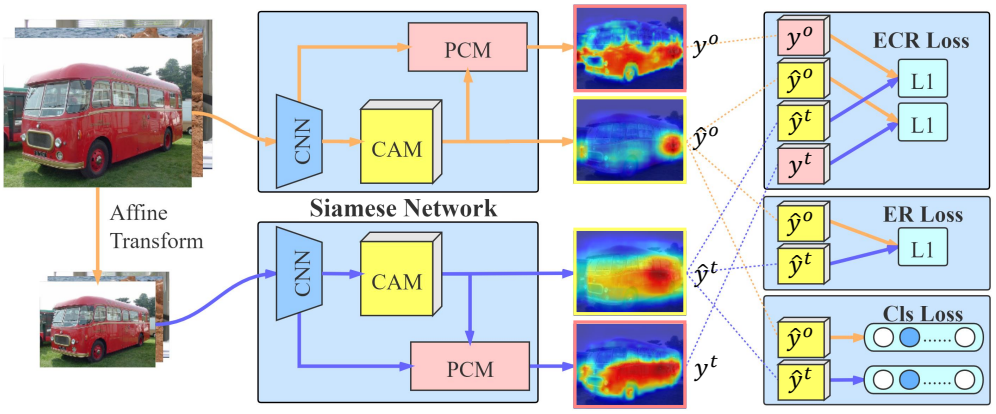

Figure 4.1: The Siamese network architecture of SEAM[19]

However, CAMs often only label the most discriminative region of the object (under-activation) or incorrectly label the background region as the object (over-activation).

Furthermore, CAMs show inconsistent results when the images are augmented with affine transformation. In SEAM, we apply consistency regularisation on CAMs using various transform images to provide self-supervision for Network learning. We also introduce the pixel correlation model (PCM), which launch affinity attention maps to revise original CAMs based on captured context appearance information for each pixel.

Two Siamese networks are used here, which use equivariant cross regularisation and shared weights. The purpose of regularisation here is to bridge the gap between fully and weakly supervised segmentation. The concept of self-attention is also used here to capture feature dependencies [19].

4.1.1 Motivation Behind Using This Module

The module argues over the hypothesis that weakly-supervised and fully-supervised semantic segmentation tasks have the same optimal parameters. Thus based on the aforementioned hypothesis, most models train a classifier first and then remove the pooling layer to perform segmentation. They argue that pooling operation introduces an invariance and semantic segmentation tasks are more inclined towards equivariant. There is no equivariant constraint for the classification function. Thus the aforementioned hypothesis does not hold, and self-attention mentioned must be used alongside equivariant regularisation to improve the ability of the network for consistent predictions. The source paper explains the argument in depth [19].

As mentioned above this is why we use generalised self-attention mechanism (see Section 3.2) incorporated with Equivariant Regularisation in an attempt to improve the approximation ability of the network and to generate consistent predictions.

4.1.2 Model Description

4.1.2.1 Equivariant Regularisation

In standard supervised segmentation, the supervision is provided in the form of pixel-level labels. The labels have a similar affine transformation as the input image resulting in an implicit equivariant constraint that helps to regularise the module's output. This option to perform transformations similar to the input image lacks weak supervision, and to overcome this lack of constraint, equivariant regularisation is proposed [19]:

$$\mathcal{R}_{ER} = \|F(A(I)) - A(F(I))\| \tag{4.1}$$

Where F is our Neural Network and A any spatial affine transformations we perform. To integrate this method in the module, we use Siamese network architecture with shared weights. Where one branch of the network performs these operations on the output of the network and the other branch performs these operations on the image before it passes to the feed-forward stage [19] (see Figure 4.1).

4.1.2.2 Affine Transformations

Following Affine transformations are chosen performed by the module [77]:

- Random Horizontal flip: Horizontally flip the given image randomly with a given probability 0.5
- Resize: 256x256
- Colour Jitter: Randomly change the brightness, contrast and saturation of an image with a factor of 0.7,0.5,0.2 respectively.
- Random Rotation: rotate the image randomly between -10 to 10 degrees.
- Random Resized Crop: Crop a random portion of image and resize to 256 × 256.

Following code snippet shows the implementation in the code. Transform compose is used to compose several transformations together. These are part of trochvision from Pytorch (see Figure 4.2).

```
DATA_TRANSFORMS = transforms.Compose([
    transforms.Resize((256, 256)),
    transforms.RandomHorizontalFlip(),
    transforms.ColorJitter(brightness=0.7, contrast=0.5,
                           saturation=0.2, hue=0),
    transforms.RandomRotation((-10, 10)),
    transforms.RandomHorizontalFlip(),
    transforms.RandomResizedCrop(256),
    transforms.ToTensor()
])
```

Figure 4.2: Code Snippet for Transforms Compose

4.1.2.3 Pixel Correlation Module

Even though regularisation tries to bridge the gap between supervision by providing additional supervision, it is still hard to get viable equivariance from just the convolution layers, so we incorporate self-attention to refine the predictions and capture context information efficiently. In order to integrate classical self-attention into our module; we can use Equations 3.9 and 3.10 and derive the following Equation:

$$\mathrm{y}_i = \frac{1}{\mathcal{C}\left(\mathrm{x}_i\right)} \sum_{\forall j} e^{\theta(\mathrm{x}_i)^{\mathrm{T}} \phi(\mathrm{x}_j)} g\left(\hat{\mathrm{y}}_j\right) + \hat{\mathrm{y}}_i, \tag{4.2}$$

Here $\hat{\mathrm{y}}$ is the original CAM and y is the revised CAM. $\hat{\mathrm{y}}$ is embedded into the residual space by function g. From Equation 3.10 we get the similarity score. Each pixels aggregates with other pixels based on this score. Other three functions θ, ϕ, g are implemented here with the help of individual 1x1 convolution layers. To further integrate the low-level feature of each pixel at the end of the network, we are using the pixel correlation module (PCM) [19]. PCM is trained using the supervision from the equivariant regularisation. The structure

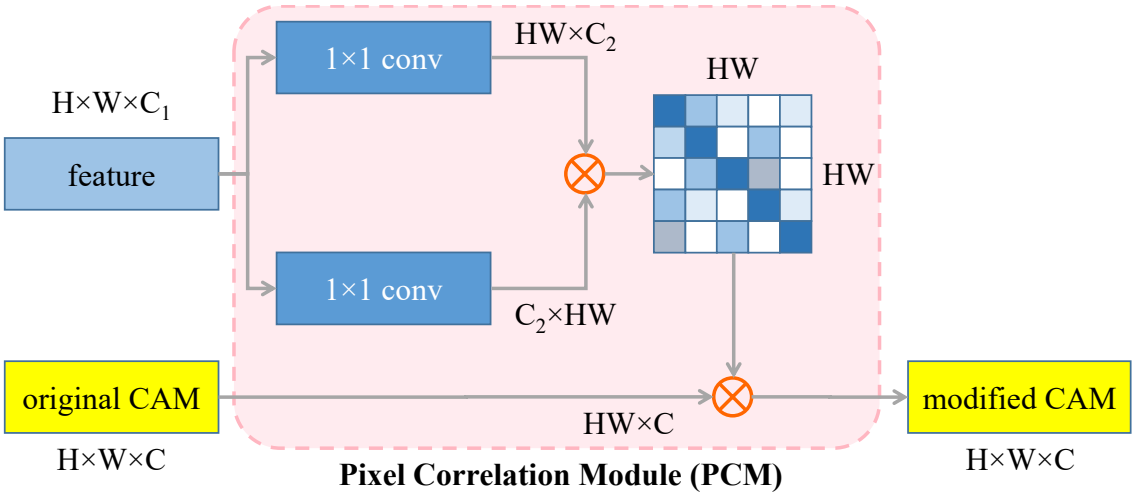

Figure 4.3: The structure of PCM [19], where $H, W, C/C_1/C_2$ denote height, width and channel numbers of feature maps respectively.

of PCM is based on the self-attention mechanism. For inter-pixel feature similarity, cosine distance is used and can be formulated as:

$$f\left(\mathrm{x}_i, \mathrm{x}_j\right) = \frac{\theta\left(\mathrm{x}_i\right)^{\mathrm{T}} \theta\left(\mathrm{x}_j\right)}{\left\|\theta\left(\mathrm{x}_i\right)\right\| \cdot \left\|\theta\left(\mathrm{x}_j\right)\right\|} \tag{4.3}$$

The structure of PCM is based on the self-attention mechanism. In this module, Affinity between pixel i with other pixel is calculated in a normalised feature space f from Section 4.3 can be integrated into Equation 4.2 we get:

$$\mathrm{y}_i = \frac{1}{\mathcal{C}\left(\mathrm{x}_i\right)} \sum_{\forall j} \mathrm{ReLU}\left(\frac{\theta\left(\mathrm{x}_i\right)^{\mathrm{T}} \theta\left(\mathrm{x}_j\right)}{\left\|\theta\left(\mathrm{x}_i\right)\right\| \cdot \left\|\theta\left(\mathrm{x}_j\right)\right\|}\right) \hat{\mathrm{y}}_j \tag{4.4}$$

The parameters are further constrained by removing embedding functions ϕ, g to combat the issue of overfitting on inaccurate supervision since the other branch provides only pixel-level supervision to PCM, which is not as good as the ground truth. In comparison with the classical self-attention, residual connections are removed to maintain the pixel intensity of the original CAM [19]. The structure of the PCM can be visualised in Figure 4.3.

4.1.2.4 Loss Design of SEAM

We are only using Image-Level classification labels here. The module deviates from the source paper Wang et. al.[19] in this section as the loss function is modified. We are using the global adaptive average 2D pooling layer (see Section 3.14)at the end of the network to get a prediction vector p.
Final loss for SEAM is defined as:

$$\mathcal{L} = \mathcal{L}_{cls} + \mathcal{L}_{ER} + \mathcal{L}_{ECR}. \tag{4.5}$$

Classification loss is defined as $C-1$ class object categories as:

$$\begin{aligned} \ell_{cls}(\mathrm{z}, l) = -\frac{1}{C-1} \sum_{c=1}^{C-1} [l_c \log(\frac{1}{1+e^{-\mathrm{z}_c}}) \\ + (1-l_c) \log(\frac{e^{-\mathrm{z}_c}}{1+e^{-\mathrm{z}_c}})]. \end{aligned} \tag{4.6}$$

Where C is 2 (blood puddle, background). Formally, the CAM generated from the branch of the Siamese networks, which has the original image as input, is denoted as $\hat{y}^o$ and the output CAM of the other branch which takes transformed image is denoted as $\hat{y}^t$. The global average pooling layer aggregates them into prediction vector z^o and z^t respectively. The classification loss is calculated on two branches as:

$$\mathcal{L}_{cls} = \frac{1}{2}(\ell_{cls}(z^o, l) + \ell_{cls}(z^t, l)). \tag{4.7}$$

The source paper mentions that they have empirically determined that the output of PCM falls into local minima quickly, such that all pixels are predicted to be in the same class. So they have added equivariant cross regularisation (ECR):

$$\mathcal{L}_{ECR} = ||A(y^o) - \hat{y}^t||_1 + ||A(\hat{y}^o) - y^t||_1. \tag{4.8}$$

It is empirically evident that the inclusion of this loss helps in our case, even though we are doing binary semantic segmentation. The original CAMs regularises PCM outputs on the other branch of the Siamese network. This aids in avoiding CAM degeneration during PCM refinement [19].

Since our task has only one foreground class, the remaining pixels belong to the background and are represented as zeros in original CAMs. These pixels cannot generate gradients to push feature representations. Therefore, we use background score:

$$\hat{y}_{i,bkg} = 1 - \max_{1 \le c \le C-1} \hat{y}_{i,c}, \tag{4.9}$$

where $\hat{y}_{i,c}$ is the activation score of original CAM for category c at position i. We normalise the activation vectors of each pixel by suppressing foreground non-maximum activations to zeros and concatenate with additional background score. During inference, we only keep the foreground activation results and set the background score as $\hat{y}_{i,bkg} = \alpha$, where α is the hard threshold parameter.
In summary the losses are used roughly as follows:

- The classification loss is used for object localisation. This is represented as L_{cls}
- ER loss is used to narrow the supervision gap. This is represented as $\mathcal{L}_{ER}$
- ECR Loss is used for integrating PCM to the trunk of the neural network.This is represented as $\mathcal{L}_{ECR}$.This also helps in maintaining consistency of predictions over various affine transformations.

4.1.3 Implementation Details

The module uses ResNet-38, (see Section 3.17) where we integrate the PCM as shown in Figure 4.4

```
cam_rv = F.interpolate(self.PCM(cam_d_norm, f), (H,W), mode='bilinear',
                                        align_corners=True)
cam = F.interpolate(cam, (H,W), mode='bilinear', align_corners=True)
return cam, cam_rv
```

Figure 4.4: Code Snippet PCM

Since the source paper performs multi-class classification, during the evaluation phase, they only consider the images where there is an instance of the object, and to accomplish this, they use image-level labels to filter out the cams of the object classes present in

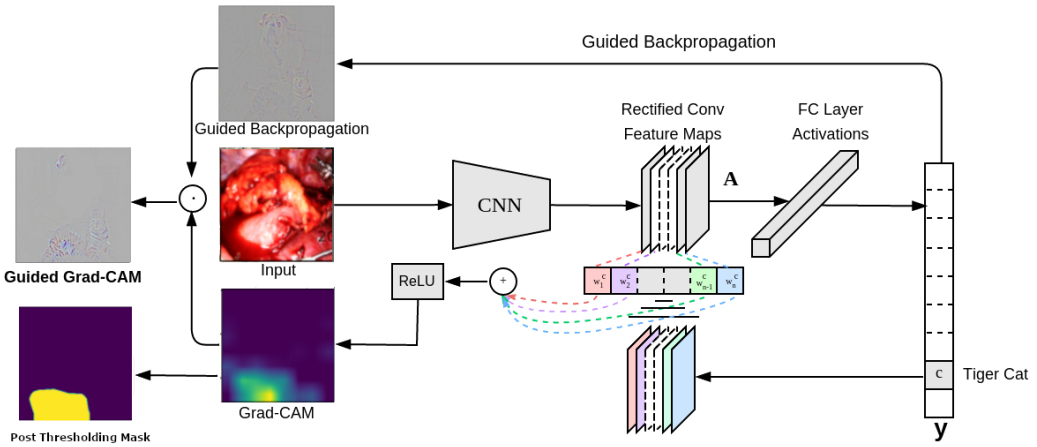

Figure 4.5: [6]Overview: Given an image I, and a category C as input, we forward propagate the image through the model to obtain the raw class scores. This signal is then Backpropagated to the rectified convolutional feature map of interest, where we can compute the coarse Grad-CAM localisation (blue heatmap), we then threshold on this to generate our predicted masks. Optionally, we can pointwise multiply the heatmap with Guided Backpropagation to get visualisations which are both high-resolution and class-discriminative. This step is not implemented in this work.

the source image. However, for the implementation in this work, some images do not have any foreground objects. The source paper's official implementation uses image-level labels as keys to selectively choose the CAMs corresponding to the classes present in the image during evaluation. However, this information is not available in real-world scenarios; hence, we can introduce a pooling layer to generate labels from the classifier. We generate an empty mask when the classifier predicts that the image has no blood. Separate classification accuracy scores are also recorded. To generate final masks dynamic thresholding technique, namely Otsu's thresholding (see Section 3.13) is used. A fixed thresholding value that generates satisfactory qualitative and quantitative results has also been empirically determined here and has been shown in the results.

4.2 Grad-CAM

4.2.1 Motivation Behind Using This Module

Gradient-weighted Class Activation Mapping approach is a generalisation of CAM which was introduced in the Section 3.15. This module aims to create a weakly supervised guided Grad-CAM that generates visual explanations for any CNN-based network without requiring architectural changes or re-training. We combine the class-discriminative ability of Grad-CAM and weak supervision [78, 79, 80] (see Figure 4.5).

4.2.1.1 Model Description

It has been established in various previous works that deeper layers in CNN capture higher-level visual constructs, and convolutional layers naturally retain spatial information that is generally lost in fully connected layers. The papers Vinogradova et al.[81] and Selvaraju et al.[5] build on this notion and states that the final convolutional layer has the best compromise between high-level semantics and detailed spatial information. According to the authors, the neurons in this layer look for semantic class-specific information in the image [82, 83, 84]. Grad-CAM uses the gradient information flowing into the last convolutional layer of the CNN to assign importance values to each neuron for the regions where the blood gets accumulated [85, 86]. The source paper applies this to various tasks like "Image classification", "Image Captioning" and "Visual Question and Answering"; however, we adopt the network for semantic segmentation. The computation of the attention map

is similar to CAMs (see Section 3.15), we modify the Equation (see Section 3.29) and, instead of the weights w_k^c we use new weights G_k^c. The Backpropagation algorithm is used to calculate the weights G_k^c eventuating in the global pooling layer to no longer be required. Hence, the attention maps can be computed from any network layer; however, keeping with convention and results from the source, the last layer is chosen. The new weights are computed as:

$$G_k^c = \frac{1}{Z} \sum_{u,v} \frac{\partial \hat{y}}{\partial f_k} \tag{4.10}$$

Here where Z is the number of pixels in the feature map f_k and u, v the index of N pixels. Each weight G_k^c is computed as the average over all of the pixels of the derivative of the output $\hat{y}$ with respect to the feature maps f_k of the last convolution layer. The attention map $M_{\text{Grad-CAM}}$ is then computed as a linear combination of the feature maps weighted by the G_k, and upsampled with linear interpolation to compensate the max-pooling layers [78, 80, 5]:

$$M_{\text{Grad-CAM}} = \sum_k^N G_k^c f_k \tag{4.11}$$

During computation of G_k^c while Backpropagating gradients with respect to activations, the exact computation amounts to successive matrix products of the weight matrices and the gradient with respect to activation functions till the final convolution layer that the gradients are being propagated to. Hence, this weight G_k^c represents a partial linearisation of the deep network downstream from f, and captures the "importance" of feature map k for a target class c [78, 80, 5]. Hence the class-discriminative localisation map Grad-CAM $L_{\text{Grad-CAM}}^c \in \mathbb{R}^{u \times v}$ of width u and height v for any class c can be obtained by performing a weighted combination of forward activation maps, and follow it by a ReLU.
For semantic segmentation the $\hat{y}$ is replaced by $\sum_{(i,j)\in\mathcal{M}} y_{ij}^c$ in 4.10, where $\mathcal{M}$ is a set of pixel indices of interest in the output mask. The final class-discriminative localisation map can be mathematically represented as [78, 80, 5]:

$$L_{\text{Grad-CAM}}^c = \text{ReLU}\left(\sum_k G_c^k A^k\right) \tag{4.12}$$

4.2.2 Implementation Details

The backbone network used here is ResNet-50 (see Section 3.17), and we take the last layer before global average pooling as our target layer. Similar to Section 4.1.3 we apply Otsu's thresholding here as well, and qualitative and quantitative results again empirically prove that Otsu's thresholding is the best option for this module as well. Furthermore, similar to Section 4.1.3 we create blank masks for the images the classifier predicts the absence of blood. Grad-CAM generates classification labels alongside cams in the module, and we utilise these labels [5, 87, 88, 89].

4.3 Puzzle-CAM

4.3.1 Motivation Behind Using This Module

When we generate pseudo-labels from CAMs using a Classifier, the module's primary focus is generally on the most discriminative parts of the object in the image. The paper targets this issue and attempts to minimise differences between the features from separate patches and the whole image. The method consists of a puzzle module and two regularisation terms to discover the most integrated region in an object. This module can activate the overall region of an object using image-level supervision without requiring extra parameters,

making it ideal for use in a resource-constrained environment. The paper is based on the observational premise that the tiled sections of the image generate different and inconsistent CAMs in contrast to the CAMs generated by using the whole image. The module uses Siamese network architecture, and reconstruction regularisation loss minimises the loss between reconstructed and original images [26]. The implementation in the thesis deviates from the source paper. The source paper uses ResNet-101 (see Section 3.17) and ResNet-269 as the backbone network of the model; however, the currently available resources are not sufficient to use these networks. Therefore, ResNet-50 is being used as the backbone of the model. The source paper also uses Batch Normalisation which is explained in Section 3.19; however, due to resource constraints, the maximum Batch size we could use with this was 6. This size severely affects the performance of Batch Normalisation [90]. The reason being the Batch Normalisation layer has to calculate mean and variance to normalise the previous outputs across the Batch. To counteract this issue Batch Normalisation layer has been replaced with Group Normalisation layer, and weights from a pre-trained ResNet-50 [91] has been used as well. These two approaches are further compared in the Section 5 (see Figure 4.6)

4.3.2 Model Description

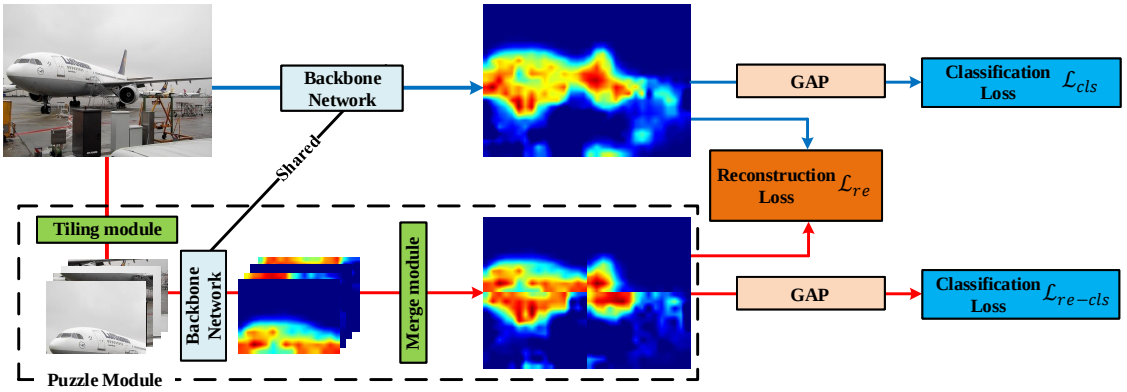

Figure 4.6: Architecture of Puzzle-CAM[26], here the backbone is ResNet-50 (see Section 3.17)

4.3.2.1 CAMs

We first define CAM method for generating the initial attention map. Let F be a feature extractor and θ a classifier, we can generate CAMs A. After training the classifier by image-level supervision, we apply the weights of the c-channel classifier as θ^c on feature map $f = F(I)$ from input image I to obtain the CAM of class (Blood accumulation) c as follows:

$$A_c = \theta_c^\top f. \tag{4.13}$$

The generated CAM is normalised by using the maximum value of A_c.

4.3.2.2 Puzzle Module

The key idea behind matching full and partial features is to bridge the gap between weak and full supervision. To accomplish this goal Puzzle module is introduced; it consists of tiling and merging modules. Given an input image I of size $W \times H$, the tiling module generates non-overlapping tiled patches $\{I^{1,1}, I^{1,2}, I^{2,1}, I^{2,2}\}$ of size $W/2 \times H/2$. Subsequently , we generate $A^{i,j}$ CAMs for each $I^{i,j}$. Finally, the merging module attaches all $A^{i,j}$ into a single CAMs A^{re} that has the same shape as A^s which is the CAMs of I.

4.3.3 Loss

The paper employed a GAP (see Section 3.14) layer at the end of the network to incorporate prediction vector $\hat{Y} = \sigma(G(A_c))$ for image classification and to use binary cross-entropy loss (see Section 3.7) for the classification task, which is in contrast to source material that uses multi-label soft margin loss [56]. For notational convenience, we define Y_t as:

$$\hat{Y}_t = \begin{cases} \hat{Y}, & \text{if } Y = 1 \\ 1 - \hat{Y}, & \text{otherwise} \end{cases} \tag{4.14}$$

$$\ell_{cls}(\hat{Y}, Y) = -log(Y_t). \tag{4.15}$$

The CAMs of the original (A^s) and tiled (A^{re}) images are converted using the GAP layer with prediction vectors $\hat{Y}^s = G(A^s)$ and $\hat{Y}^{re} = G(A^{re})$, respectively. The classification losses for the original and reconstructed images are respectively calculated as:

$$\mathcal{L}_{cls} = \ell_{cls}(\hat{Y}^s, Y), \tag{4.16}$$

$$\mathcal{L}_{p-cls} = \ell_{cls}(\hat{Y}^{re}, Y). \tag{4.17}$$

These two classification losses are used to improve the image classification performance. To reinforce the CAMs from the original image, reconstructing regularisation is added to correspond with the original and reconstructed CAMs. The reconstruction loss for the original CAM is defined as:

$$\mathcal{L}_{re} = \left\| A^s - A^{re} \right\|_1. \tag{4.18}$$

In summary, the final loss can be written as:

$$\mathcal{L} = \mathcal{L}_{cls} + \mathcal{L}_{p-cls} + \alpha \mathcal{L}_{re}. \tag{4.19}$$

where α is the balance of the weights for the different losses. The classification losses, $\mathcal{L}_{cls}$ and $\mathcal{L}_{p-cls}$, are used to roughly estimate the region of the object. The reconstruction loss, $\mathcal{L}_{re}$, is used to narrow the gaps between the pixel- and image-level supervision.

4.3.4 Implementation Details

As mentioned in Section 4.3.1 due to lack of resources and time ResNet-50 is chosen as the backbone for the module. The source paper uses ResNet-101 or ResNet-269 with Batch Normalisation (see Section 3.19) and a batch size of 32. The labels are generated in a similar fashion to Section 4.1.3, and we apply global Otsu's Thresholding; however, unlike the procedure in Section 4.1.3 fix threshold value cannot be determined because whenever we choose a fixed value for a significant number of images, the results become zero and we get a blank mask. It can be hypothesised that due to Puzzle-CAM's inability to generate CAMs for our datasets properly, we only get the most discriminative regions of the object, and the proposed core idea by the authors does not hold for the datasets we have utilised. PolyOptimizer is used for optimisation with weight decay=$1e-4$ and nesterov momentum update (see Section 3.22).

4.4 U-Net/Fully Supervised Approach

4.4.1 Motivation Behind Using This Module

Ronneberger et al.[32] is the paper that first proposed this architecture (see Figure 4.7). The architecture focuses on training using strong data argumentation. The network has

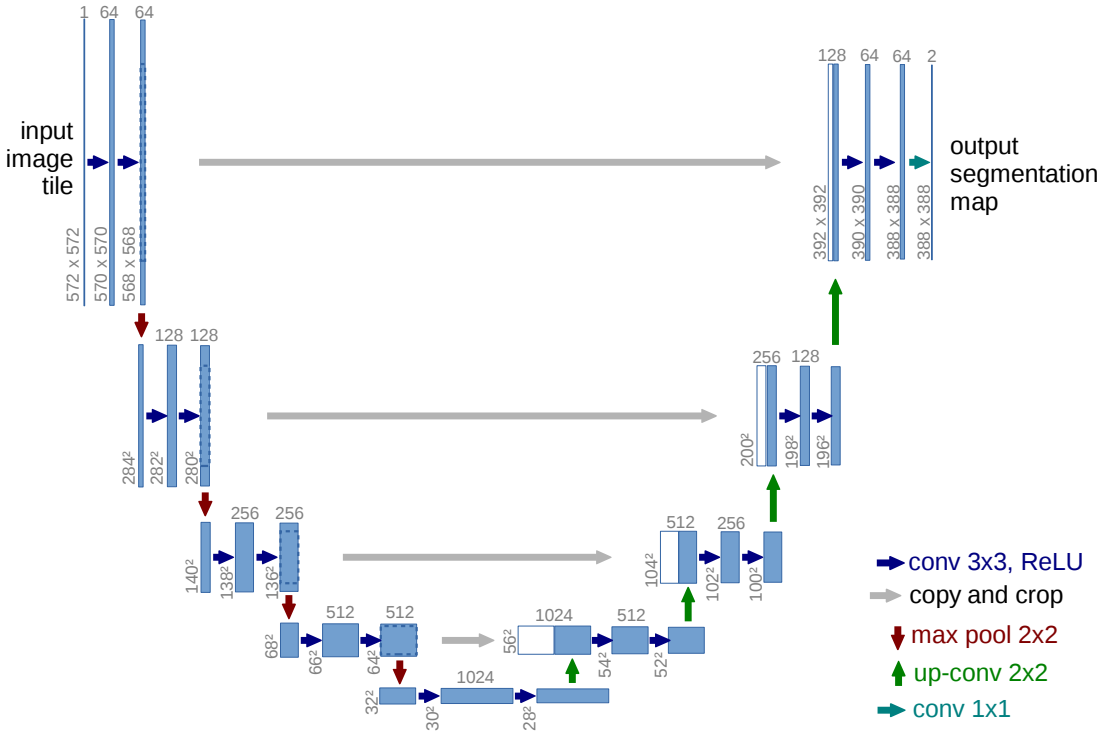

Figure 4.7: U-net architecture (example for 32x32 pixels in the lowest resolution). Each blue box corresponds to a multi-channel feature map. The number of channels is denoted on top of the box. The x-y-size is provided at the lower left edge of the box. White boxes represent copied feature maps. The arrows denote the different operations. [32]

a Contracting path that captures context and an Expanding path that helps with precise localisation. A neural network based on the approach in this paper is chosen as a benchmark. It allows us to compare the performance of weakly supervised networks with a fully supervised approach. This module is used to visualise the gap between two supervision approaches on the same datasets. Other networks out there will perform better than this approach, but they require a lot more resources. Furthermore, this network is frequently used in literature, which allows us to perform a rough comparison to other networks. As of February 2022, the work has been cited over 14,000 time.

4.4.2 Model Description

The core idea of this module is the usual contracting method of a fully convolutional network (FCN) with successive layers in which the pooling operations are replaced with upsampling operators. Thus the output image's resolution is increased. The contracting paths are combined with upsampled outputs to localise high-resolution features. Then a convolution layer can learn to assemble more precise output based on the information. The architecture looks like an "U" because the expansive path is almost similar to the contracting path. The skip connection is used in each level of U-Net to ensure that the low-level features are present in the feature map [32].

4.4.2.1 Network Architecture

The network architecture is illustrated in Figure 4.7 . It has two paths contracting path (left half) and an expansive path (right half).

1. **Contracting path [32]:** It has a typical architecture of a convolutional network. It is made by repeatedly applying two 3×3 convolutions (unpadded convolutions). Each

one of these is followed by a rectified linear unit (ReLU) (see Section 3.14) and a 2×2 max pooling operation. This operation has a stride 2 and is used for downsampling. The number of feature channels is doubled at each downsampling step.

2. **Expansive path [32]:** Every step consists of the following three operations:
   - Upsampling of the feature map that is followed by a 2x2 convolution ("up-convolution") that halves the number of feature channels.
   - Concatenation of two 3×3 convolutions with the correspondingly cropped feature map from the contracting path is done.
   - ReLU is added to the end of each of them.

   Cropping is necessary to combat the loss of border pixels in every convolution. At the final layer, a 1×1 convolution is used to map each 64-component feature vector to the desired number of classes. In total, the network has 23 convolutional layers [32].

4.4.2.2 Training

The input images and their corresponding segmentation maps are used to train the network using stochastic gradient descent (see Section 3.11). The output image is smaller (by a constant border width) than the input image [32]. The loss used here is dice loss (see Section 3.9). Due to the aforementioned reason, the Batch size is set to one image and the network's momentum is set to 0.99. The higher momentum forces network to base the current optimisation step on a large number of previously seen samples.

4.4.2.3 Data Augmentation

Data augmentation is essential to teach the network the desired invariance and robustness properties when only a few training samples are available. Here this follows the same augmentations that were used in SEAM (see Section 4.1.2.2).

4.4.3 Implementation Details

We are using structure based on the source paper [32] and Pytorch module [92]. Training is done for 45 epochs.

4.5 Active Blood Detection: CV-Based non-learning Approach

The paper titled "Active Blood Detection in a High-Resolution Capsule Endoscopy using Colours Spectrum Transformation." [30] proposes a technique to detect bleeding automatically. The same technique is being used here to perform the task of blood accumulation detection. The technique in our case uses colour spectrum transformation methods and morphological filtering to detect the site of blood accumulation by generating pseudo masks that show the location of the blood accumulation (see Figure 4.8).

4.5.1 Motivation Behind Using This Module

The module is a baseline module that uses classical computer vision techniques without any neural networks. The idea behind including this is to see how well the weakly supervised networks perform and establish a baseline that tells us if they are effective compared to naive non-learning based approaches, given the resource requirements of neural networks are always higher.

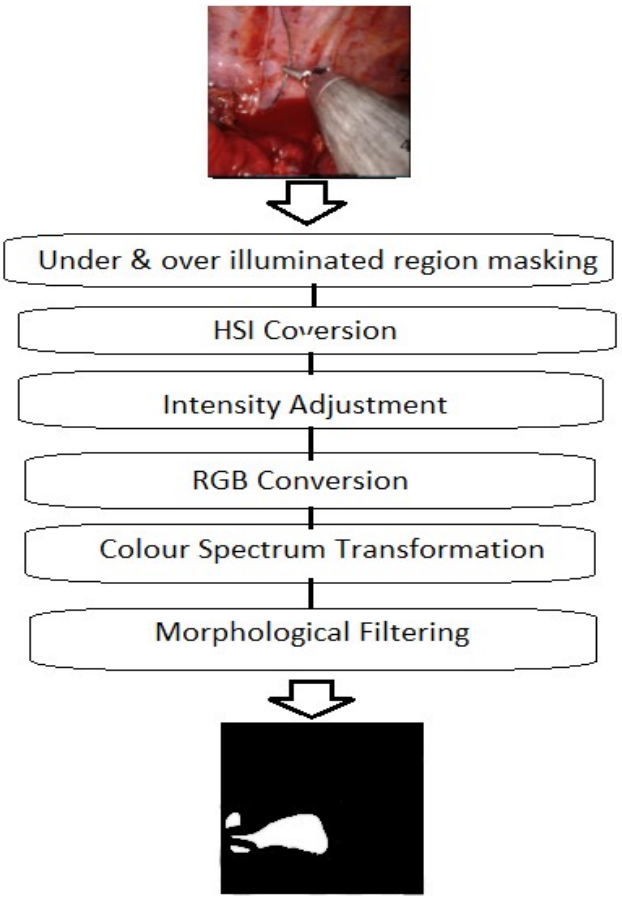

Figure 4.8: non-learning based approach [30]

4.5.2 Method Description

The method has three steps [30]:

- Brightness adjustment
- Colour spectrum transformation
- Morphological filtering

4.5.2.1 Brightness Adjustment

There are two sub steps performed.

1. Under-& Over-Illuminated region masking
   The image captured from the camera can have under or over illuminated regions. This makes distinguishing object of interest in darker regions difficult.

   If,

$$(R_{(i,j)}/(G_{(i,j)} + B_{(i,j)} + d_1))\&(\sqrt{R^2_{(i,j)} + G^2_{(i,j)} + B^2_{(i,j)}} < d_2) \tag{4.20}$$

Then

:dark region masking ($i, j = 0 - 320$ natural number)
if,

$$(\sqrt{R^2_{(i,j)}} > b_1)\&(\sqrt{R^2_{(i,j)} + G^2_{(i,j)} + B^2_{(i,j)}} > b_2) \tag{4.21}$$

:brightness region masking ($i, j = 0 - 320$ natural number)
In the experiment it was empirically determined that $d_1 = 10, d_2 = 63, b_1 = 190, b_2 = 190$ are the most suitable values of the variables.

2. Intensity adjustment
For fine-tuning the brightness, the intensity value is improved. The intensity values are removed backwards and forwards and then whole intensity values are smoothed.

4.5.2.2 Colour Spectrum Transformation

Colour spectrum transformation is used here. The goal is to separate the bleeding region and the normal region. This is done on the results of step 1 (see 4.5.2.1). The high value regions with bleeding are separated by using the Equation:

$$I_{transform} = \kappa_1 R_{(i,j)} / \kappa_2 (R_{(i,j)} + G_{(i,j)} + B_{(i,j)} + p) \tag{4.22}$$

Empirically p is set to 10. After this a boundary value is found that divides the image into two regions of bleeding and normal region. Threshold method is applied to $I_transform$ in order to separate the high pixel value of histogram, the bleeding region from whole $I_transform$ image. Various experiments were performed to find the ideal value for this parameter. The are explained in depth in Section 4.5.2.3

4.5.2.3 Morphological Filtering

There are tiny regions that are not the actual bleeding spots. These isolated spots are considered noise as actual bleeding does not occur as isolated pixels or in a small-sized area. After each white spot is identified, they are filtered out using a predefined threshold. This is implemented in code as shown in Figure 4.9.

```
image = np.apply_along_axis(\
        validate_pixel, -1, image)[:, :, 0].astype(np.uint8)\
        nb_components, output, stats, \
        centroids =cv2.connectedComponentsWithStats(\
        image, connectivity=8)
sizes = stats[1:, -1]nb_components = nb_components - 1
min_size = 8
img2 = np.zeros((image.shape))
for i in range(0, nb_components):
    if sizes[i] >= min_size:
        img2[output == i + 1] = image[output == i + 1]
result = img2
```

Figure 4.9: Code Snippet non-learning approach

See Table 4.1 to get an overview of Hyperparameters. Table 4.2 compares Hyperparameters of all the models at a glance.

d1	d2	b1	b2	min-cluster size
10	63	190	190	8

Table 4.1: Overview of Hyperparameters

model	Backbone Network	Learning Rate	Momentum	Epochs	Optimiser
SEAM	ResNet-38	0.001	0.99	15	adam
Puzzle-CAM	ResNet-50	0.001	nesterov	10	PolyOptimizer(weight decay=1e-4)
Grad-CAM	ResNet-50	0.001	0.99	15	Adam
U-Net	Source Paper Based [92, 32]	0.001	0.99	50	Adam

Table 4.2: Hyperparameters of the different models at a Glance

4.6 Data

All the evaluated models are trained and evaluated on following two datasets:

1. The first dataset contains images with a foreground object (positive) and images without the aforementioned foreground object (negative). The negative images are easy to distinguish from positive images as they do not contain any discernible regions with blood. This dataset is henceforth referred to as the **Simple Negative dataset**. The Positive images in this dataset have one or many regions where blood is accumulated due to active bleeding. The negative images have no such regions where the blood is accumulated and also no regions with active bleeding. The dataset may contain images with surgical instruments in them.

2. Like the first dataset, the second dataset contains both positive and negative images, but the negative image also has samples where the foreground object is hard to identify. The reason is that the negative images have some blood that cannot be considered as blood accumulation. The dataset may contain images with surgical instruments in them. This dataset is henceforth referred to as the **Hard Negative dataset**. The Positive images in this dataset have one or many regions where blood is accumulated due to active bleeding. The negative images have no such regions where the blood is accumulated or sites where there was an incision but no active bleeding or incision sites that are cauterised.

The data is divided into four-folds to perform k-fold cross-validation and the data is split by keeping images of same surgery in one split and ensuring that no images from that

Count	Fold-1	Fold-2	Fold-3	Fold-4
Training(Bloody)	2868	2923	2135	2462
Training(Non-Bloody)	2868	2923	2135	2462
Validation(Bloody Images)	487	432	1220	893
Validation(Non-Bloody)	487	432	1220	893
Surgeries type (Training)	esophagus, thymus	rectum	esophagus, stomach, rectum, pancreas, sigmoid colon	esophagus, rectum, esophagus, large intestine, pretonium, abdominal wall, gallbladder

Table 4.3: Data count for 4-fold Split Information with a list of organs operated on in the surgeries. The counts are the same for both datasets for consistency.

surgery is used in evaluation. The model weights are save and loaded separately for each splits. The original in-house datasets has imbalanced numbers of positive and negative images and when we load the data the ratio of negative and positive images is made 50/50 to avoid the data imbalance problems. These Ground Truth mask are not used during the training phase of all but one neural network (U-Net) which relies on full supervision. These masks are however used to evaluate the performance of the modules in evaluation phase where we compare them to the predicted binary masks. The image-level labels are not used in the evaluation phase as in real-world scenario this information will not be available to the model.

It can be seen in Table 4.3 how the four folds/splits are made, image count and organs involved in surgeries belonging to the training set.

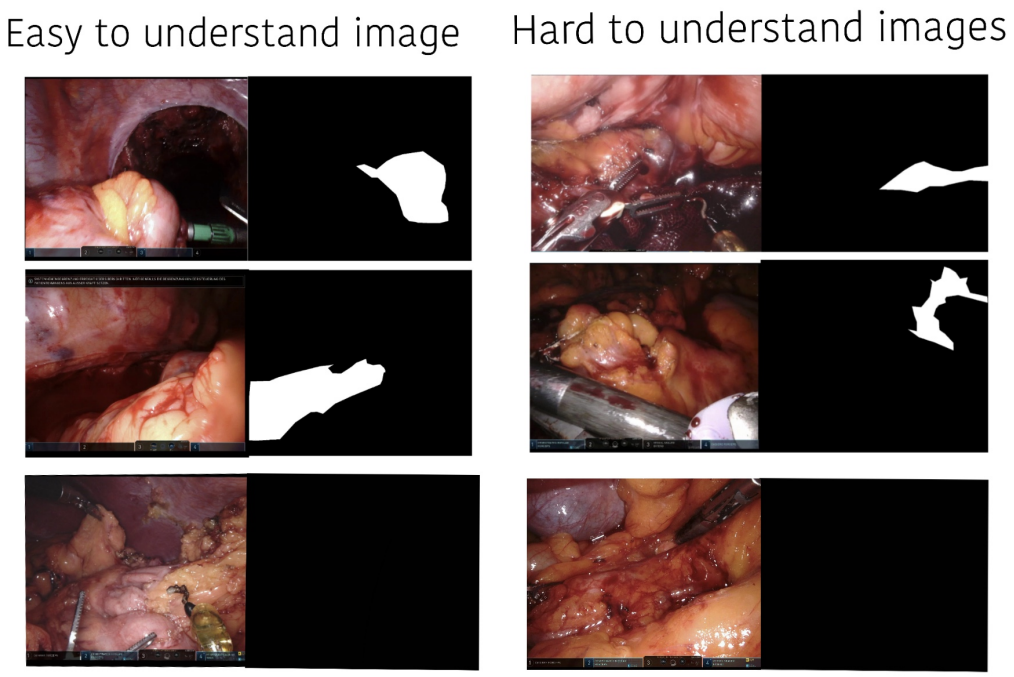

Figure 4.10: Easy and Hard to understand Image samples

The Hard and Simple samples can be seen in Figure 4.10. Figure 4.11 shows sample from the 4 folds we have made. All the surgical images that are present in this work is processed and used according to General Data Protection Regulation [93] and other related Medical Image processing rules.

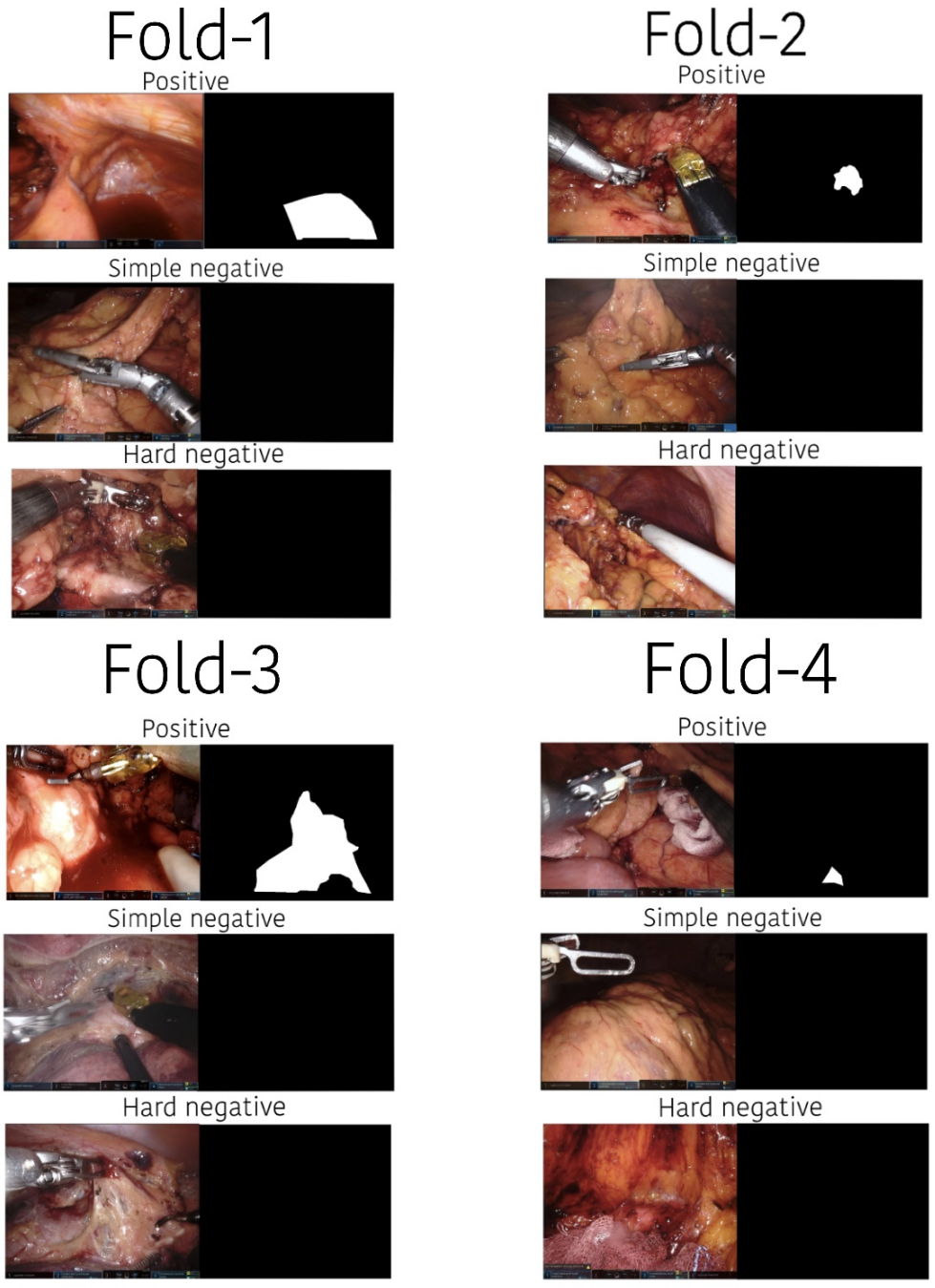

Figure 4.11: Image sample from each folds (**Top**: positive, **Middle**: Simple negative and **Bottom**: Hard negative).

5. Evaluation

A wide range of experiments were conducted using image-level labels to investigate the possibility of using weakly supervised semantic segmentation. We also observe the impact of the inclusion of hard to evaluate negative samples to the standard dataset. In this section, using qualitative and quantitative evaluation, three main questions are answered:

1. How to get a binary segmentation mask from heatmaps (thresholding)?
   - The effect of Otsu versus Fix thresholding methods on the final mask is evaluated empirically through comprehensive experiments.
2. How do weakly supervised approaches compare to supervised or unsupervised approaches?
3. How does the inclusion of Hard negatives affect performance?

Since the real-world application scenario, where we need to identify the rough location of the blood accumulation, does not require these models to produce the exact boundaries of the blood accumulation, the qualitative results may not best represent the module's performance. Therefore a visual examination is also performed in this section, where we compare the predicted masks generated by different models for the same image input. Even though these qualitative results penalise the modules for not predicting the exact boundaries, we gain valuable insights into the modules overall performance in generating segmentation masks. Hence any conclusion based on just quantitative or qualitative observations may be misleading, and this particular issue is briefly discussed here with the aid of Classifier Accuracy. An additional observation about the time taken to compute each image is also stated as this highlights the model's ability to work in time-constrained scenarios (real-time).

5.1 Quantitative Evaluation

First, the performance of the two thresholding methods mentioned above is evaluated on individual modules. Since we have two different datasets (Simple negative, Hard negative), we train the models on both datasets and evaluate them on the validation sets from the respective databases; however, the validation sets are different so we cannot comment on the impact of the inclusion of Hard negatives. To illustrate the impact of the inclusion of Hard negatives, we further evaluate all the models trained on the Hard negative dataset with the validation set of the Simple negative dataset.

We are using four-fold cross-validation, so the results for each fold are reported for the three cases abbreviated as follows:

- **SNSN**=Trained on Simple Negative and Evaluated on Simple Negative.
- **HNSN**=Trained on Hard Negative and Evaluated on Simple Negative.
- **HNHN**=Trained on Hard Negative and Evaluated on Hard Negative.

These abbreviations will be used throughout this thesis report henceforth.
F1-Score is used as the main metric for the module evaluation here as it presents the overall ability of the models to predict a segmentation mask. F1-Score indicates the overlap of the predicted segmentation mask with the Ground Truth mask. Training losses are not reported here as their value does not tell us anything significant about the model.

Over-activation occurs when the pixels that do not belong to the foreground class are also labelled as foreground, and under-activation occurs when the pixels belonging to the foreground class is labelled as the background. To illustrate this, we use Precision and Recall to give us some information about over-activation and under-activation as low precision, but high Recall indicates over-activation. [73]

Accuracy is not a useful measure here as we have two classes, and if the module predicts all the pixels as background, then we will have high accuracy, but we will not see anything in the generated masks. [73]

However, since three weakly supervised models can act as classifiers, we also predict the labels in the image and an accuracy score for classification is reported as well. IOU scores were also calculated but are omitted here as they tend to penalise wrong predictions more and IOU and F1-Scores are always within a factor of 2 $F/2 \leq IoU \leq F$ and they are positively co-related. [94]

To calculate these results TP/TN/FP/FN information is collected for each image and the final matrices values are calculated on these values at the end of evaluation using mean. This is done as we have images where we don't have a foreground and these images will influence and pump up the results in an unnatural manner.

5.1.1 SEAM

A fixed threshold of 72% has been empirically deduced for this model. The three tables report the results of training SEAM on two datasets and performing the aforementioned three experiments.

1. For the SNSN experiment, Otsu thresholding performs better than the fixed threshold; we observe an increase in the F1-Score in folds 1,2,3 but a slight decrease in fold 4. This anomaly likely occurred as fold 4 contains smaller sized Ground Truth regions in comparison to other regions, and the fixed threshold acts as a more strict threshold suppressing more unwanted pixel activations than the dynamically decided less strict Otsu threshold. However, we can conclude from the evidence that the Otsu thresholding method generates satisfactory results.
2. For HNHN, Otsu thresholding performs better than fix thresholding in all four validation folds. This information is not so significant as this only indicates the model's F1-Score when it was trained and evaluated on Hard negative dataset.
3. For HNSN, Otsu thresholding performs better than the fixed threshold of 72%. Other threshold values were also tried to improve the results, but they did not produce any significant improvements.

When SNSN and HNSN are compared, it is observed that the inclusion of Hard negatives for folds 2 and 3 significantly improved the F1-Score, but for folds 1 and 4, the inclusion of Hard negatives proved detrimental. It is hypothesised that this behaviour is due to the smaller size of the truth region and decreased classification accuracy. These observations can be seen in Tables 5.1, 5.2 and 5.3

Trained on Simple Negative and Evaluated on Simple Negative.								
	Precision		Recall		F1		IOU	
Folds	otsu	fix	otsu	fix	otsu	fix	otsu	fix
1	**0.11582**	0.04812	**0.9045**	0.6212	**0.2053**	0.08932	**0.1144**	0.0467
2	**0.01823**	0.00356	**0.0737**	0.0272	**0.0292**	0.00631	**0.0148**	0.0031
3	**0.11695**	0.04980	**0.8335**	0.5044	**0.2051**	0.09066	**0.1142**	0.0474
4	0.25104	**0.30194**	0.9520	**0.9643**	0.3973	**0.45990**	0.2479	**0.2986**

Table 5.1: Training on Simple Negative on Evaluation on Simple Negative

Trained on Hard Negative and Evaluated on Hard Negative.								
	Precision		Recal		F1		IOU	
Folds	otsu	fix	otsu	fix	otsu	fix	otsu	fix
1	0.0747	**0.0892**	0.7726	**0.8364**	0.1363	**0.1612**	0.0731	**0.0876**
2	**0.160629**	0.1568	**0.9464**	0.573454	**0.2746**	0.246263	**0.1554**	0.137
3	**0.1935**	0.15479	**0.9006**	0.9767	**0.3186**	0.2672	**0.1895**	0.133
4	**0.0502**	0.0443	**0.7497**	0.6936	**0.0941**	0.0833	**0.0494**	0.0435

Table 5.2: Training on Hard Negative on Evaluation on Hard Negative

Trained on Hard Negative and Evaluated on Simple Negative.								
	Precision		Recall		F1		IOU	
Folds	otsu	fix	otsu	fix	otsu	fix	otsu	fix
1	**0.0905**	0.0274	**0.8363**	0.3768	**0.1634**	0.0511	**0.0890**	0.0262
2	**0.1570**	0.1522	**0.9463**	0.9006	**0.2694**	0.2604	**0.1557**	0.1497
3	**0.1928**	0.1255	**0.9005**	0.7737	**0.3176**	0.2160	**0.1888**	0.1211
4	**0.0507**	0.01700	**0.7497**	0.3537	**0.0949**	0.03245	**0.0498**	0.0164

Table 5.3: Training on Hard Negative on Evaluation on Simple Negative

5.1.2 Grad-CAM

A fixed threshold of 40% has been empirically deduced for this model. Throughout the three experiments, it is observed that precision is significantly higher than other weakly supervised models and the unsupervised method. The classifier accuracy is also higher than other weakly supervised models and the unsupervised method. The three tables report the results of training Grad-CAM on two datasets and performing the aforementioned three experiments. These observations can be seen in Tables 5.4, 5.5 and 5.6

1. For SNSN, Otsu thresholding performs better than the fixed threshold on all folds. There are no anomalies.
2. For HNHN, Otsu thresholding performs better than the fixed threshold on all folds. There are no anomalies.
3. For HNSN, Otsu thresholding performs better than the fixed threshold on all folds. Except fold 3, we see an increase in the F1-Score, indicating that the introduction of Hard negatives helps with localising the blood accumulation region.

Trained on Simple Negative and Evaluated on Simple Negative.								
	Precision		Recall		F1		IOU	
Folds	Otsu	Fix	Otsu	Fix	Otsu	Fix	Otsu	Fix
1	0.250445	**0.254017**	**0.322908**	0.319768	**0.3213**	0.2831	**0.165**	0.1415
2	**0.329285**	0.321402	0.335454	**0.338192**	**0.3324**	0.3295	**0.166**	0.164
3	**0.298322**	0.291148	**0.336095**	0.321536	**0.3160**	0.3055	**0.158**	0.176
4	**0.282638**	0.282198	**0.318593**	0.318159	**0.2995**	0.2991	**0.150**	0.150

Table 5.4: Grad-Cam Results for both Otsu and Fix Thresholding, Trained on Simple Negatives and evaluate on Simple Negatives

Trained on Hard Negative and Evaluated on Hard Negative.								
	Precision		Recall		F1		IOU	
Folds	Otsu	Fix	Otsu	Fix	Otsu	Fix	Otsu	Fix
1	**0.258913**	0.251731	0.307301	**0.312526**	**0.2810**	0.2788	**0.140**	0.139
2	**0.372377**	0.372377	**0.331359**	0.312526	**0.3625**	0.3506	**0.181**	0.175
3	0.280157	**0.288621**	**0.334420**	0.331826	0.3048	**0.3087**	0.152	**0.167**
4	**0.297494**	0.288621	0.327979	**0.331826**	**0.3119**	0.3087	**0.155**	0.154

Table 5.5: Grad-Cam Results for both Otsu and Fix Thresholding, Trained on Hard Negatives and evaluated on Hard Negatives

Trained on Hard Negative and Evaluated on Simple Negative.								
	Precision		Recall		F1		IOU	
Folds	Otsu	Fix	Otsu	Fix	Otsu	Fix	Otsu	Fix
1	**0.314755**	0.264999	**0.329050**	0.317041	**0.3217**	0.2886	**0.1605**	0.1443
2	**0.376408**	0.375888	**0.343478**	0.339231	**0.3591**	0.3566	**0.1795**	0.1780
3	0.285980	**0.290386**	0.330677	**0.339188**	0.3067	**0.3128**	0.153	**0.1605**
4	**0.292036**	0.282803	0.317490	**0.318906**	**0.3042**	0.2997	**0.1521**	0.1498

Table 5.6: Grad-Cam Results for both Otsu and Fix Thresholding, Trained on Hard Negatives and evaluated on Simple Negatives

5.1.3 Puzzle-CAM

Otsu Thresholding is the only viable approach for thresholding here because when a fixed threshold value is used, in a significant section of validation, all pixels are classified as background. This happens because there is a considerable difference in generated prediction masks. It is further observed that using Group Normalisation instead of Batch Normalisation improves the F1-Score of the module on first and second folds however it proves to be detrimental for third and fourth folds. For the fourth fold, both Batch and Group Normalisation based neural networks do not generate any results that can be considered viable, and the model fails even to localise the foreground class. These observations can be seen in Tables 5.7, 5.8 and 5.9. It is hypothesised that the aforementioned behaviour is because of three major factors:

1. The model cannot classify whether the image has blood accumulation or not.

2. The third and fourth fold have small-sized ground truth regions, and when the Puzzle-CAM shuffles the CAM to manipulate the attention of the network due to the small-sized Ground Truth region, the attention is distributed in other regions of the image. Introducing Hard negatives somewhat alleviates the issue but does not provide any significant improvements. This can also be seen in the qualitative evaluation in Section 5.2

3. Given the current hardware limitations, the model may require more training epochs or bigger batch size. The original paper uses much larger ResNet-101 and ResNet-269 with a batch size of 32.

4. In the case of the Batch Normalisation based variant, the poor performance can possibly also be attributed to the small batch size of 6.

5. Another hypothesis is that introduction of Group Normalisation is not so beneficial as the network architecture is still dependent on Batch Normalisation, and we have to repeat architecture search or do weight standardisation as the following papers Liu et al.[95] and Brock et al.[95, 96] do to see any significant changes.

Trained on Simple Negative and Evaluated on Simple Negative.								
	Precision		Recall		F1		IOU	
Folds	Batch	Group	Batch	Group	Batch	Group	Batch	Group
1	**0.1149**	0.0653	0.5025	**0.6511**	**0.187**	0.1187	**0.1031**	0.0631
2	**0.1346**	0.0304	**0.5029**	0.0457	**0.2124**	0.0365	**0.1188**	0.0186
3	**0.1236**	0.0828	**0.505**	0.0674	**0.1986**	0.1121	**0.1103**	0.0674
4	**0.0162**	0.0096	**0.4923**	0.00854	**0.0314**	0.0090	0.0159	**0.0040**

Table 5.7: Puzzle Results for both Group and Batch Normalisation based architectures, Trained on Simple Negatives and evaluated on Simple Negatives

Trained on Hard Negative and Evaluated on Hard Negative.								
	Precision		Recall		F1		IOU	
Folds	Batch	Group	Batch	Group	Batch	Group	Batch	Group
1	0.0066	**0.1267**	0.2378	**0.4691**	0.0128	**0.1995**	0.0064	**0.1108**
2	**0.19**	0.1398	0.4988	**0.5356**	**0.2752**	0.2218	**0.1596**	0.1247
3	0.021	**0.0226**	**0.3942**	0.0628	**0.0399**	0.0332	**0.0204**	0.0169
4	**0.0213**	0.0001	**0.3918**	0.0	**0.0404**	0.0	**0.0206**	0.0

Table 5.8: Puzzle Results for both Group and Batch Normalisation based architectures, Trained on Hard Negatives and evaluated on Hard Negatives

Trained on Hard Negative and Evaluated on Simple Negative.								
	Precision		Recall		F1		IOU	
Folds	Batch	Group	Batch	Group	Batch	Group	Batch	Group
1	0.0118	**0.0903**	**0.4207**	0.1593	0.0229	**0.1152**	0.0116	**0.0611**
2	**0.1933**	0.0792	0.5011	**0.7166**	**0.279**	0.1426	**0.1621**	0.0768
3	**0.1368**	0.0400	**0.5007**	0.0592	**0.215**	0.0477	**0.1204**	0.0244
4	**0.0506**	0.0502	**0.5053**	0.05000	**0.092**	0.0090	**0.0482**	0.0444

Table 5.9: Puzzle Results for both Group and Batch Normalisation based architectures, Trained on Hard Negatives and evaluated on Simple Negatives

5.1.4 U-Net

This model is fully supervised and as such do not require thresholding. This model does not make image-level predictions. This model was trained using pixel-level labels to provide full supervision, and the three experiments similar to weakly supervised semantic segmentation were performed.

It is observed that the inclusion of Hard negatives is detrimental to the model's F1-Score. The model performs significantly worse even when we train and evaluate on the Hard negative dataset. When we train on the Hard negative dataset and evaluate on Simple negative dataset, then also we see a drop in F1-Score. Based on the papers Faghri et el.[97] and Xia et el.[98] it can be hypothesised that the inclusion of Hard negatives creates a scenario where the loss function is dominated by negative results generated by Hard negative samples and thus gets stuck in local minima. This problem is probably being further accentuated by the fact that we have a class imbalance in the images. Which when combined with information from Hard negative samples, depending on the ratio of Hard negatives to Simple negative samples, focuses the model's attention more towards the background resulting in more regions of the image being falsely classified as background or foreground and thus reducing the overall F1-score of the model. These observations can be seen in Tables 5.10, 5.11 and 5.12

Trained on Simple Negative and Evaluated on Simple Negative.				
Folds	Precision	Recall	F1	IOU
1	0.6972	0.6475	0.67148	0.5054
2	0.7406	0.7147	0.7274	0.5716
3	0.6171	0.5914	0.6040	0.4327
4	0.6250	0.5834	0.6035	0.4321

Table 5.10: Evaluation Results of U-Net approach. Trained on Simple Negative and Evaluated on Simple Negative

Trained on Hard Negative and Evaluated on Hard Negative.				
Folds	Precision	Recall	F1	IOU
1	0.6333	0.6628	0.6477	0.4790
2	0.6747	0.6883	0.6814	0.5168
3	0.4892	0.7254	0.5843	0.4128
4	0.6220	0.5494	0.5835	0.4119

Table 5.11: Evaluation Results of U-Net approach. Trained on Hard Negative and Evaluated on Hard Negative

Trained on Hard Negative and Evaluated on Simple Negative.				
Folds	Precision	Recall	F1	IOU
1	0.5952	0.6628	0.6272	0.4569
2	0.665	0.6883	0.6766	0.5113
3	0.4840	0.7254	0.5806	0.4090
4	0.6053	0.5494	0.5760	0.4045

Table 5.12: Evaluation Results of U-Net approach. Trained on Hard Negative and Evaluated on Simple Negative

5.1.5 CV-Based Approach

This model also serves as a baseline to evaluate the performance of weakly supervised semantic segmentation. Since there is no training involved, both dataset's validation sets are passed to this algorithm to generate masks. From Table 5.13 it is observed that Otsu's thresholding performs better than the empirically determined best performing threshold of 70 % and removing any cluster groups smaller than 8 pixels. There is no significant difference between evaluations on both datasets.

Simple Negative Data								
	Precision		Recall		F1		IOU	
Folds	fix	Otsu	fix	Otsu	fix	Otsu	fix	Otsu
1	0.0438	0.0738	0.3007	0.5088	0.0875	**0.1289**	0.0458	0.0689
2	0.1236	0.0196	0.2691	0.1486	**0.2471**	0.0347	0.1410	0.0177
3	0.0780	0.0618	0.3197	0.3825	**0.1559**	0.1064	0.0845	0.0562
4	0.0355	0.0436	0.2994	0.4720	0.0710	**0.0799**	0.0368	0.0416
Hard Negative Data								
1	0.0399	0.0769	0.2836	0.5088	0.0699	**0.1336**	0.0416	0.0716
2	0.1215	0.0201	0.2658	0.1486	**0.1667**	0.0354	0.1382	0.0180
3	0.0760	0.0622	0.3517	0.3825	**0.1249**	0.1070	0.0822	0.0565
4	0.0339	0.0437	0.4075	0.4720	0.0625	**0.0800**	0.0351	0.0417

Table 5.13: Evaluation Results of CV-based approach

5.1.6 Comparative Quantitative Analysis

It is observed in the previous section that Otsu's Threshold gives us the best results, so here we compare the F1-Scores of all the models on SNSN and HNHN sets and report the score in Table 5.14. Classification Accuracy and time taken to Process each image during the evaluation phase is also shown in Tables 5.15 and 5.16 to discuss the real-world relevance of the model. We observe that all weakly supervised models perform better than the CV (active blood detection) approach. These models, however, do not perform as well as fully supervised U-Net. Among the weakly supervised models, Grad-CAM performs the best but still lacks the ability to generate full resolution images. On the other hand, SEAM performs slightly worse than Grad-CAM but generates full resolution images. The reduced performance is likely due to lower classification accuracy than Grad-CAM. Puzzle-CAM does not perform satisfactorily on folds 3 and 4, resulting in an overall low score. It is evident from the observations that the inclusion of Hard negatives helps with improving foreground localisation and better boundary predictions. However, accurate boundaries are not highly relevant in the real world implementation, and thus we perform a qualitative evaluation in the next section.

A further hypothesis can be formed from the observations above that all weakly supervised models' poor or unsatisfactory performance can be attributed to Information Bottleneck. This hypothesis is based on the paper by Lee et al.[99] and builds its argument on the premise that even though the final feature map of the classifier is rich in information, the final classification layer filters out most of it. Since CAM only includes information processed by the final classification weight thus, the CAM cannot identify the entire area of the target object. Table 5.14 shows condensed results and Table 5.15 explains a single value to compare the results and provide classification accuracy and image processing time.

Evaluation Results (F1)								
	V1		V2		V3		V4	
models	SNSN	HNSN	SNSN	HNSN	SNSN	HNSN	SNSN	HNSN
SEAM	0.2053	0.1634	0.0292	0.2694	0.2051	0.3176	0.3973	0.0949
Grad-CAM	0.3213	**0.3217**	0.3324	**0.3591**	**0.3160**	0.3067	0.2995	**0.3042**
Puzzle-CAM	0.187	0.0229	0.2124	0.279	0.1986	0.215	0.0314	0.092
U-Net	0.67148	0.6272	0.7274	0.6766	0.6040	0.5806	0.6035	0.5760

Table 5.14: Condensed training results to compare the impact of the inclusion of Hard negatives during training and performance of each network in contrast to supervised segmentation.

model	F1			Image Processing Time	Accuracy of Classifier		
	SNSN	HNSN	HNHN		SNSN	HNSN	HNHN
SEAM	0.2092	**0.2113**	0.2059	2.06s /Image	69.69	**70.04**	69.37
Grad-CAM	0.3479	**0.3548**	0.3107	1.92s/Image	79.82	**83.21**	75.35
Puzzle-CAM	0.0920	**0.1522**	0.0961	2.42s /Image	**50.61** (B),48.81(G)	49(B),48.74(G)	47.97(B),49.87(G)
U-Net	**0.6515**	0.6151	0.6242	1.09s/Image	n.a	n.a	n.a
CV	0.0874	**0.0890**	0.0881	7.62s/Image	n.a	n.a	n.a

Table 5.15: Single Values from 4-fold to unclutter the visualisation of the data. We can clearly see that introduction of Hard negatives have a slight impact on the overall performance as it increases the performance. B and G are abbreviations for Batch and Group Normalisation.

module	Training Time for each module
SEAM	15hr/fold
Grad-CAM	50hr/fold
Puzzle-CAM	12hr/fold
U-Net	8hr/fold
CV	n.a

Table 5.16: Training Time for each module

5.2 Qualitative Evaluation

This section presents various snippets of masks produced by every module on HNHN, SNSN, and HNSN for four-folds. The samples contain a few images with foreground class and a few samples of images without foreground class. We evaluate the performance of the modules and the shortcomings of each module individually. We can also visually ascertain the impact of using Otsu thresholding. This section aims to see if the module is viable for the real-world scenario. Grad-CAM has the best localisation ability, while SEAM produces boundary shapes close to the Ground Truth; however, they both fall short and does not perform as well as U-Net. The masks generated from Grad-CAM seems viable for utilisation. Since Hard negatives give the best score in the weakly supervised approach on F1-Scores, all the results here are from the HNSN experiment. In the case of U-Net, results are from SNSN, as the introduction of Hard negatives reduces the F1-Scores. These particular experiments are chosen to present the best-case scenario.

5.2.1 SEAM

It is observed that Otsu Threshold performs better than the Static Thresholds in all the folds. We see that there is still significant background pixels labelled as foreground

pixels, and also in fold-3; it can be seen that the predicted mask has regions that belong to foreground being labelled as background. Furthermore, it is observed that there are instances where SEAM fails to distinguish containing blood from a bloodless image. We also observe the results of HNHN in Image 5.5. The results for SNSN and HNSN can be seen in 5.1, 5.2,5.3,5.4.

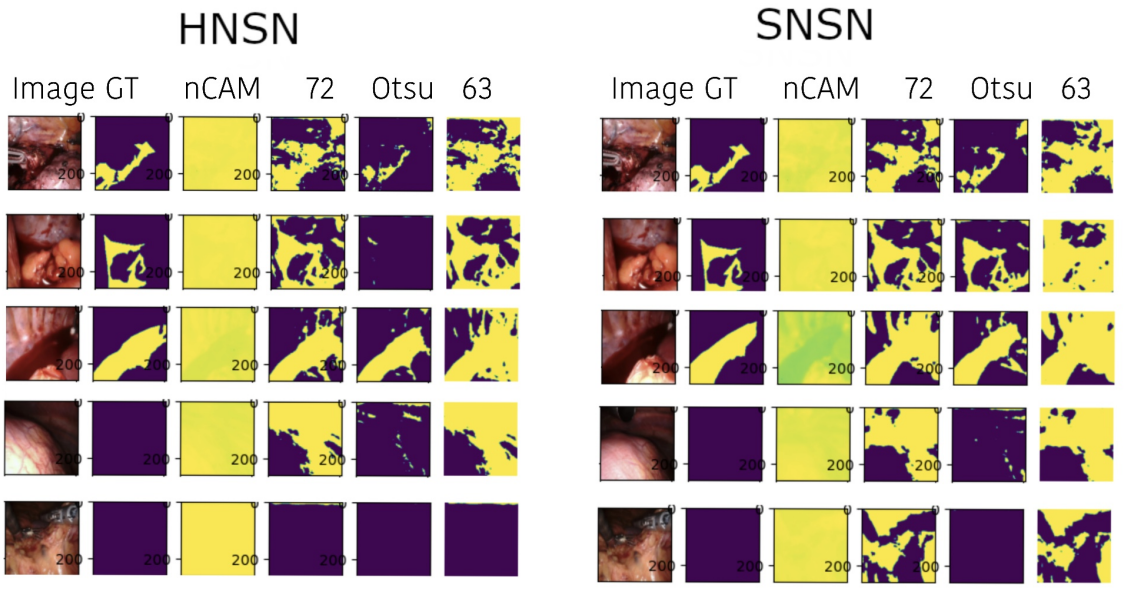

Figure 5.1: SEAM results for Fold-1

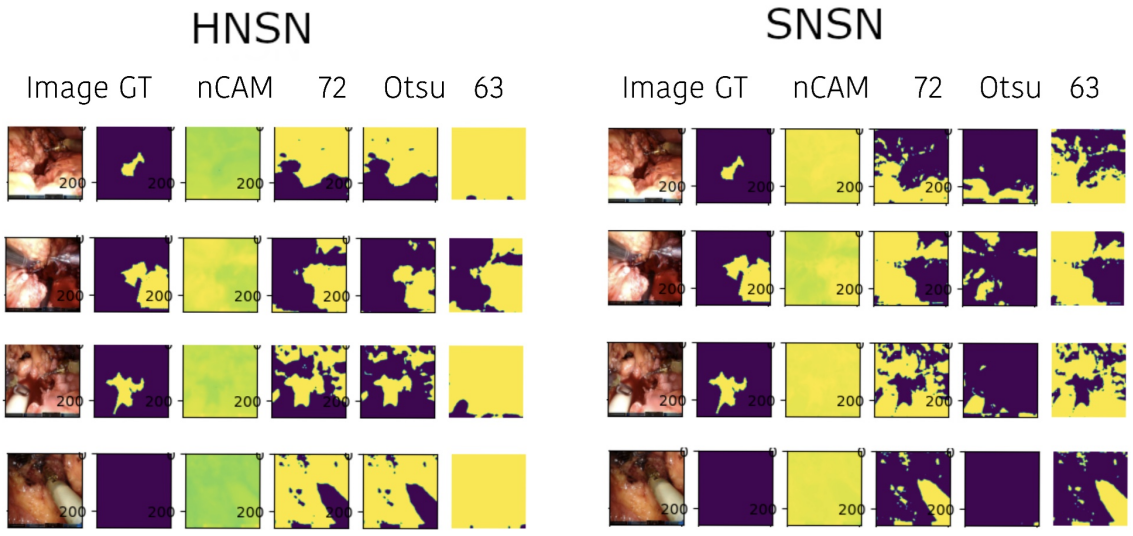

Figure 5.2: SEAM results for Fold-2

Figure 5.3: SEAM results for Fold-3

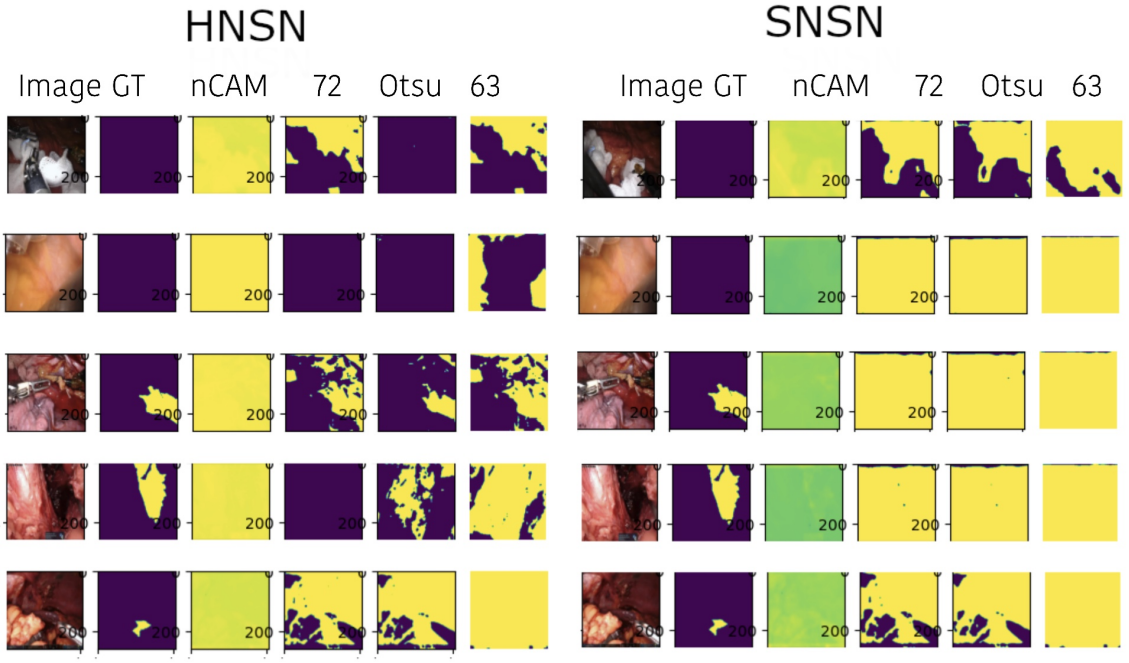

Figure 5.4: SEAM results for Fold-4

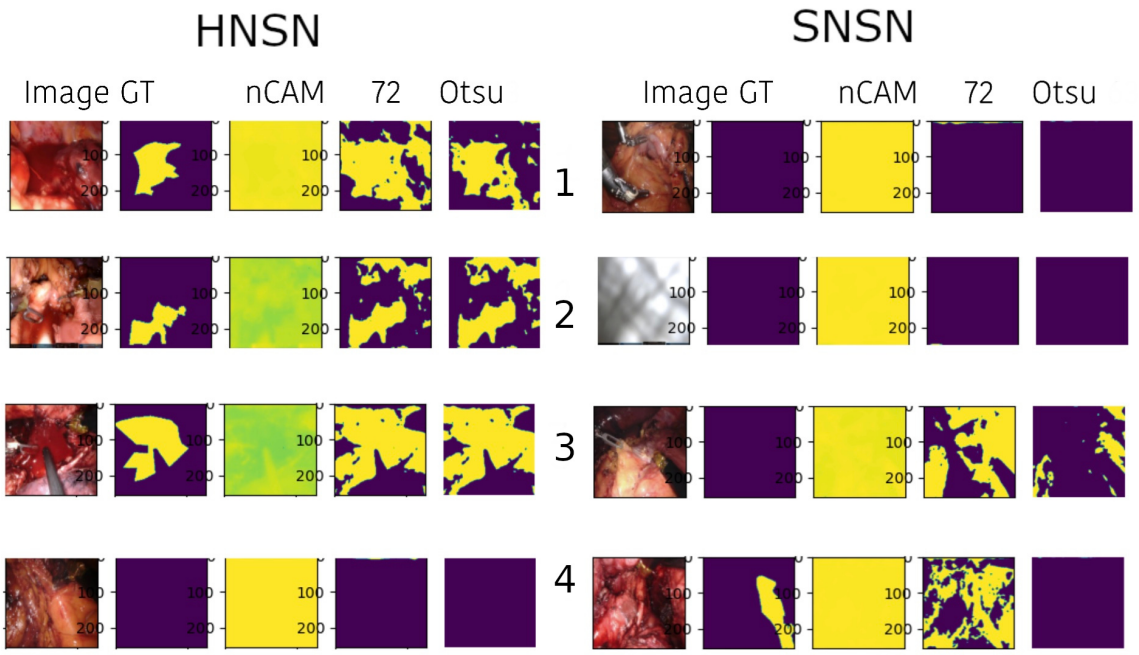

Figure 5.5: SEAM results for HNHN on all folds

5.2.2 Grad-CAM

From a qualitative perspective, the masks generated by Grad-CAM are similar to SEAM but have the resolution of the last layer of the neural network to which Grad-CAM was attached. It can be observed that Grad-CAM performs an excellent job of localisation and even predict similar boundaries to SEAM. We also observe the results of HNHN in Image 5.10. The results for SNSN and HNSN can be seen in 5.6, 5.7,5.8,5.9.

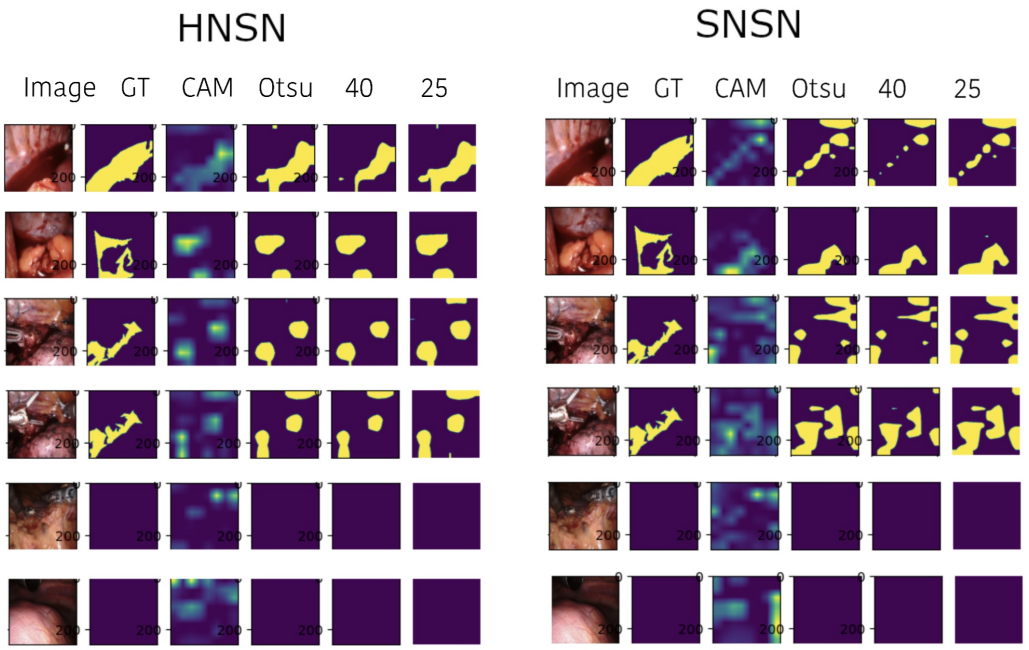

Figure 5.6: Grad-CAM results on fold-1

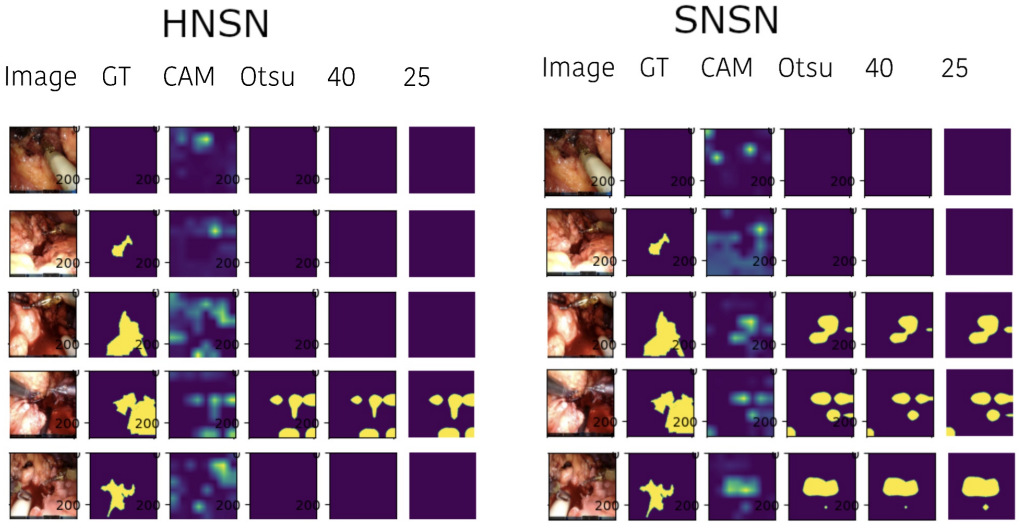

Figure 5.7: Grad-CAM results on fold-2

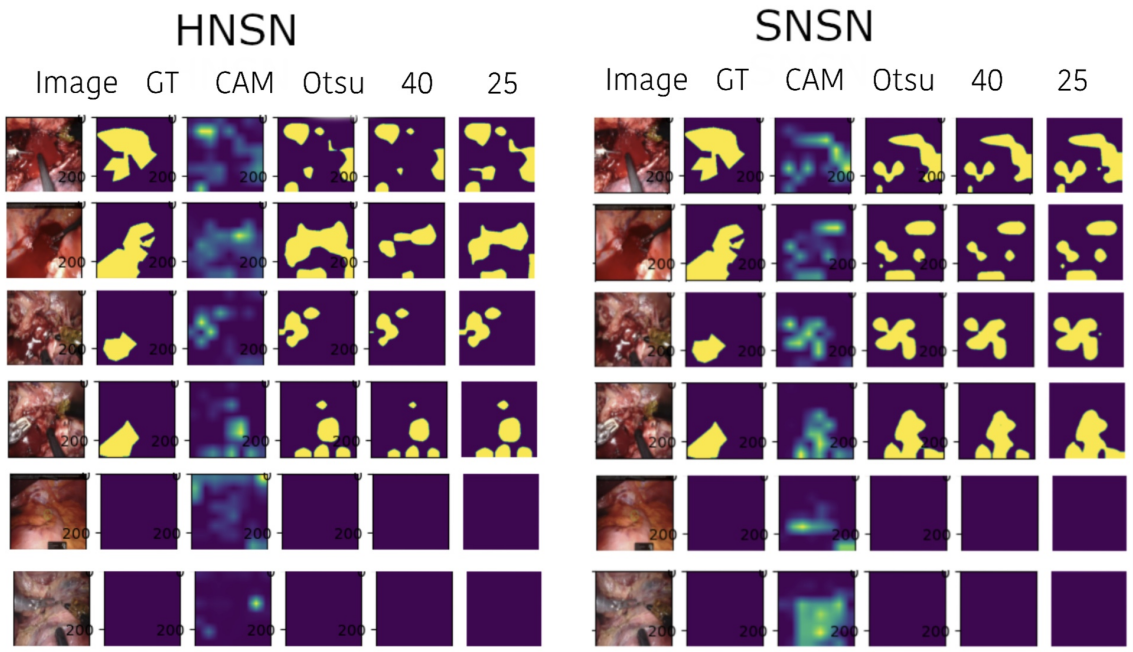

Figure 5.8: Grad-CAM results on fold-3

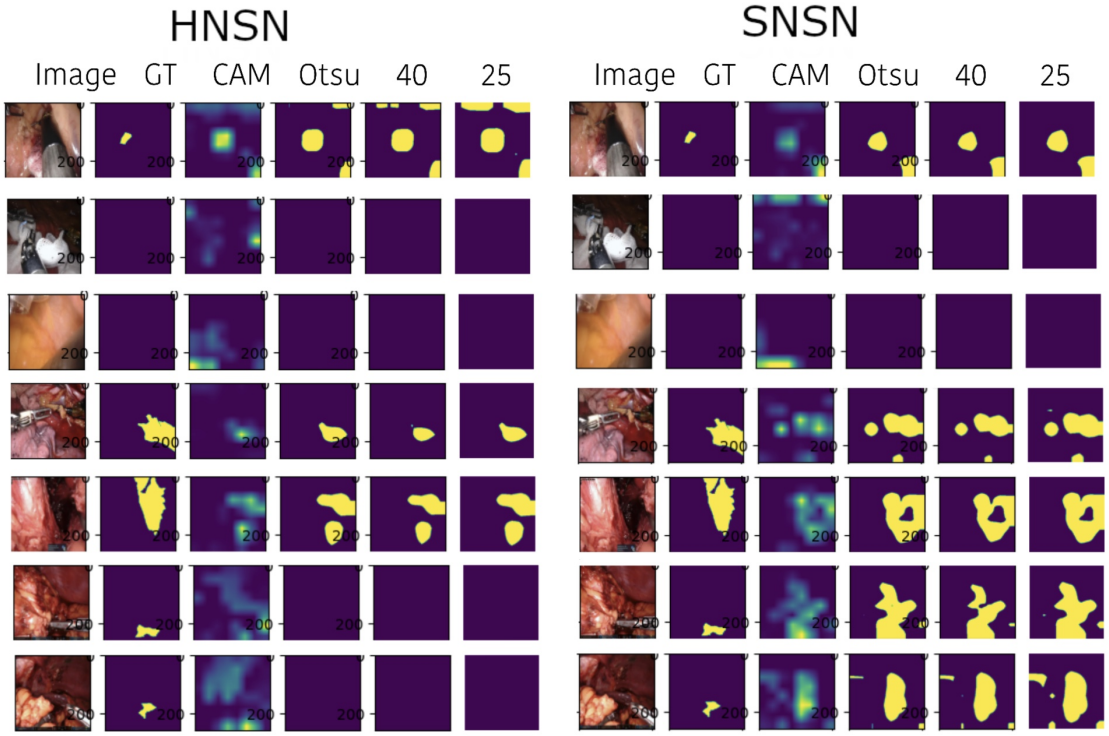

Figure 5.9: Grad-CAM results on fold-4

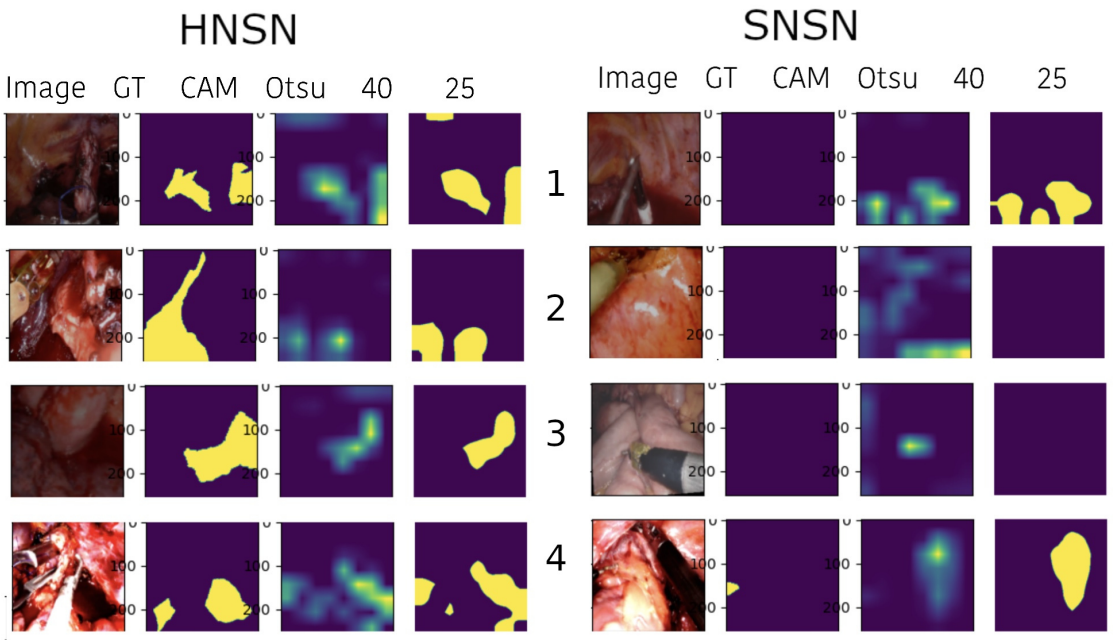

Figure 5.10: Grad-CAM results for HNHN on all folds

5.2.3 Puzzle-CAM

We observe that in folds 1 and 2, we get good instances of proper localisation and boundaries, but in folds 3 and 4, we only see a point or just some isolated island (most discriminative regions). These observations are in stark contrast to the observations mentioned in the source paper. For folds 1 and 2 Group Normalisation based model performs a satisfactory job; however, for folds 3 and 4, we do not have viable/satisfactory results. We also observe the results of HNHN in Image 5.15. The results for SNSN and HNSN can be seen in 5.11, 5.12,5.13,5.14.

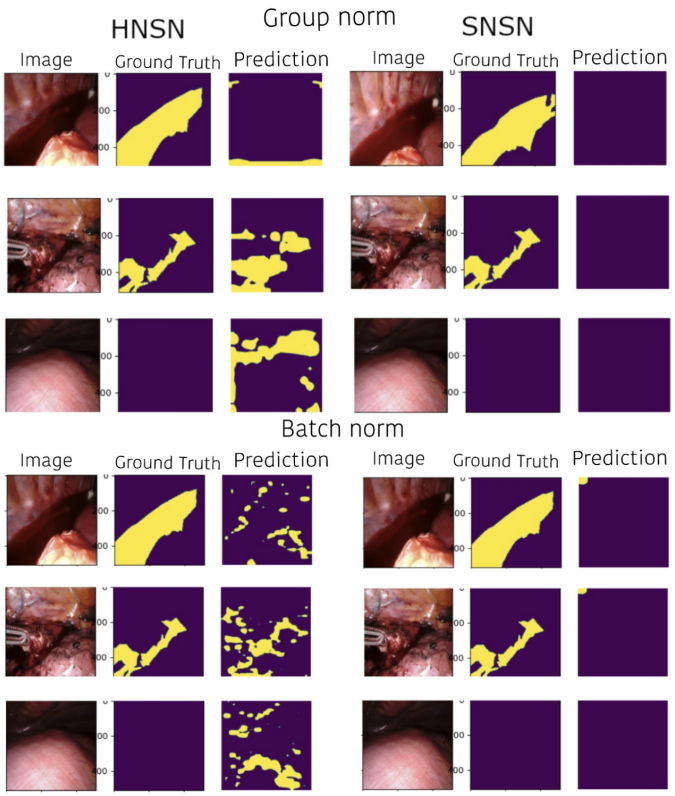

Figure 5.11: Puzzle-CAM results on fold-1

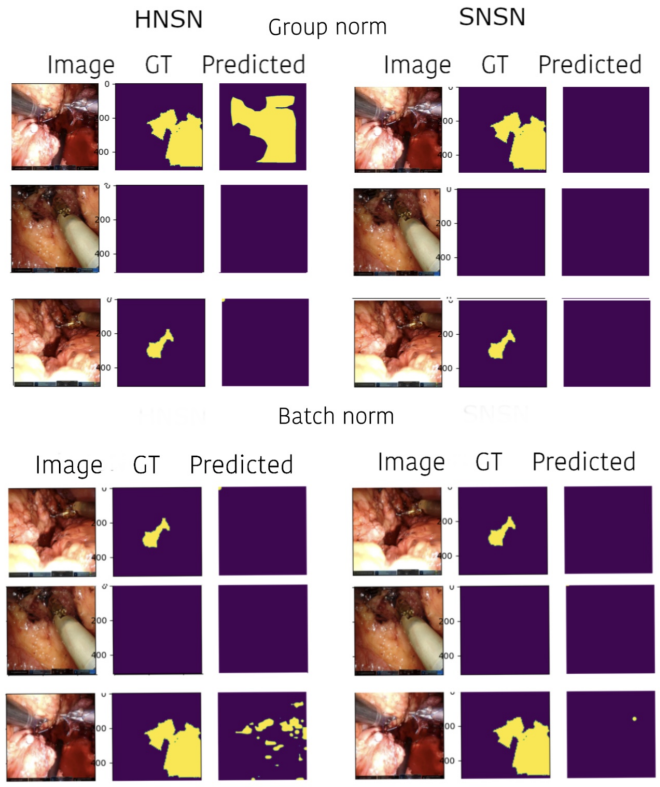

Figure 5.12: Puzzle-CAM results on fold-2

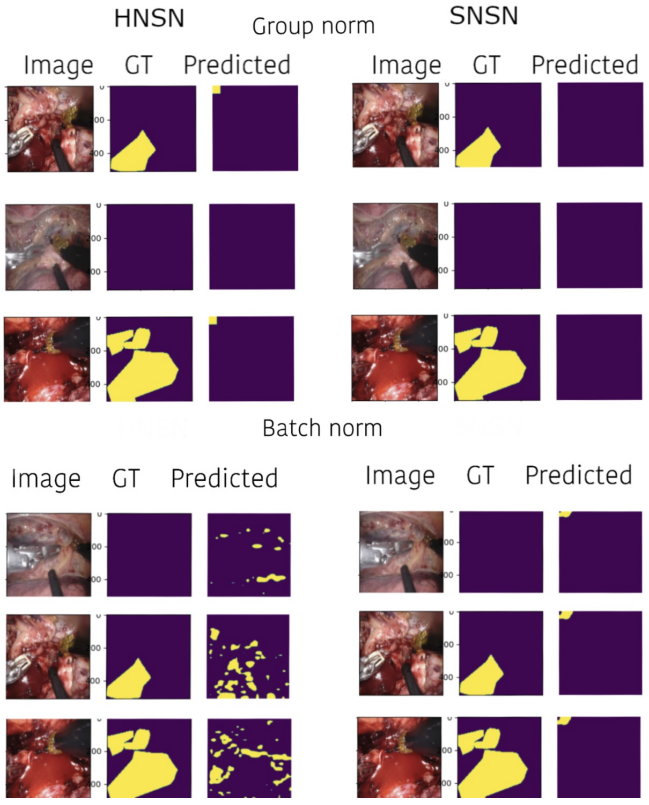

Figure 5.13: Puzzle-CAM results on fold-3

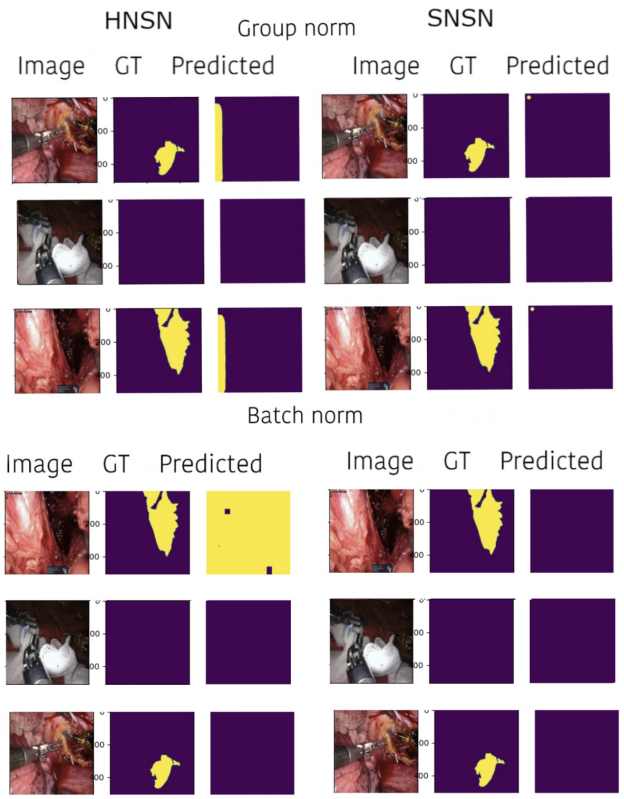

Figure 5.14: Puzzle-CAM results on fold-4

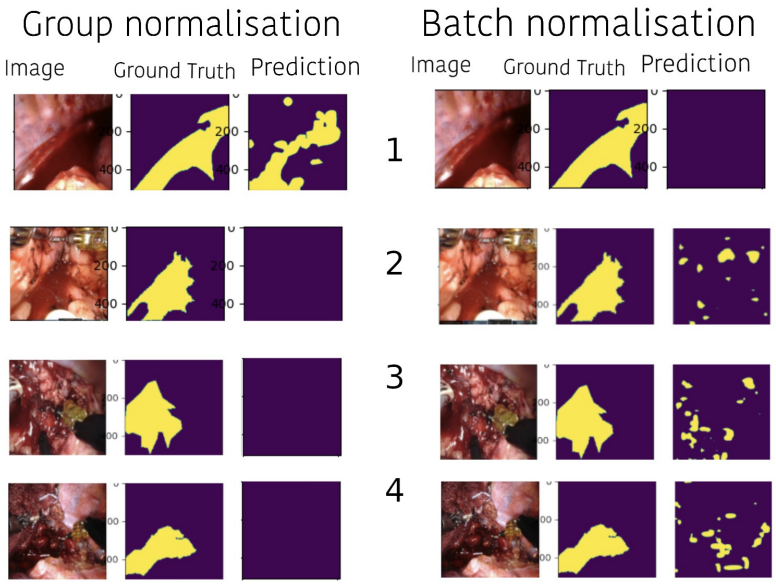

Figure 5.15: Puzzle-CAM results for HNHN on all folds

5.2.4 U-Net

The results are excellent for all the folds, but some under and over-activation is seen in folds 3 and 4. When trained on Hard negative and evaluated on Simple negative, the model produces significantly bad results. These results corroborate the quantitative observations we have made. We also observe the results of HNHN in Image 5.20. The results for SNSN and HNSN can be seen in 5.16, 5.17,5.18,5.19.

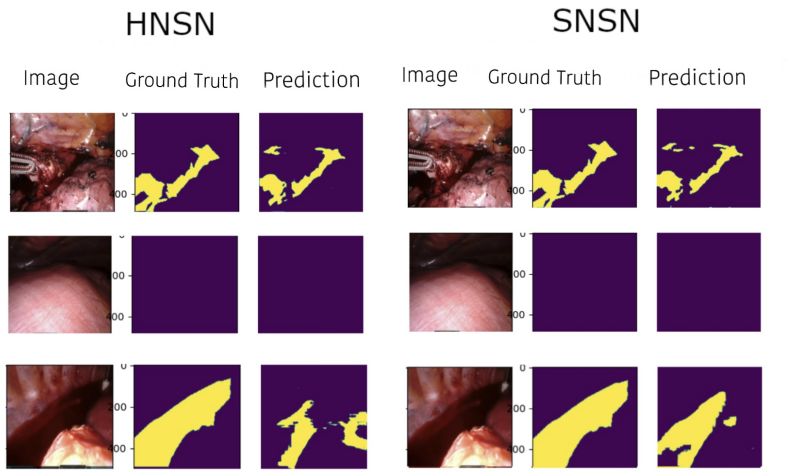

Figure 5.16: U-Net results on fold-1

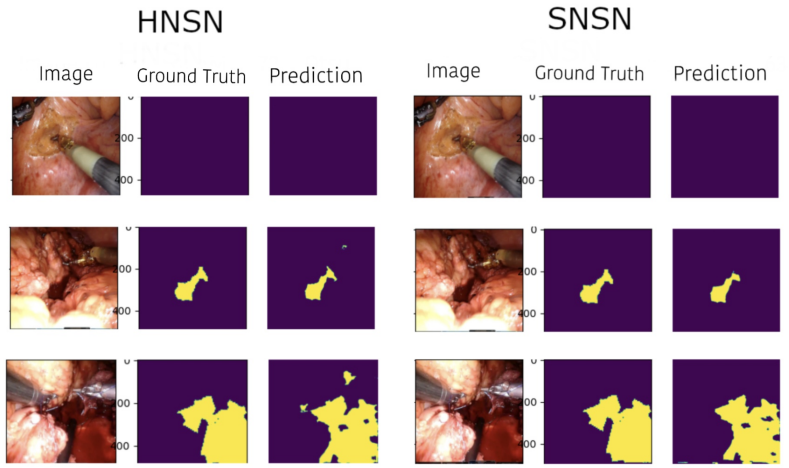

Figure 5.17: U-Net results on fold-2

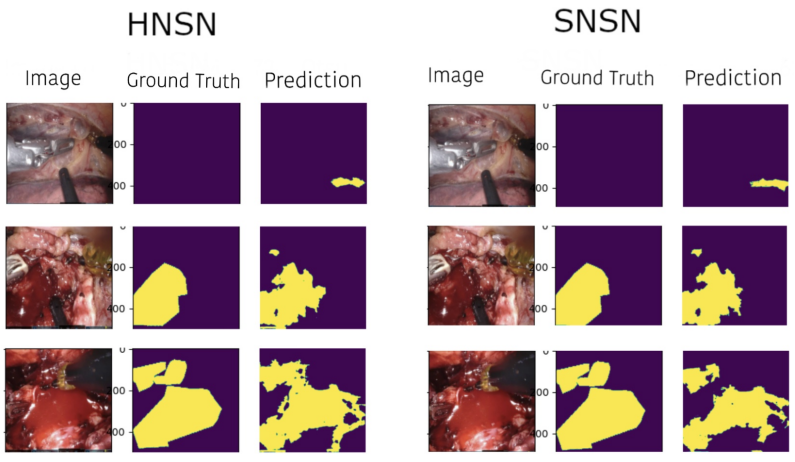

Figure 5.18: U-Net results on fold-3

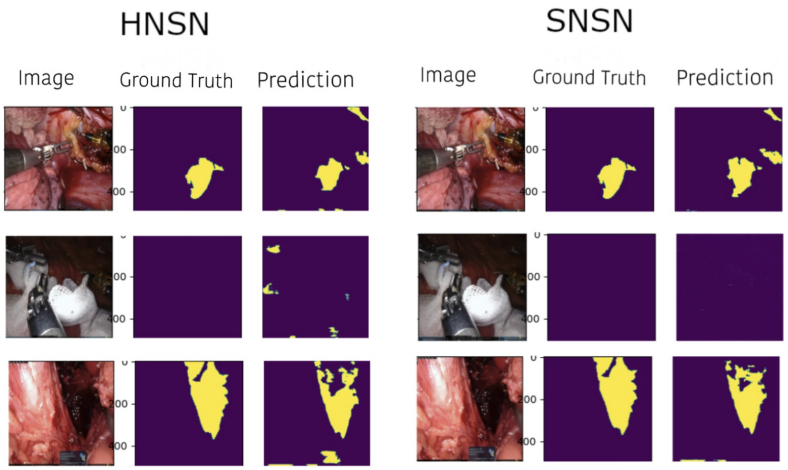

Figure 5.19: U-Net results on fold-4

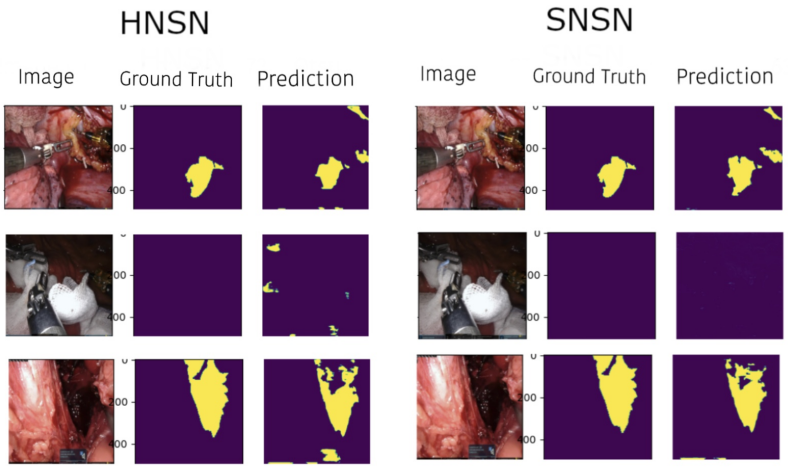

Figure 5.20: U-Net results for HNHN on all folds

5.2.5 Non-Learning-Based Approach

Since there is no training performed, we can see that there is a significant amount of over and under-activation and also, there are instances where muscle fibres are considered regions with blood accumulation. The results can be seen in Image 5.21

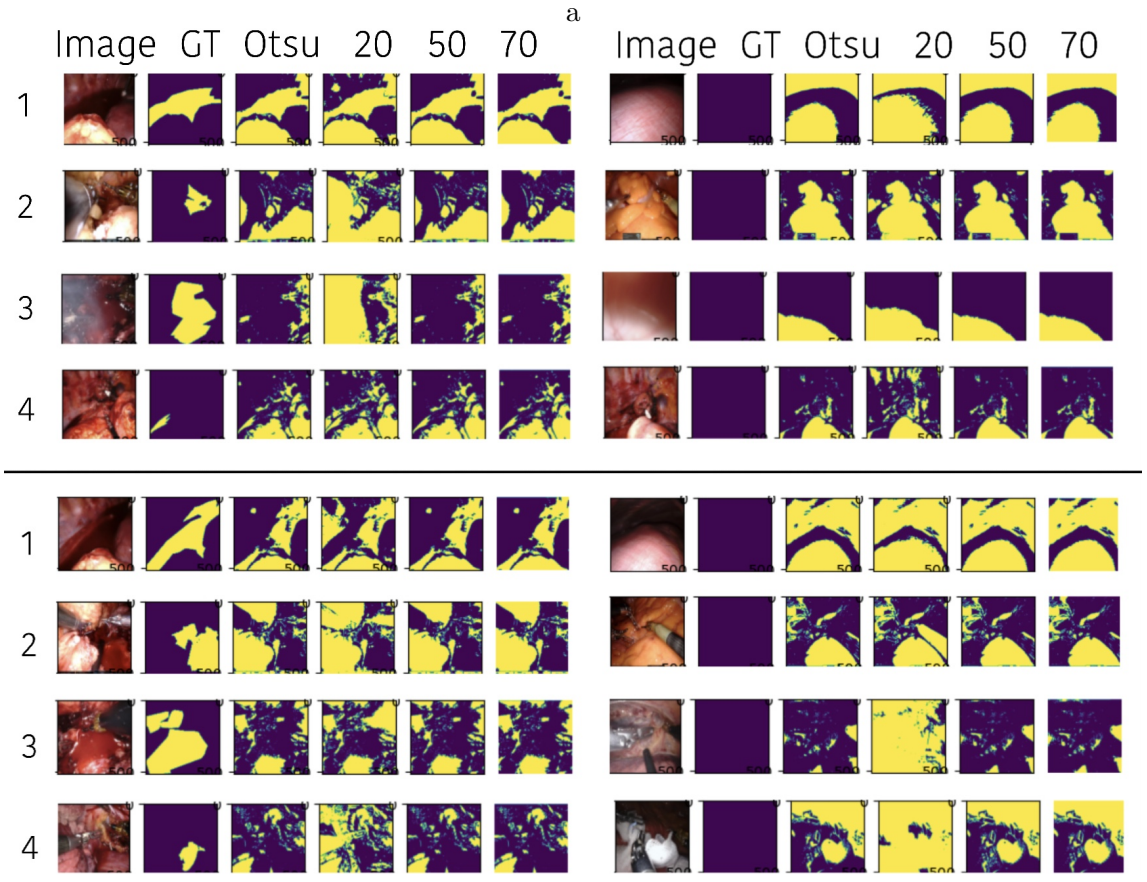

Figure 5.21: active blood detection results on all fold where **top** are for Simple negative **bottom** are for Hard negatives

5.2.6 Comparative Visualisation

5.2.6.1 fold-1

It's observed that all modules provide satisfactory/viable results but Puzzle-CAM and Grad-CAM presents a lot of overactivation. CV based approach fails to detect blood and instead focused on surgical instruments and background tissue. The results can be seen in Image 5.22.

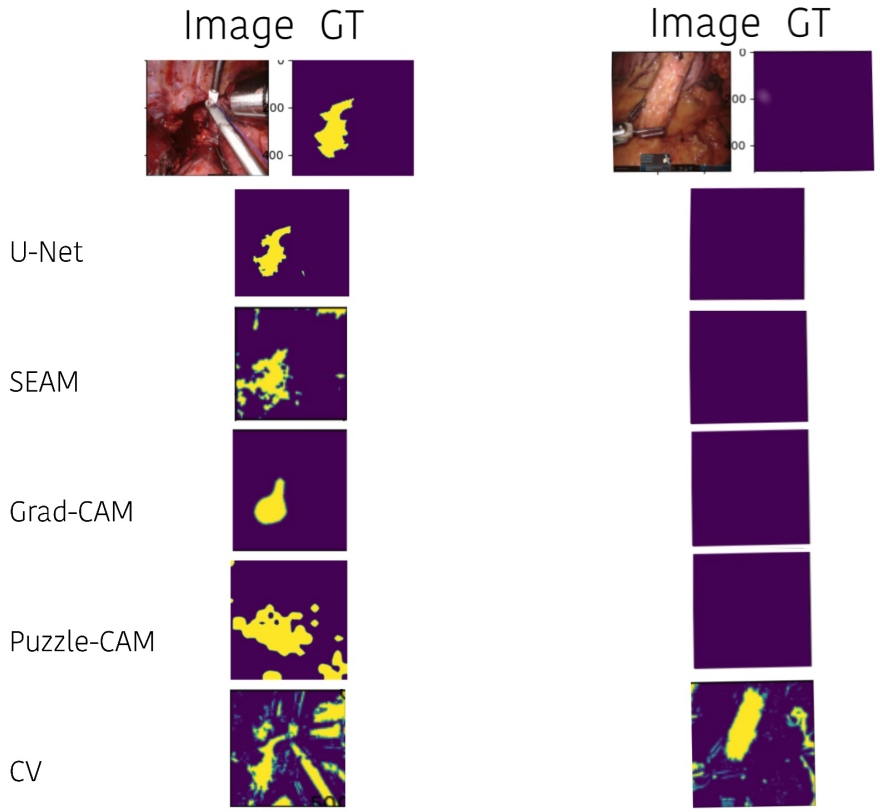

Figure 5.22: Condensed results on fold-1

5.2.6.2 Fold-2

It is observed that all the methods except Puzzle-CAM provide satisfactory/viable results. However, SEAM produces a mask in the negative sample and fails to classify the image correctly. The results can be seen in Image 5.23.

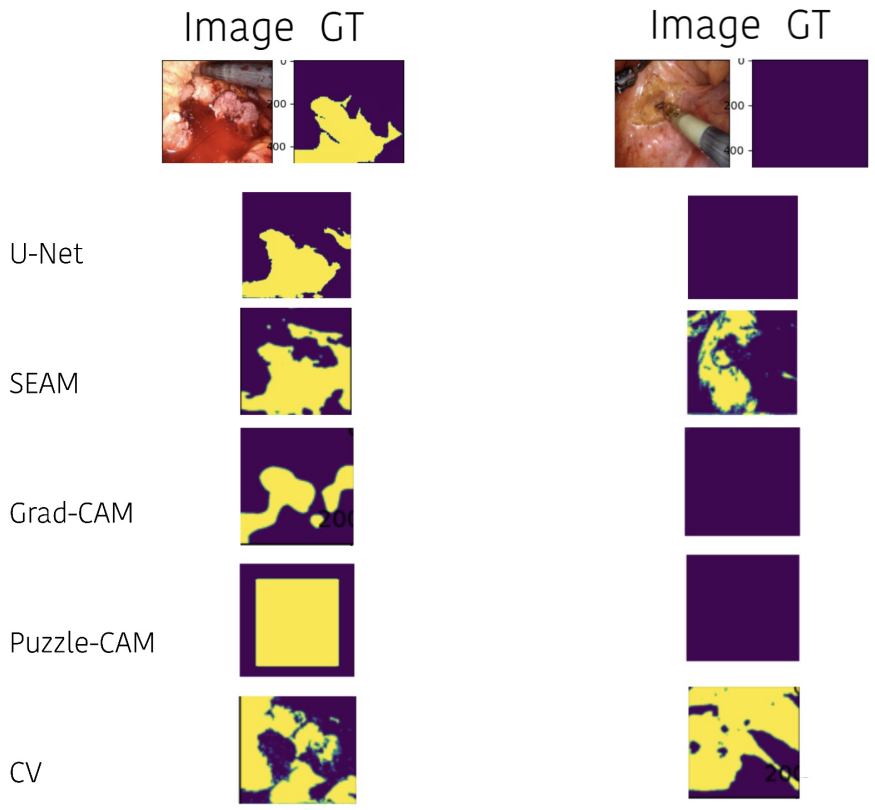

Figure 5.23: Condensed results on fold-2

5.2.6.3 Fold-3

Out of three weakly supervised approaches, Grad-CAM is the only model that produces usable results. Even though SEAM is predicting the blood accumulation correctly, it is also falsely predicting another region as well. The results can be seen in Image 5.24.

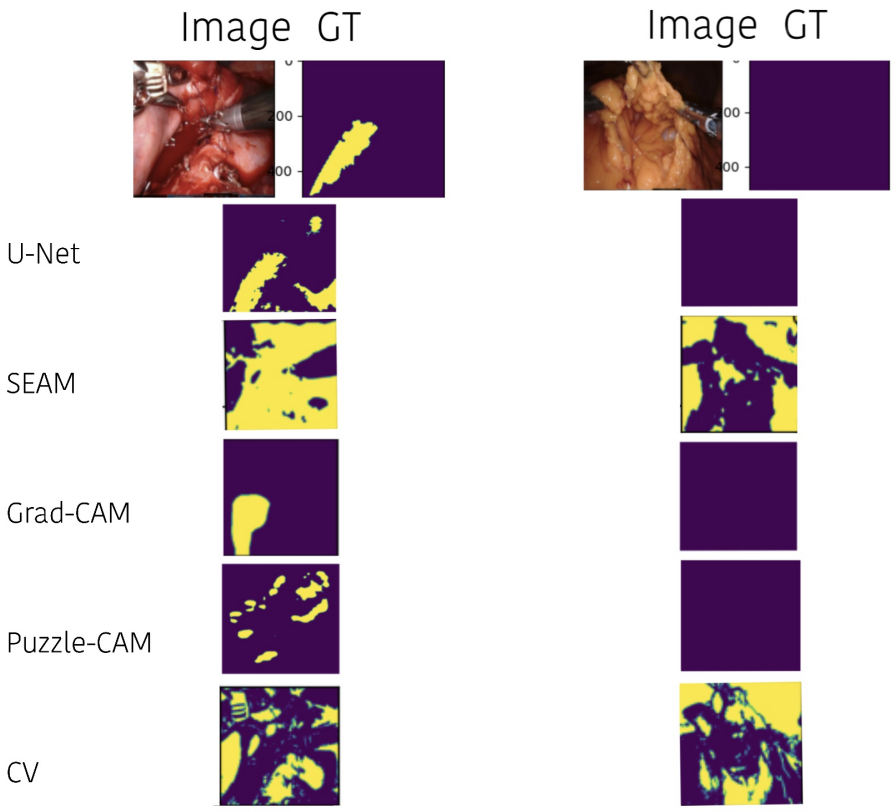

Figure 5.24: Condensed results on fold-3

5.2.6.4 Fold-4

Out of three weakly supervised approaches, Grad-CAM and SEAM are the only models that produce usable results. However, SEAM faults in negative sample. The results can be seen in Image 5.25.

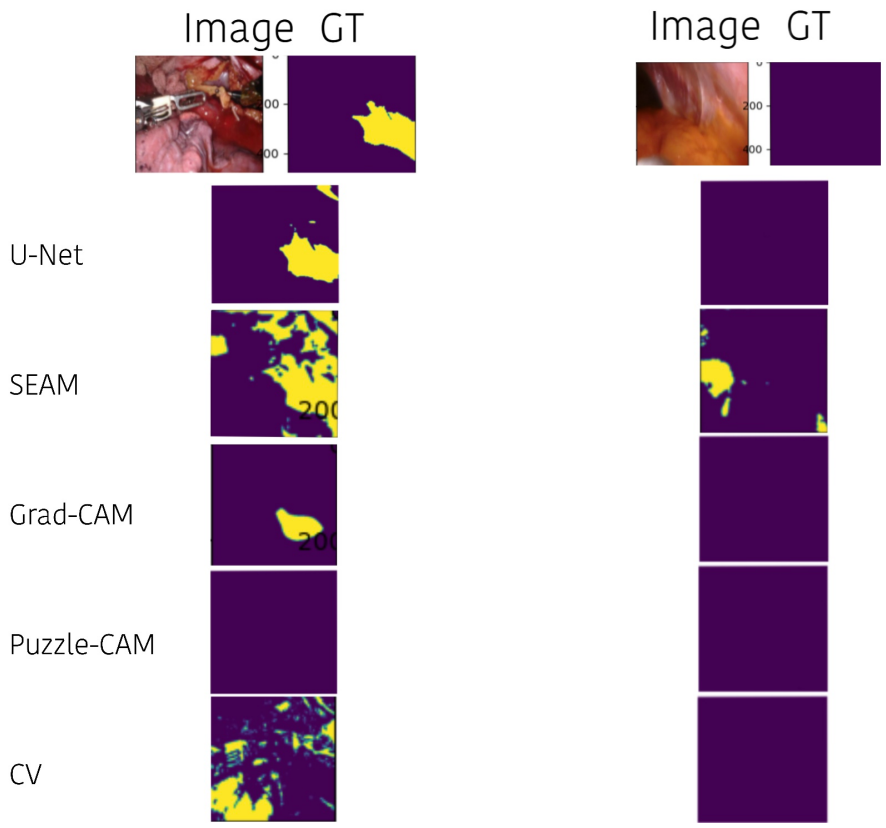

Figure 5.25: Condensed results on fold-4

5.3 Discussion

Except for U-Net and Non-training based approaches, all of the aforementioned methods were developed for multi-class Weakly-supervised semantic segmentation, and several changes were made to allow them to work for binary semantic segmentation. A thresholding function was introduced, and the last layer of the classifier was also included to make the task of mask generation more efficient. In this section, a detailed summary of the aforementioned results is presented.

The Fully supervised semantic segmentation approach utilises pixel-level labels and thus outperforms all the weakly supervised approaches in generating masks with nearly exact localisation and shape to the actual blood accumulation site. However, the application does not require exact boundaries but rough boundaries and precise accumulation location; we can use the Weakly-supervised approaches here since they can predict the location and generate a mask with viable borders.

To generate binary segmentation masks from the heatmaps generated by weakly-supervised methods, we require a thresholding function. In an extensive implementation and evaluation search, it appears that Otsu's thresholding does a satisfactory job of generating segmentation masks among all the thresholding functions that were evaluated [1]; this can be corroborated by looking at the observations of individual models in Chapter 5. To

evaluate the performance of the models properly, we use four-fold cross-validation. This number was empirically chosen. Fixed values for the threshold were also determined, but these values are often too restrictive and result in entirely blank masks (with no pixels labelled as foreground) in significantly many instances. Two of the source papers SEAM [19] and Puzzle-CAM [26] uses image-level labels in the evaluation phase to selectively evaluate on the classes belonging to the image. This kind of information is not available in a real-world scenario, and thus we use the approach mentioned above.

We observe that Grad-CAM outperforms the other two methods among the weakly-supervised approaches. This observation is substantiated with both qualitative and quantitative approaches. Grad-CAM is able to localise the site of the bleeding, but it does not produce a full resolution image; thus, the boundaries of the site are murky. On the other hand, SEAM can produce nice boundaries but often labels background regions that are not the actual site of accumulation as foreground, thus creating false regions that can cause distractions if utilised in a real-world scenario. This can be corroborated from the observation that the overall F1-Score of SEAM is 0.2113 vs that of Grad-CAM is 0.3548, and classifier accuracy of SEAM is 70.04 versus that of Grad-CAM is 83.21.

It is observed that the inclusion of Hard negatives helps in increasing the ability of a weakly supervised approach to localise the accumulation region. This can be corroborated by looking at the results from the Tables 5.14 and 5.15. However, the inclusion of Hard negatives visibly impact the performance of the supervised segmentation method in a negative way and also causes a significant drop in F1-Score.

Based on the quantitative evaluation, there are some anomalous observations to be made here as well that in fold 3, Grad-CAM performs worse when trained using Hard negatives and in fold 1 and 4 SEAM performs worse when trained on Hard negatives. This can probably be attributed to the fact that whenever Hard negatives are utilised in the set where the ground truth region's size is small, the model tends to misbehave and generate false regions in predicted masks.

Puzzle-CAM was the third model that was chosen to be evaluated. This model had a Mean IoU of 66.9 on the VOC2012 validation dataset. However, the original paper [26] utilises ResNet-101 as the backbone, and in this work, this was replaced with ResNet-50 for the sake of computation. It was observed that this model performs satisfactorily on folds 1 and 2 however fails to predict any viable masks on folds 3 and 4. This discrepancy can also probably be attributed to Ground Truth regions being too small and also because we use a small batch size of 6, resulting in the network performing worse.

The ResNet-50 backbone was further modified to use Group Normalisation as the max batch size that could be used due to hardware limitation was 6. This experiment did not produce any significant improvement, with an exception in the case of fold 1 trained on Hard negatives and evaluated on Simple negatives. This can probably be attributed to the model being dependent on Batch Normalisation. This can partially be corroborated by the observation that the model can still localise accumulation regions in folds 3 when we use Batch Normalisation. A further investigation is warranted to either change the network to adopt Group Normalisation or use different loss functions to improve the classification accuracy as the model performs poorly in this task. All the models use Dice loss as the Classification loss; however, this loss does not distinguish between Hard and Simple negatives, so a further investigation with other loss functions can probably improve the model's performance.

The non-learning based approach performs the worst among all the methods used in this work. The results often completely miss the blood accumulation regions or background is considered as blood accumulation region. In this method, Otsu's Thresholding does a satisfactory task.

In terms of time taken to generate the results, we observe that U-Net is the quickest, followed by Grad-CAM, SEAM, Puzzle-CAM and at last, we have the non-learning based approach. This slower time of approximately 7.62 seconds is a direct result of calculation delays caused by how NumPy handles the calculation in the colour spectrum transformation phase.

For future work, the Grad-CAM method can be further improved using Guided Backpropagation, as shown in Figure 4.5. This approach will allow Grad-CAM to produce a full resolution image and thus possibly bringing the performance close to a fully supervised approach. Puzzle-CAM's performance can be improved by performing a thorough Hyperparameter tuning and performing architecture search or weight standardisation.

Further studies on using a Graph-Convolutional Network can be done; however, this method requires significantly more resources than the methods used in this work and possibly require more Hyperparameter tuning. Visual analysis is performed on the results of all the methods on all four-folds to understand how well the localisation is done and intuitively understand the boundary discrepancies.

Finally we can hypothesise that all the training bases methods performs differently from the source papers because of the implicit class imbalance in the images. The region of blood accumulation is smaller in comparison to the background region and this creates a bias. A possible solution to this issue is to use a Weighted loss function [100] or Focal loss [101].

6. Conclusion

This work's two primary goals were to compare Weakly- supervised semantic segmentation approaches to a Fully-supervised semantic segmentation approach and a Non-learning segmentation approach and evaluate the impact of including Hard negatives during the training phase. These methods were motivated by the problems plaguing semantic segmentation for surgical image data. The data required for training often do not have proper pixel-level labels, and manual generation of these labels requires a lot of time and resources and often needs expert knowledge. Human biases can also influence labels.

Further, this work tries to find a method that can be used in a real-world scenario, for example, during laparoscopic surgery.

While doing laparoscopic surgery, surgeons might require assistance to monitor the site of active bleeding and achieve haemostasis. When a surgeon tries to achieve primary haemostasis, they might utilise a gauze or suction tool to remove blood, but this may partially affect the surgeon's vision and then it becomes harder to achieve secondary haemostasis. In this scenario, the results generated from the model can assist the surgeon in localising the site of bleeding and give vital time for surgeons to react. There are various ways in which weak supervision can be provided to the network, and due to its simplicity and low cost involved in the production, image-level labels have been used here.

Consequently, this work attempted to produce segmentation masks using three Weakly-supervised segmentation approaches and generate comparable results to a Fully-supervised segmentation approach. Furthermore, we compare them to a non-learning-based approach baseline that tells us if these Weakly-supervised approaches are effective compared to this Non-learning based approach.

The three Weakly-supervised semantic segmentation approaches use CAMs and perform further refinements to generate segmentation masks. They initially train a classifier and then remove the last layer to generate CAMs.

- Grad-CAM: information from the last layer before the pooling layer is extracted to generate segmentation masks.
- SEAM: uses a self-attention mechanism and introduces consistency regularisation to improve the CAMs.
- Puzzle-CAM: tries to improve the CAMs by shuffling the CAMs generated to recalibrate the attention of the network allowing more regions (less discriminative) of the object to show up in CAMs.

We evaluated and compared the methods on two different datasets, one containing Simple negatives and the other containing Hard negatives. Four-fold cross-validation was performed, and the impact of Hard negatives (i.e. images which contain blood but no accumulation) was investigated.

We observed that Weakly-supervised segmentation methods can produce viable results when compared to Fully-supervised approaches, and Grad-CAM performs the best among all the methods used here. The ad hoc nature, relatively less hardware requirement and shorter training time make this method promising. Complicated and more resource-hungry methods clearly perform worse and, in multiple instances, generate unusable results. The loss function (Dice loss) suffers from some issues and requires further investigation.

A visual evaluation indicates that SEAM also generates acceptable results. In some instances, it generates outstanding results; however, since this module also generates a lot of false positives, this can be considered worse than Grad-CAM.

On the other hand, Puzzle-CAM's performance probably can be improved, but there is a possibility that due to the inherent nature of the data used, the results may not improve. The inclusion of Hard negatives yielded positive results in weak supervision but negative results in full supervision. This is an interesting outcome and makes a good candidate for future work. There was a method WSGCN [16] mentioned in related work but was never used due to certain limitations. This method can provide a fresh approach to this task and would be interesting to investigate how we can extract and evaluate adjacency and saliency information. Furthermore, the limitations of Grad-CAM can be overcome by using Guided Backpropagation(see Figure 4.5) and this can be a future experiment.

List of Figures

List of Tables

Bibliography

[1] P. Guruprasad, "Overview of different thresholding methods in image processing," 06 2020.

[2] J. Schmidhuber, "Deep learning in neural networks: An overview," *Neural Networks*, vol. 61, p. 85–117, Jan 2015. [Online]. Available: http://dx.doi.org/10.1016/j.neunet.2014.09.003

[3] L. Chan, M. S. Hosseini, and K. N. Plataniotis, "A comprehensive analysis of weakly-supervised semantic segmentation in different image domains," *International Journal of Computer Vision*, vol. 129, no. 2, p. 361–384, Sep 2020. [Online]. Available: http://dx.doi.org/10.1007/s11263-020-01373-4

[4] J. Ahn and S. Kwak, "Learning pixel-level semantic affinity with image-level supervision for weakly supervised semantic segmentation," 2018.

[5] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, no. 2, p. 336–359, Oct 2019. [Online]. Available: http://dx.doi.org/10.1007/s11263-019-01228-7

[6] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, "Grad-cam: Why did you say that?" 2017.

[7] T. Durand, T. Mordan, N. Thome, and M. Cord, "WILDCAT: Weakly Supervised Learning of Deep ConvNets for Image Classification, Pointwise Localization and Segmentation," in *IEEE Conference on Computer Vision and Pattern Recognition (CVPR 2017)*. Honolulu, HI, United States: IEEE, Jul. 2017, pp. 5957–5966. [Online]. Available: https://hal.archives-ouvertes.fr/hal-01515640

[8] H. Kervadec, J. Dolz, M. Tang, E. Granger, Y. Boykov, and I. Ben Ayed, "Constrained-cnn losses for weakly supervised segmentation," *Medical Image Analysis*, vol. 54, p. 88–99, May 2019. [Online]. Available: http://dx.doi.org/10.1016/j.media.2019.02.009

[9] D. Knowles, "Lagrangian duality for dummies," 2010. [Online]. Available: https://www-cs.stanford.edu/people/davidknowles/lagrangian_duality.pdf

[10] L. László, "Random walks on graphs: A survey, combinatorics, paul erdos is eighty," *Bolyai Soc. Math. Stud.*, vol. 2, 01 1993.

[11] G. Papandreou, L.-C. Chen, K. Murphy, and A. L. Yuille, "Weakly- and semi-supervised learning of a dcnn for semantic image segmentation," 2015.

[12] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," 2017.

[13] Z. Huang, X. Wang, J. Wang, W. Liu, and J. Wang, "Weakly-supervised semantic segmentation network with deep seeded region growing," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2018, pp. 7014–7023.

[14] R. Adams and L. Bischof, "Seeded region growing," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 16, pp. 641–647, 1994.

[15] J. Fan, Z. Zhang, T. Tan, C. Song, and J. Xiao, "Cian: Cross-image affinity net for weakly supervised semantic segmentation," 2020.

[16] S.-Y. Pan, C.-Y. Lu, S.-P. Lee, and W.-H. Peng, "Weakly-supervised image semantic segmentation using graph convolutional networks," 2021.

[17] S. Zhang, H. Tong, J. Xu, and R. Maciejewski, "Graph convolutional networks: a comprehensive review," *Computational Social Networks*, vol. 6, no. 1, p. 11, Nov 2019. [Online]. Available: https://doi.org/10.1186/s40649-019-0069-y

[18] Y. Wang, J. Zhang, M. Kan, S. Shan, and X. Chen, "Self-supervised scale equivariant network for weakly supervised semantic segmentation," 2019.

[19] ——, "Self-supervised equivariant attention mechanism for weakly supervised semantic segmentation," 2020.

[20] B. Kim, S. Han, and J. Kim, "Discriminative region suppression for weakly-supervised semantic segmentation," 2021.

[21] Y. Li, Z. Kuang, L. Liu, Y. Chen, and W. Zhang, "Pseudo-mask matters in weakly-supervised semantic segmentation," 2021.

[22] J. Lee, E. Kim, and S. Yoon, "Anti-adversarially manipulated attributions for weakly and semi-supervised semantic segmentation," 2021.

[23] G. Sun, W. Wang, J. Dai, and L. V. Gool, "Mining cross-image semantics for weakly supervised semantic segmentation," 2020.

[24] D. Zhang, H. Zhang, J. Tang, X. Hua, and Q. Sun, "Causal intervention for weakly-supervised semantic segmentation," 2020.

[25] S. Lee, M. Lee, J. Lee, and H. Shim, "Railroad is not a train: Saliency as pseudo-pixel supervision for weakly supervised semantic segmentation," 2021.

[26] S. Jo and I.-J. Yu, "Puzzle-cam: Improved localization via matching partial and full features," *2021 IEEE International Conference on Image Processing (ICIP)*, Sep 2021. [Online]. Available: http://dx.doi.org/10.1109/ICIP42928.2021.9506058

[27] Q. Yao and X. Gong, "Saliency guided self-attention network for weakly and semi-supervised semantic segmentation," 2020.

[28] T. Okamoto, T. Ohnishi, H. Kawahira, O. Dergachyava, P. Jannin, and H. Haneishi, "Real-time identification of blood regions for hemostasis support in laparoscopic surgery," *Signal, Image and Video Processing*, vol. 13, pp. 405–412, 2019.

[29] M. Hajabdollahi, R. Esfandiarpoor, S. M. R. Soroushmehr, N. Karimi, S. Samavi, and K. Najarian, "Segmentation of bleeding regions in wireless capsule endoscopy images an approach for inside capsule video summarization," 2018.

[30] Y. S. Jung, Y. H. Kim, D. H. Lee, and J. H. Kim, "Active blood detection in a high resolution capsule endoscopy using color spectrum transformation," in *2008 International Conference on BioMedical Engineering and Informatics*, vol. 1, 2008, pp. 859–862.

[31] R. Zhao, B. Qian, X. Zhang, Y. Li, R. Wei, Y. Liu, and Y. Pan, "Rethinking dice loss for medical image segmentation," in *2020 IEEE International Conference on Data Mining (ICDM)*, 2020, pp. 851–860.

[32] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015.

[33] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, May 2015. [Online]. Available: https://doi.org/10.1038/nature14539

[34] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems*, F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper/2012/file/c399862d3b9d6b76c8436e924a68c45b-Paper.pdf

[35] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016, http://www.deeplearningbook.org.

[36] "A comprehensive guide to convolutional neural networks — the eli5 way." [Online]. Available: https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[37] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," 2015.

[38] "Types of convolution kernels : Simplified." [Online]. Available: https://towardsdatascience.com/types-of-convolution-kernels-simplified-f040cb307c37

[39] "cs231n stanford." [Online]. Available: https://cs231n.github.io/

[40] J. Singh and R. Banerjee, "A study on single and multi-layer perceptron neural network," in *2019 3rd International Conference on Computing Methodologies and Communication (ICCMC)*, 2019, pp. 35–40.

[41] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische mathematik*, vol. 1, no. 1, pp. 269–271, 1959.

[42] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2017.

[43] H. Zhang, I. Goodfellow, D. Metaxas, and A. Odena, "Self-attention generative adversarial networks," 2019.

[44] B. Holländer, "Self-attention in computer vision," 2019. [Online]. Available: https://towardsdatascience.com/self-attention-in-computer-vision-2782727021f6

[45] J. Fernandez, "Attention in computer vision implementing multihead and cbam attention modules in pytorch," 2021. [Online]. Available: https://towardsdatascience.com/attention-in-computer-vision-fd289a5bd7ad

[46] "What are saliency maps in deep learning?" [Online]. Available: https://analyticsindiamag.com/what-are-saliency-maps-in-deep-learning/

[47] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 20, pp. 1254 – 1259, 12 1998.

[48] K. Simonyan, A. Vedaldi, and A. Zisserman, "Deep inside convolutional networks: Visualising image classification models and saliency maps," 2014.

[49] "Activation functions." [Online]. Available: https://ml-cheatsheet.readthedocs.io/en/latest/activation_functions.html#relu

[50] "intro to relu." [Online]. Available: https://machinelearningmastery.com/rectified-linear-activation-function-for-deep-learning-neural-networks/

[51] S. Jadon, "A survey of loss functions for semantic segmentation," *2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB)*, Oct 2020. [Online]. Available: http://dx.doi.org/10.1109/CIBCB48159.2020.9277638

[52] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006, rOC Analysis in Pattern Recognition. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S016786550500303X

[53] D. Powers, "Evaluation: From precision, recall and f-factor to roc, informedness, markedness correlation," *Mach. Learn. Technol.*, vol. 2, 01 2008.

[54] C. Sammut and G. I. Webb, *Encyclopedia of Machine Learning*, 1st ed. Springer Publishing Company, Incorporated, 2011.

[55] S. Du, "Understanding dice loss for crisp boundary detection," 2020. [Online]. Available: https://medium.com/ai-salon/understanding-dice-loss-for-crisp-boundary-detection-bb30c2e5f62b

[56] "Pytorch multi label soft margin loss." [Online]. Available: https://pytorch.org/docs/stable/generated/torch.nn.MultiLabelSoftMarginLoss.html#multilabelsoftmarginloss

[57] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[58] "What is the difference between test set and validation set?" [Online]. Available: https://stats.stackexchange.com/questions/19048/what-is-the-difference-between-test-set-and-validation-set

[59] M. Stone, "Cross-validatory choice and assessment of statistical predictions," *Journal of the Royal Statistical Society: Series B (Methodological)*, vol. 36, no. 2, pp. 111–133, 1974. [Online]. Available: https://rss.onlinelibrary.wiley.com/doi/abs/10.1111/j.2517-6161.1974.tb00994.x

[60] "Cross-validation (statistics) wikipedia." [Online]. Available: https://en.wikipedia.org/wiki/Cross-validation_(statistics)

[61] N. Otsu, "A threshold selection method from gray level histograms," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 9, pp. 62–66, 1979.

[62] "Otsu's method for image thresholding explained and implemented." [Online]. Available: https://muthu.co/otsus-method-for-image-thresholding-explained-and-implemented/

[63] M. Lin, Q. Chen, and S. Yan, "Network in network," 2014.

[64] B. Zhou, A. Khosla, L. A., A. Oliva, and A. Torralba, "Learning Deep Features for Discriminative Localization." *CVPR*, 2016.

[65] "Transposed convolution demystified," 2020. [Online]. Available: https://towardsdatascience.com/transposed-convolution-demystified-84ca81b4baba

[66] "Transposed convolution." [Online]. Available: https://d2l.ai/chapter_computer-vision/transposed-conv.html

[67] "Autoencoder: Downsampling and upsampling." [Online]. Available: https://kharshit.github.io/blog/2019/02/15/autoencoder-downsampling-and-upsampling

[68] "transposed convolution demystified." [Online]. Available: https://towardsdatascience.com/transposed-convolution-demystified-84ca81b4baba

[69] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[70] A. Kolesnikov and C. H. Lampert, "Seed, expand and constrain: Three principles for weakly-supervised image segmentation," 2016.

[71] "Normalization techniques in deep neural networks." [Online]. Available: https://medium.com/techspace-usict/normalization-techniques-in-deep-neural-networks-9121bf100d8

[72] "Batch normalization in convolutional neural networks." [Online]. Available: https://www.baeldung.com/cs/batch-normalization-cnn

[73] E. Tiu, "Metrics to evaluate your semantic segmentation mode," 2019. [Online]. Available: https://towardsdatascience.com/metrics-to-evaluate-your-semantic-segmentation-model-6bcb99639aa2

[74] "Nesterov accelerated gradient." [Online]. Available: https://compsci682-fa21.github.io/notes/neural-networks-3/

[75] "Nesterov accelerated gradient and momentum batch normalization in convolutional neural networks." [Online]. Available: https://jlmelville.github.io/mize/nesterov.html

[76] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2017.

[77] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of Big Data*, vol. 6, no. 1, p. 60, Jul 2019. [Online]. Available: https://doi.org/10.1186/s40537-019-0197-0

[78] F. Dubost, H. Adams, P. Yilmaz, G. Bortsova, G. van Tulder, M. A. Ikram, W. Niessen, M. W. Vernooij, and M. de Bruijne, "Weakly supervised object detection with 2d and 3d regression neural networks," *Medical Image Analysis*, vol. 65, p. 101767, 2020. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1361841520301316

[79] S. MUELLER, "Car model classification iii: Explainability of deep learning models with grad-cam," 2021. [Online]. Available: https://www.statworx.com/at/blog/erklaerbbarkeit-von-deep-learning-modellen-mit-grad-cam/

[80] R. R. Selvaraju, M. Cogswell, A. Das, R. Vedantam, D. Parikh, and D. Batra, "Grad-cam: Visual explanations from deep networks via gradient-based localization," *International Journal of Computer Vision*, vol. 128, no. 2, p. 336–359, Oct 2019. [Online]. Available: http://dx.doi.org/10.1007/s11263-019-01228-7

[81] K. Vinogradova, A. Dibrov, and G. Myers, "Towards interpretable semantic segmentation via gradient-weighted class activation mapping," 2020.

[82] S. Ulyanin, "Implementing grad-cam in pytorch," 2019. [Online]. Available: https://medium.com/@stepanulyanin/implementing-grad-cam-in-pytorch-ea0937c31e82

[83] ——, "Implementing grad-cam in pytorch," 2019. [Online]. Available: https://medium.com/@stepanulyanin/implementing-grad-cam-in-pytorch-ea0937c31e82

[84] M. Chetoui, "Gradient-weighted class activation mapping - grad-cam," 2019. [Online]. Available: https://medium.com/@mohamedchetoui/grad-cam-gradient-weighted-class-activation-mapping-ffd72742243a

[85] D. Reiff, "Understand your algorithm with grad-cam," 2021. [Online]. Available: https://towardsdatascience.com/understand-your-algorithm-with-grad-cam-d3b62fce353

[86] S. Ulyanin, "Implementing grad-cam in pytorch," 2019. [Online]. Available: https://medium.com/@stepanulyanin/implementing-grad-cam-in-pytorch-ea0937c31e82

[87] R. Draelos, "Grad-cam: Visual explanations from deep networks," 2020. [Online]. Available: https://glassboxmedicine.com/2020/05/29/grad-cam-visual-explanations-from-deep-networks/

[88] "Explore semantic segmentation network using grad-cam." [Online]. Available: https://in.mathworks.com/help/deeplearning/ug/explore-semantic-segmentation-network-using-gradcam.html

[89] R. Khandelwal, "Evaluating performance of an object detection model," 2020. [Online]. Available: https://towardsdatascience.com/evaluating-performance-of-an-object-detection-model-137a349c517b

[90] "Curse of batch normalization," 2020. [Online]. Available: https://towardsdatascience.com/curse-of-batch-normalization-8e6dd20bc304

[91] Y. Wu and K. He, "Group normalization," 2018.

[92] "U-net source." [Online]. Available: https://github.com/mateuszbuda/brain-segmentation-pytorch/blob/master/unet.py

[93] "General data protection regulation." [Online]. Available: https://gdpr-info.eu/

[94] "F1/dice-score vs iou / stackexchange." [Online]. Available: https://stats.stackexchange.com/questions/273537/f1-dice-score-vs-iou

[95] Z. Liu, H. Mao, C.-Y. Wu, C. Feichtenhofer, T. Darrell, and S. Xie, "A convnet for the 2020s," 2022.

[96] A. Brock, S. De, S. L. Smith, and K. Simonyan, "High-performance large-scale image recognition without normalization," 2021.

[97] F. Faghri, D. J. Fleet, J. R. Kiros, and S. Fidler, "Vse++: Improving visual-semantic embeddings with hard negatives," 2018.

[98] X. Xia, Q. Lu, and X. Gu, "Exploring an easy way for imbalanced data sets in semantic image segmentation," *Journal of Physics: Conference Series*, vol. 1213, no. 2, p. 022003, jun 2019. [Online]. Available: https://doi.org/10.1088/1742-6596/1213/2/022003

[99] J. Lee, J. Choi, J. Mok, and S. Yoon, "Reducing information bottleneck for weakly supervised semantic segmentation," 2021.

[100] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," 2016.

[101] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," 2018.