UKRAINIAN CATHOLIC UNIVERSITY

BACHELOR THESIS

On Salient Object Detection

Author:
Olha PROTS

Supervisor:
Mr. Orest KUPYN

A thesis submitted in fulfillment of the requirements for the degree of Bachelor of Science

in the

Department of Computer Sciences
Faculty of Applied Sciences

Lviv 2022

Declaration of Authorship

I, Olha PROTS, declare that this thesis titled, "On Salient Object Detection" and the work presented in it is my own. I confirm that:

- This work was done wholly or mainly while in candidature for a research degree at this University.
- Where any part of this thesis has previously been submitted for a degree or any other qualification at this University or any other institution, this has been clearly stated.
- Where I have consulted the published work of others, this is always clearly attributed.
- Where I have quoted from the work of others, the source is always given. With the exception of such quotations, this thesis is entirely my own work.
- I have acknowledged all main sources of help.
- Where the thesis is based on work done by myself jointly with others, I have made clear exactly what was done by others and what I have contributed myself.

Signed:

Date:

"Don't cry because it's over. Smile because it happened."

Dr Seuss

UKRAINIAN CATHOLIC UNIVERSITY

Faculty of Applied Sciences

Bachelor of Science

On Salient Object Detection

by Olha PROTS

Abstract

Salient object detection is the process of finding the most visually catching objects in the image. This can be very beneficial to outline the region of interest and use that information for further image processing. In this paper, we review the most common approaches to this problem and propose a simple approach that strives to be compact and efficient. Since most SOTA solutions achieve their accuracy by sacrificing computational efficiency, they are not suitable for limited resources. We test an approach that achieves comparative results on much smaller UNet and Unet++ models.

Acknowledgements

First of all, I want to express my gratitude towards my supervisor Mr. Orest KUPYN for helping me with this topic, its research and ideas for experiments. Thank you to Oles Dobosevych for assisting my research as well. Moreover, I want to say thank you to all the teachers, professors and mentors I was lucky to have during my time at UCU and a big thank you to the Faculty of Applied Sciences.
Finally, great thanks to my dear beloved friends and family for pushing me to finish this paper.

Contents

List of Figures

Dedicated to my dear teachers

Chapter 1

Introduction

1.1 Context

In recent years, salient object detection has seen great improvement in research. Given the fact that we produce images and videos every day, it can often be beneficial to know the parts of them the viewer finds most entertaining or important. This task is not new, but thanks to the rise in deep learning networks creation and use, the proposed ways to solve it have drastically changed throughout the years. Earlier methods tried their best to utilize window feature extraction [39, 40, 8, 34, 24], be it through pixel contrast, key-points or edge detection, or relative pixel intensity. Today, convolutional neural networks [31, 27, 21] manage to extract much deeper features and find multiple salient objects even in extremely visually crowded environments.

Nowadays, finding visually attentive regions is a pre-processing step for many other computer vision tasks. Saliency segmentation is used in automated image cropping [4], object re-identification [32], content-aware image compression [56, 3], robot navigation [9] and utilized in such tasks as image classification, object detection and semantic segmentation. The knowledge about a person or an object that is captured on camera can help save resources in many possible ways. Compressing the less salient regions of images to save space, detecting a potential distraction object in view while driving a car, changing the direction of the camera by tracking the most interesting objects in the frame can all benefit from computers learning what catches our eye. Thanks to the many applications saliency segmentation has, the research continues.

1.2 Motivation

One of the aspects that still need improvement is the ability of a model to be precise even with little computing, memory and time resources. Embedded devices need to have a solution that does not lack in efficiency but requires little memory, especially considering that saliency detection is often only a pre-processing step before another computation happens. Most SOTA approaches result in a model that is 100 mb or bigger, and achieve great accuracy by sacrificing speed and utilizing high resource machines.

That is why we wanted to experiment with utilising known backbones such as MobileNet [36] and EfficientNet[38] and proven architectures such as U-Net [33] and U-Net++ [55] to create a solution that can compete with SOTA models while also being compact. U-Net architectures have shown amazing results in medical image segmentation by capturing both global context and local details through the use of an encoder-decoder architecture. For the encoder, we chose EffiicentNet, which is

a compact convolutional network that achieves great results on transfer learning datasets, and MobileNet, which is a mobile architecture with a custom module that brings SOTA performance to models for limited resource devices.

1.3 Goal

1. Provide an overview of methods used for salient object detection.
2. Find a compact and accurate solution for low-memory devices that need to leverage the results of saliency segmentation models. The model has to be able to extract both high-level features and pixel-wise information.
3. For this, we will experiment on a convolutional neural network with an encoder-decoder architecture. The encoder will be a compact backbone network, pre-trained on [10] for image classification.

1.4 Approach

In this experiment, we will use an encoder-decoder network architecture with skip connections, created for semantic segmentation. Such a network manages to capture both local and global features of the image and has proven to be successful in medical image segmentation [33] as well as general semantic segmentation tasks. The networks we will use are U-Net [33] and U-Net++ [55], an improvement based on U-Net, with redesigned dense skip connections.

Two small backbones, MobileNetV2 [36] and EfficientNet-lite [38], pre-trained on ImageNet, will be used as encoders. Both backbones were created with mobile/IoT devices in mind, EfficientNet-lite being an optimization of the original EfficientNet architecture.

The dataset we will use for training is a 10,553 images DUTS-TR [41] dataset with simple augmentations, and for testing, we will use its 5,019 image companion, DUTS-TE, as well as HKU-IS, DUT-OMRON, PASCAL-S and ECSSD. We will compare the model with SOTA approaches using three most common evaluation metrics - max F-measure, MAE and S-measure.

1.5 Thesis Structure

Chapter 2: Background

Here we give a the general idea of neural networks and how they work. We also give a description of convolutional neural networks and an encoder-decoder architecture that we use for salient object segmentation.

Chapter 3: Related works

Here we give a brief summary of methods used to detect saliency in the past and SOTA approaches that aim to detect salient objects today.

Chapter 4: Methodologies

Here we outline the architectures we use to solve the task of fast and compact salient object segmentation. Also, we describe the losses used to train the network.

Chapter 5: Experiments

Here we describe the evaluation metrics, implementation details of the solution and our results compared to SOTA approaches.

Chapter 6: Conclusions

Here we make conclusions based on the results and talk about future work.

Chapter 2

Background

In order to turn an RGB image into a binary representation of salient and non-salient regions or objects in the image and in order to do it fast, we can use a CNN Encoder-Decoder network. Let's discuss the main concepts that go into it.

2.1 Neural Networks

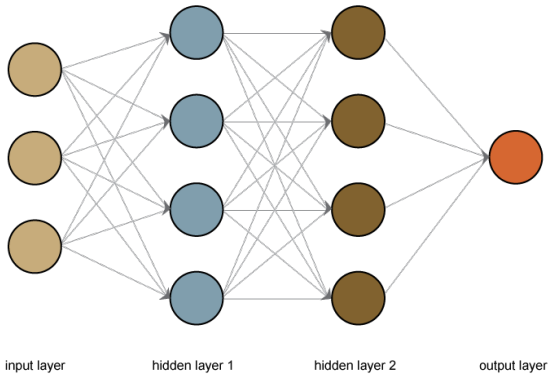

FIGURE 2.1: An example of an artificial neural network

A neural network is a layered system of computations performed on some input. It outputs data that represents an answer to the question the system is trained to solve. Originally, it was inspired by the biological neural networks present in the brain, hence the name. The first layer is the input layer, then come one or more hidden layers that consecutively perform operations on said input, transforming it along the way, and lastly goes output layer that combines information from the last hidden layer into a solution.

Hidden layers consist of nodes, or neurons; each node collects weighted inputs from the nodes of the previous layer, sums them up, adds a certain bias and uses an activation function to add non-linearity to the system. The activation function and the weights that are assigned to the inputs influence the strength of the signal that the neuron sends further along the network. Thus, the weights represent the importance of the inputs and the neuron connection they travel along. By finding the most optimal weights, the system can leverage complex patterns or parts of the input to make correct predictions from the given data.

The process of doing so is what constitutes training the network, and it is done by minimizing the difference between the prediction and the ground truth - the error.

The error, calculated using a so-called loss function, is then back propagated through the network layers to update the weights in an attempt to get a smaller error on the next try.

Figure 2.1 shows a four-layer neural network. The layers are fully-connected, meaning each node from the previous layer sends its output to each node of the next. Other types of layers that are present in more complex networks include convolution, de-convolution, recurrent, pooling and batch normalization. Convolution layers are commonly used in networks designed for computer vision tasks.

2.2 Convolutional Neural Networks

CNNs are a type of networks that work efficiently with structured data such as images. Their main feature is a convolution layer, which employs a filter, a matrix of weights that is applied to each two-dimensional image channel (or any other grid-like input structure) using a sliding window approach. The dot product between the weight matrix and a local part of the input is calculated for each possible square of input features of the same size as the filter. The size of the square, the stride with which the matrix moves across the input grid, the padding around the image border and the amount of filters used are set up manually, while the entries of the filter matrices are optimized during the training. After a filter is run across the image, an activation map (or feature map) is created as a result, and an activation function is applied to the map to introduce non-linearity.

Non-linearity is needed in all neural networks since linear functions are limited in the range of input data they can find a distinct output for. Without activation functions, a neural network could potentially be reduced to a simple vector product with an added bias, which is not complex enough of a function for the problems neural networks solve today.

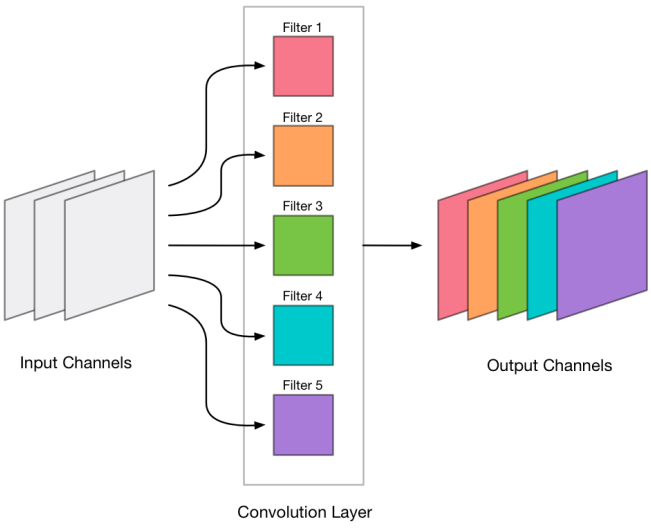

FIGURE 2.2: An example of convolution layer. Input gets processed by 5 separate filters, which allows to extract 5 different features of the same image

As shown in figure 2.2, more filters create more dimensions of data, which can heavily increase the amount of data the network has to process. Thankfully, the process of convolution down-samples the image, due to the nature of dot product operation, and making the stride bigger can also help condense the image data into a

much smaller representation. As seen in figure 2.3, nine entries from previous layer are combined into one in the next, but the amount of dimensions was multiplied by four.

The main idea behind this process is that the filters of the first layer can learn to look for small local features like edges, colors or lines, and each new convolution layer will receive more condensed information about the image parts due to down-sampling, so the last layers will be able to extract much more complex patterns, since they will receive more high-level information.

One major benefit of this approach on images is that for each feature extracting filter, one weight matrix is used across all local image regions, so the amount of parameters to be learned does not depend on the size of the image, which is not the case with previously mentioned fully-connected layers. This is called partial connectivity, or local connectivity, and usually in a convolutional network, a fully connected layer comes after one or more convolutions to bring the local information together and form the output.

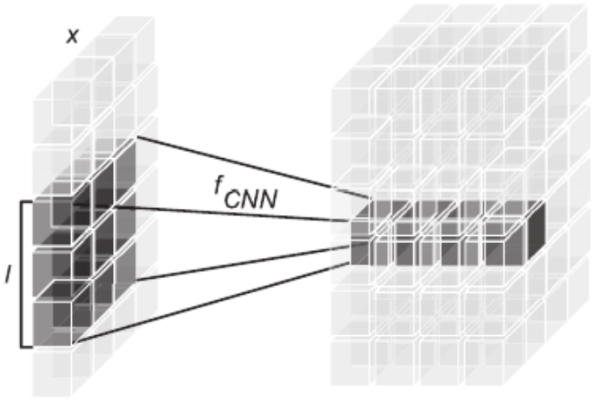

FIGURE 2.3: An example of a convolution layer. A feature window of fixed size from one dimension is transformed using four kernel functions into four dimensions of down-sampled features, image from [16]

2.3 Encoder-Decoder Network for Semantic Segmentation

Apart from convolution layers, CNNs and other types of networks use a pooling layer, which is used to down-sample the image. A common pooling technique is max pooling, where a max value is taken from each 2x2 or 3x3 square of the input layer to summarize the presence of features. This layer, along with a down-sampling convolution, allow CNNs to achieve deep learning, as they allow for creation of many dimensions, each of which brings out a new feature from the image. Convolutional neural networks perform great on many computer vision tasks, including image classification, image segmentation, object detection and recognition.

However, in case of image segmentation, the output has to answer two questions at once: what is present in the image and which pixels contain said object or objects. For this to be possible, we need to up-sample the output of convolution layers back to a feature map that can represent the image and show us where the objects are. This is where an Encoder-Decoder CNN proves to be efficient.

Figure 2.4 shows an example of an encoder-decoder network - U-Net, which we use in our experiments. U-Net can also be considered a so-called fully convolutional

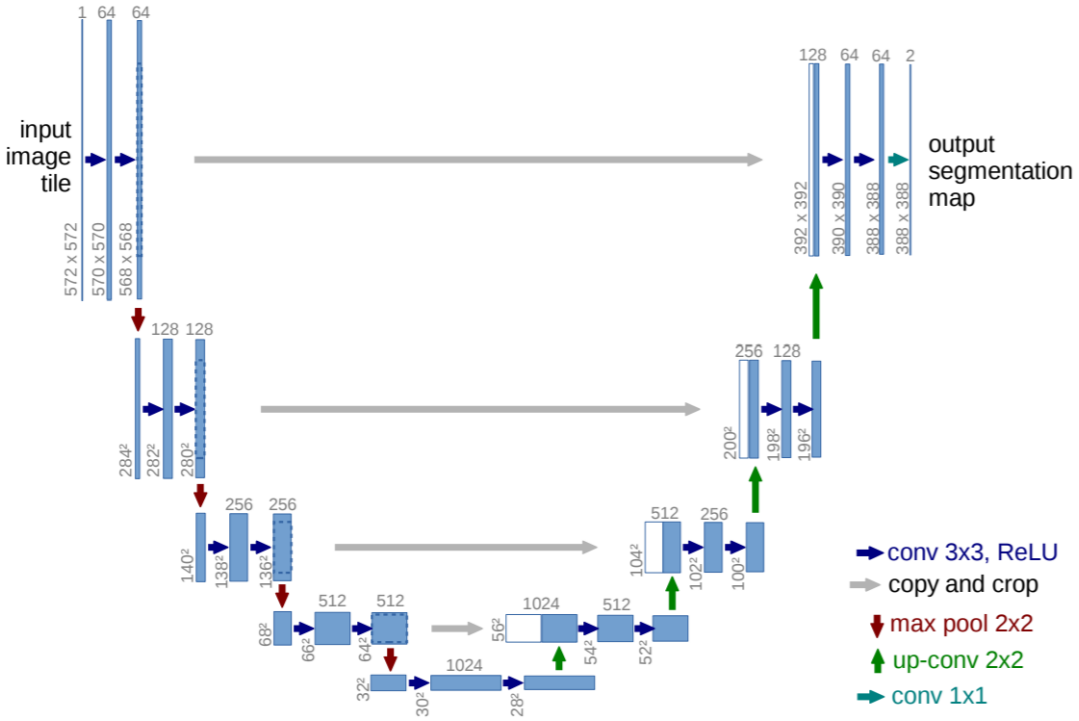

FIGURE 2.4: Architecture of U-Net

network, since it is made up of convolution layers, but does not have a fully connected layer unlike other CNNs. The lack of this layer also allows the network to work with any size of input images. U-Net architecture is revisited in Chapter 4: Methodologies.

An encoder-decoder network can be divided into two virtually symmetrical parts: a contracting side, that consists of convolution and down-sampling layers that extract high level features of the image, and an expansion side where the output of the contracting side is up-sampled back to a segmentation binary mask of the input image. A highly dimensional vector that the encoder outputs is shrunk in dimensions, but given back the grid-like structure of local features. Thus, the training process now consists of optimizing matrix weights in filters that extract features in the encoder as well as those that try to up-sample those features in the decoder.

Chapter 3

Related works

There are numerous works on saliency-based image segmentation and object detection. Before the deep neural networks gained their popularity, researchers mostly used hand-crafted features to detect salient regions of the image. Today, on the other hand, quite a few deep learning models have been created for this task, and many more have been utilized for it as well.

Two most used types of data are collected eye-fixations that create blob-like salient regions on the image, and binary segmentation masks, where the subject decides what is the salient object in the image before him. Both of these can give us valuable information about the human visual attention, although it will not be completely similar. The approaches that use these two distinct data types have usually little overlap, with segmentation masks' based ones gaining popularity because of their ability to be helpful in many further important applications, such as object recognition, object detection, image segmentation. Overall, one can distinguish between two approaches to detecting saliency: bottom-up and top-down.

3.1 Bottom-up techniques

Bottom-up techniques get their motivation from the idea that human visual attention has a bottom-up stage: we first notice rare, surprising, striking parts of the image at the very detailed level. Using this idea, they try to start from the low-level image qualities, get some saliency measurement of separate pixels or small regions based on those qualities, and later use normalized pixel-wise saliency levels to determine the highest ones by some non-linear operations. Qualities that are used in these techniques include contrast (local or global), intensity, motion, edges and others. Also, center bias or image boundary prior [50] were used: center is often considered more likely to have an object, and the boundary very unlikely. This is very clearly not the case for many images and thus is not efficient to use today. Similarly, a background prior used in [46] utilizes a rather limited assumption that regions from background, or non-salient area, are closer in distance to each other than to a region in the salient object area.

Some examples of past approaches include one with a histogram-based contrast and a center bias [8], one that computes local saliency levels on random image patches and utilizes color space similar to human psycho-visual space and pixel intensity values [39], and a simple edge density method [34]. These and others are all generally fast, often due to them making shortcuts, and do not require any trained base. Their results usually are more resembling of eye-fixation data blobs than full segmented objects.

3.2 Top-down techniques

Apart from data-driven methods that relied on bottom-up attention, or attention based on visual stimuli, some also tried to predict attention with prior knowledge [28], which can be classified as top-down. Bottom-up methods can easily have false predictions due to the fact that they do not know what they are looking for, rather expect to find what the person is subconsciously aiming to look at. A more targeted approach is to make the model look for some object of interest. Here, [52] a conditional random field together with a learning approach is used to detect regions containing a target object, taken from a dictionary of visual words.

3.3 CNN architectures

With the introduction of convolutional neural networks, saliency segmentation became much more efficient, since they allowed to fuse bottom-up features and high-level information about the object that is in the image, thus both getting rid of many false predictions caused by cluttered backgrounds or low contrast and surpassing a simpler top-down approach.

First, fully convolutional network (FCN) was largely utilized for SOD, but since its output is a small high-level feature map, because it is achieved by multiple convolutions and pooling layers, it is hard to generate a prediction with fine boundaries out of it. Trying to fuse this output with hand-crafted low-level features, computed separately, was far from efficient. Thus, multi-scale feature extraction was introduced. The next step was to correctly communicate the features between the levels, since they are complementary to each other. Deep high-level features can tell us about the location of the most important abstract areas (giving context), and low-level features preserve edge and structural information, giving detailed clues that are present in both the salient objects and the cluttered backgrounds, so the high-level feature helps to choose where to look.

In [13], they create short connections that pass information from deep feature maps to the shallow ones, providing them with context. However, the context is also present on multiple scales, and the low-level details should also reach the deep layers. To transmit the two-side information between the high-level and the low-level layers and make sure that the information transmitted is useful, A bi-directional message passing model was created by [53].

Recurrent networks have also been utilized, either to perform refinement on selected image patches [17] using spatial transformer and recurrent network units, to transfer global semantic information from the deep to shallow levels by multi-path recurrent connections [54], or to utilize a saliency prior map in learning and iteratively correct false predictions[42].

Refinement modules or strategies which aim to create more accurate boundaries of the object were also highly researched [14, 22, 43]. A recent example, BASNet [30], adds a residual refinement module to a deeply supervised encoder-decoder architecture to go from a coarse to a fine overall prediction, and proposes a hybrid loss consisting of IOU, Binary Cross Entropy and SSIM [44] losses.

After the success of U-Net, encoder-decoder model used to segment medical images, encoder-decoder frameworks were used in salient object detection as a way to efficiently extract multi-level features, with skip connections that add low-level features to the high-level outputs. A cascaded partial decoder was proposed [47] with an assumption that the most low-level features are not important enough to sacrifice

computational resources on, since the important info such as edges still remain in their higher-level representations. Thus, the decoder only takes those higher-level features into account, still remaining accuracy, but decreasing the resolution of the inputs of decoder layers.

Another development over U-Net is U^2Net [31], a model that uses a nested U-Net like architecture composed of custom RSU blocks in order to remove the need for pre-trained backbones. They propose a Residual U-block (RSU) that contains in itself an encoder-decoder module that extracts multi-scale features and is fused with a local feature, compared to convolution layers that constitute original U-Net blocks.

A recent research proposes an architecture called TRACER[18], which makes use of a weighted loss that discriminates the object edges. Their approach consists of three parts: masked edge attention module (MEAM), union attention module and object attention module. It includes a four block encoder with EfficientNet backbone and a decoder. Edge detection is done in the MEAM using the output of the first convolution layer in an encoder architecture and applying Fast Fourier Transform to get the explicit object boundary. The union module integrates three remaining encoder blocks to detect the more important context from both channel and spatial representations of the image. The object attention module extracts the object and its complementary edge information. This approach significantly outperforms previous SOTA approaches both in accuracy, as measured by currently used metrics, and computational efficiency, since it removes a lot of redundant information and computations.

Chapter 4

Methodologies

4.1 Segmentation models

As mentioned in the Background chapter, we use an encoder-decoder architecture for salient object segmentation.

4.1.1 U-Net

U-Net is a fully convolutional model proposed by Ronneberger, P.Fischer, and Brox [33] for segmentation of 2D medical images including segmentation of neuronal structures in EM stacks and cell segmentation in light microscopy images. U-Net draws inspiration from FCN [23], a network that introduced a convolution architecture with a pixel-to-pixel prediction for image segmentation. After its introduction in 2015, U-Net was used extensively for semantic segmentation in general and many architectures build upon it, including U-Net++, Vnet [26], U^2Net [31] and more.

U-Net is composed of an encoder and decoder with skip connections between them for saving low-level features of the image that would otherwise turn abstract during down sampling. Encoder is a contracting side that down samples the image to get the high level features. In order to turn the high level features back into a pixel mask that will be able to show the location of the salient objects, the decoder side up samples the output of the encoder. Up sampling doubles the size of each feature channel and halves the number of channels. To make sure that more fine-grained features get to the up sampled result, the output of each down sampling convolution step in the encoder gets concatenated to upsampled feature maps in the decoder (Figure 2.4). After each concatenation, two convolution layers are applied to help concatenated data merge better with up sampled data, and the number of channels is halved. This way the contracting and expanding side become virtually symmetrical.

U-Net was created with dataset constraints in mind, since medical images often were and still are hard to collect and take a long time to get access to, so they heavily augmented the data with the use of elastic deformations.

4.1.2 U-Net++

In 2018, Z. Zhou, et al [55] introduced U-Net++, a model based on U-Net that tries to achieve semantic closeness between the feature maps from the encoder and the decoder before they are fused together using skip connections. It does this by putting dense convolution blocks on the path of the encoder feature maps. As shown in 6.1, each up sampled feature map in the decoder receives both features from its

level in the encoder that go through a series of dense convolutions and up sampled features from all down sampled maps that go after the corresponding encoder feature map. The up sampled features are concatenated before each consecutive convolution layer. U-Net++ shows an improvement in IOU metric on four different medical image segmentation datasets and problems.

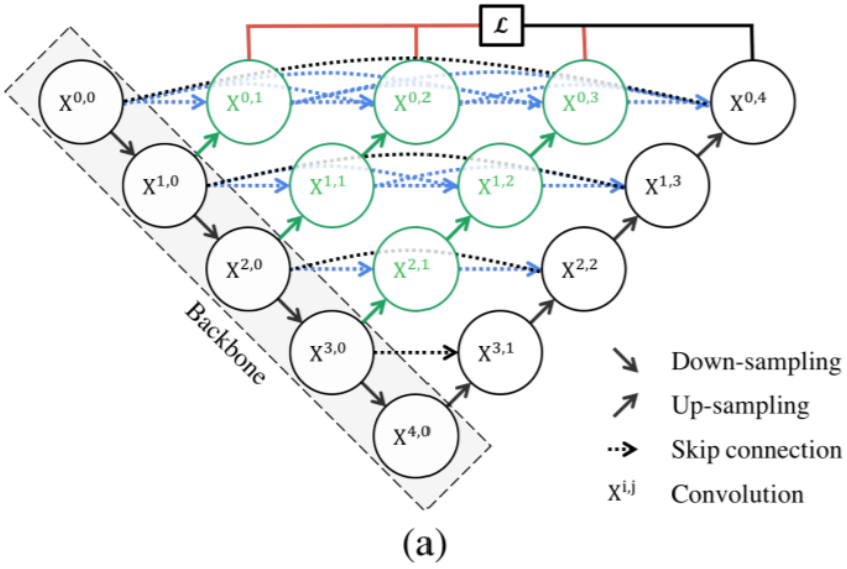

FIGURE 4.1: Architecture of U-Net++

4.2 Encoders

As a building block for the encoder part of the model, we use a pre-trained backbone. Pre-training the weights on ImageNet dataset helps to speed up the training process a lot, and using a feature extractor suited for mobile devices instead of original U-Net encoder modules should give us compact model size while simultaneously giving better accuracy. Thus, we will try MobileNetV2 and EfficientNet, two architectures that promise these things.

4.2.1 MobileNetV2

MobileNetV2 is a feature extracting architecture designed for mobile devices. It uses depth-wise separable convolution instead of standard convolution, lowering the computational cost 8 or 9 times without strong accuracy loss. The main feature of the architecture is an inverted residual block with linear bottleneck, which maps a low dimensional input into low-dimensional feature map, with depth-wise convolution performed on an expanded input in the block.

4.2.2 EfficientNet

EfficientNet model family was created by Mingxing Tan and Quoc V. Le [38] as part of research on model scaling. They wanted to achieve efficient scaling of all three network dimensions - depth, width and image resolution. The authors proposed a compound scaling method that takes in a coefficient specifying the resources

available for use and selects coefficients for uniform scaling. By using these coefficients, you get the benefits of all three dimensions being scaled up or down. They also use their scaling method on MobileNet and others to show an improvement in efficiency. As for the baseline network which was used to show the effects of their scaling method, they used Neural Architecture Search to get the best results on both accuracy and FLOPS and the resulting architecture EfficientNet-B0 was born. EfficientNet also consists of MBConv blocks, which are an upgrade on the inverted residual block used in MobileNet.

4.3 Loss

During training and validation, we use a combination of losses adopted by [30]. At the validation step, this loss combination determines the efficiency and is used to tune the learning rate and perform early stopping when the loss stops decreasing.

4.3.1 Binary Cross Entropy

BCE, or log loss, is widely used for binary segmentation and is defined as

$$l_{bce} = -\sum_{r,c}^{H,W} [G(r,c)log(S(r,c)) + (1 - G(r,c))log(1 - S(r,c))]$$

Here, r and c are indexes along the image height and width, G(r,c) is the ground truth label of the pixel and S(r,c) is the label of it on the predicted saliency map. This is a pixel-wise loss, treating both background and salient object predictions equally across the image.

4.3.2 Jaccard / IoU

Jaccard, or IOU loss is often used for training as well. It is a measure of similarity between two sets of points. Since in our case we need to compare ground truth points and the saliency probabilities, we use a soft IOU loss version defined in [25] as

$$l_{soft_iou} = 1 - \frac{\sum_{r,c}^{H,W} S(r,c)G(r,c)}{\sum_{r,c}^{H,W} [S(r,c) + G(r,c) - S(r,c)G(r,c)]}$$

4.3.3 L1 Loss

L1 loss measures the accuracy by summing all the absolute differences between the ground-truth and predicted values.

$$L1_{loss} = \sum_{i=1}^{n} |y_{true} - y_{predicted}|$$

4.3.4 SSIM

Structural Similarity Measurement was introduced in [44] for image quality assessment, based on the assumption that since human vision is adapted to extract structure from what they observe, one can say that an image with a good structure is of high quality. By comparing an image with a distorted version of it using structural similarity, they assess the quality of the latter.

This measure was successfully used by [30] with BASNet for patch-level loss evaluation of salient object detection. The loss tries to capture structural similarity of the ground-truth mask and the predicted mask. The structure similarity of a patch of an image can be defined as

$$SSIM(x, y) = \frac{(2\mu_x\mu_y + C1)(2\sigma_{xy} + C2)}{(\mu_x^2 + \mu_y^2 + C1)(\sigma_x^2 + \sigma_y^2 + C2)}$$

Here, μx, μy and σ_x, σ_y are the mean and variance of x (a patch from the ground truth mask) and y (patch from prediction map), $\sigma_(xy)$ is covariance, $C1 = 0.01^2$ and $C2 = 0.03^2$. The universal image quality index suggests they be 0, so these values help avoid unstable results in the denominator. The formula compares luminance, contrast and structure of the image, since mu_x and σ_x can be interpreted as the luminance and contrast of x, respectively, and the covariance of ground truth patch and predicted patch measures the tendency of them to vary together, thus measures structural similarity.

This formula is applied in a sliding window approach across the image, then the mean is calculated and used as a structure similarity assessment (or image quality, in the image quality assessment problem). We use this formula as a loss by subtracting the mean from 1.

Chapter 5

Datasets

5.1 Eye-fixations vs segmented objects/areas

There are numerous datasets created for saliency segmentation or salient object detection. As earlier methods focused more on salient pixels or regions, earlier datasets collected human eye-tracking information [15] as well as mouse tracking that tries to emulate human viewing behaviour[14]. They usually contained a couple hundred or thousand images, with 5-10 respondents whose eye attention was being tracked. A big problem with eye-tracker information is center bias [20], where human gaze is often thought to be biased towards the center of the image, which creates false points if there are no salient objects there. This problem does not disappear completely even with other dataset types, since photographers typically put the object to the center of the image as well.

More recently, datasets became more focused on objects that humans notice most rather than tracking separate pixel regions where people look. Since objects contain biggest visual stimuli and in the case of prior knowledge, the human eye is most likely searching for objects as well, it makes sense to try to predict object boundaries and preserve the wholeness of the salient thing the hypothetical person is concentrated on. The typical dataset used today contains a set of images with one or more salient regions or objects and a set of binary masks that classify each pixel of the image into two categories: salient and non-salient. This approach to data collecting can get rid of pixel outliers that come with eye tracking, as well as center bias. In order to prevent biasedly designed [20] image sets, SOD datasets sometimes consist of images from pre-existing large datasets. Datasets include images with complex crowded backgrounds, occluded objects, low contrast or multiple salient objects per image.

5.2 Pixel-wise Datasets

Most datasets we will use present a pixel-wise binary mask with one class being a salient object area (pixels are valued as 255) and the other - non-salient area, or background (pixels are valued as 0). One exception is PASCAL-S, which, when an image has multiple salient object areas, annotates each of them with a number between 0 and 255, creating a relative degree of saliency.

5.2.1 ECSSD

ECSSD [37] is a challenging dataset that contains 1000 images gathered from the internet that have complex multipart, structured, or textured backgrounds and objects. The difference between the background and salient object pixel distributions

in the CIElab color space is low (compared to the MSRA-1000 [1]), making it harder for the model to separate the two.

5.2.2 DUT-OMRON

DUT-OMRON [51] is a 5,168 image dataset, manually picked out of 140.000 images. It contains three different image annotations: eye fixations data, salient object pixel-wise masks and bounding boxes. 5 subjects per image were used to make the annotations. The resulting average bounding boxes were also used to remove outliers from eye-tracker points. Eye-tracker data was reported by the authors to have a center bias.

5.2.3 HKU-IS

HKU-IS [19], a 4,447 images dataset, is also challenging, with images of low color contrast, salient objects that are mostly either multiple, disconnected or touch the image boundary. These features were specifically formulated in contrast to earlier MSRA-B [15], first large dataset with pixel-wise salient object masks that had both a center bias and 98% of image border pixels containing no salient objects. HKU-IS was annotated by 3 subjects and images that had an inter-subject label inconsistency larger than 10% were left out.

5.2.4 PASCAL-S

Dataset PASCAL-S [20] comprises of 850 images from PASCAL VOC 2010 [11] validation subset. It contains both pixel-wise masks of salient objects, which are created by 12 subjects highlighting objects on images, pre-segmented into 540 category classes by [29], and a 2s per image eye-tracker data gathered from 8 subjects. The saliency value of each object is determined by the percentage of the human subjects that marked it as salient.

5.2.5 DUTS

DUTS [41] is the biggest train/test divided dataset for salient object detection. Its 10,553 training images (DUTS-TR) come from the train/val subset of famous ImageNet [10], and 5,019 testing images (DUTS-TE) are from ImageNet test set as well as SUN [48], which is a dataset for scene understanding. Images were labelled by 50 subjects, and contain challenging scenarios. DUTS-TR is often the choice for model training, and DUTS-TE is used along with other previously mentioned datasets to evaluate the model. We will also follow this approach.

5.3 Interesting data cases

Overall, after reviewing the datasets used for this task, we came to the conclusion that the definition of saliency is hard to perceive. The dataset creators usually reported that the saliency labels were consistent among different subjects who were providing their visual attention. Still, some results seem to be visually contradicting (our assumption here). For example, images from DUTS-TE in figure 5.1. If these masks truly represent the average salient representation of these images, a model that can correctly determine such cases can probably tell more about the definition of saliency than an average human subject.

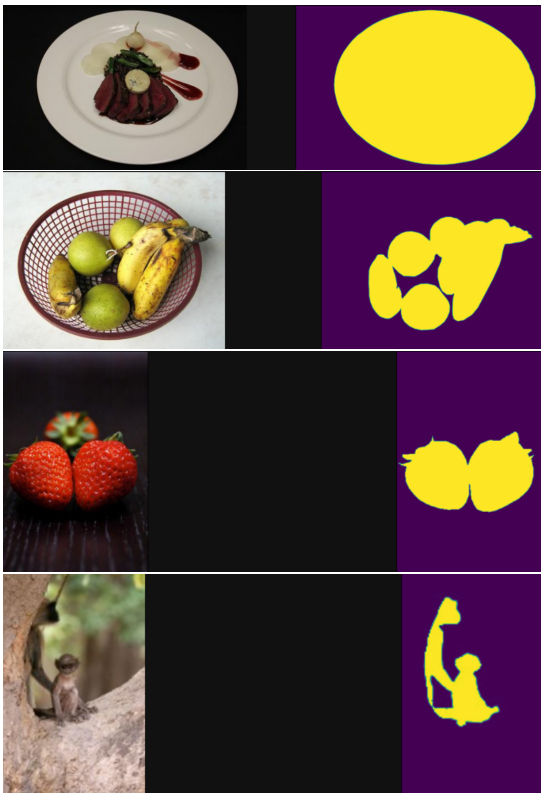

FIGURE 5.1: DUTS-TE images and masks. First and second picture both have food as salient region, ans both have food containers as semi-salient (visually) but are marked differently. Third and fourth pictures have a foreground character (strawberry and monkey) and a background character (third strawberry and mother-monkey) but are marked differently.

Also, an approach to data annotation taken by [29] seems to be superior in cases where an image has more than one salient object. Since giving both of the objects on images in figure 5.2 would ignore the fact that they are not equally the most unique and important in the image, the saliency was only assigned to one of them. In the cases such as these, it seems like giving the model relative saliency of the excluded objects would be better than giving none at all. And same can be said about giving equal degree of saliency to different objects here.

5.3.1 Possible improvement

When taking into account the possible applications of the salient object detection, e.g. image cropping, compression, adaptive image display on small screens, our point can be illustrated by cropped images from DUTS-TE and PASCAL-S in figure 5.2. A cropped area that is the result of a ground truth mask in DUTS-TE seems to be of lesser quality than if we mark the object left out by the dataset image mask as salient instead and then do the crop.

Another point can be made for the degree of saliency of the object. Saliency that is relative to the other salient objects does not necessarily reflect it, rather a saliency relative to the image overall. The eye-fixations data that is usually not used for object detection could potentially give us a more exact measurement of this. DUT-OMRON dataset has collected both eye-fixations and ground-truth binary masks, as well as bounding boxes for separate objects. The boxes can have a relative degree of saliency

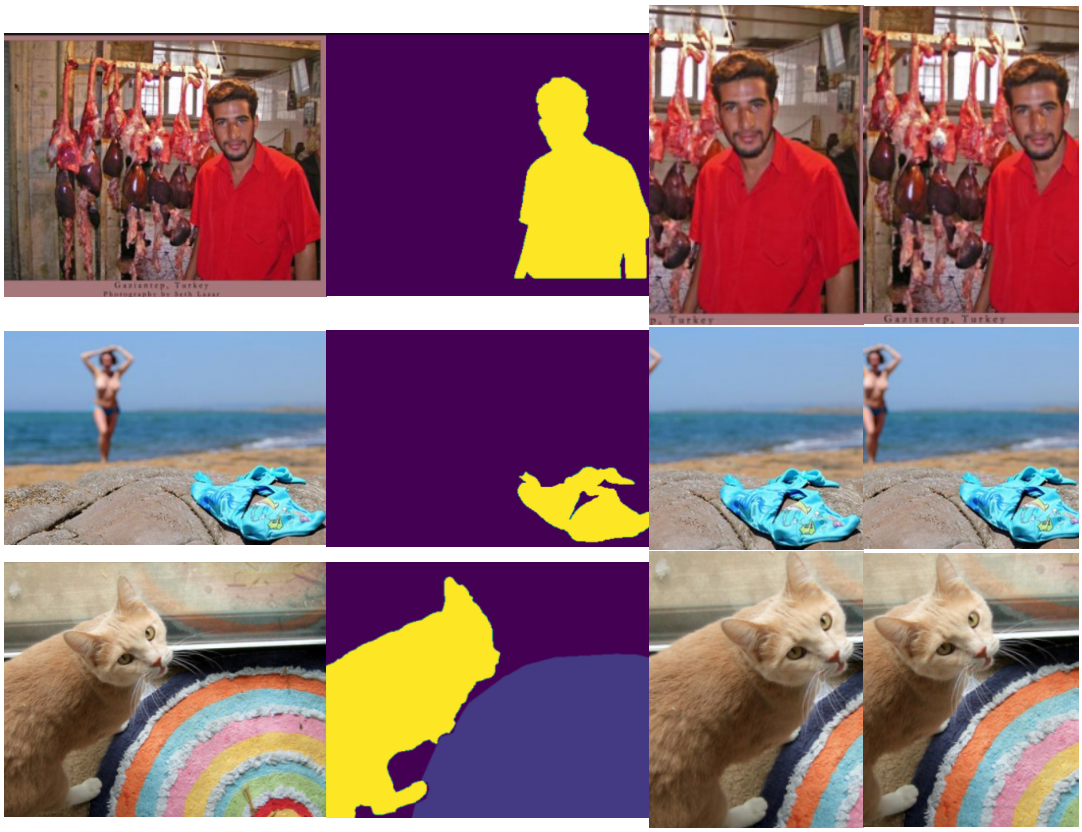

FIGURE 5.2: First two rows are images and saliency masks from DUTS-TE, third row is from PASCAL-S. The first crop (third column) is based on the saliency map in DUTS-TE and for PASCAL-S, we presume that only the cat was categorized as salient. The second crop is based on the assumption that the other object in the image would be also highlighted as salient (which is already the case with PASCAL-S).

since they gathered multiple boxes from multiple participants. The boxes can also help distinguish objects on the monolithic binary masks, although the result would be of less quality than if the masks themselves were presented like in PASCAL-S.

Overall, some intersection of what DUT-OMRON and PASCAL-S provide can possibly be utilized to create an approach with a saliency percent given to each object in the image. This can be a project for future work.

Chapter 6

Experiments and Results

6.1 Implementation details

The experiments were performed on Google Cloud instances with Tesla T4 GPUs. We used a Pytorch library of common segmentation models [49] to train an encoder-decoder with a pre-trained backbone to utilize the feature maps learned on ImageNet dataset. DUTS-TR dataset was used for training and validation. We used the hold-out technique, making a 8:2 / 17:3 split of DUTS-TR that resulted in 8.442 / 8.970 images for training and 2.111 / 1.583 for validation. The training images were augmented using a series of transformations from the Albumentations library [6].

The augmentations included:

- Horizontal and vertical flipping
- Scaling and rotation
- Optical and grid distortion of the input, as well as elastic transformation
- Adding Gaussian noise
- Random brightness and contrast change, as well as CLAHE - Contrast Limited Adaptive Histogram Equalization, which equalizes the image
- In the end, a random crop was performed to the 256x256 size

For validation and later for testing, images were simply resized to 256x256, without any post-processing.

The loss used for training was a combination of BCE, Jaccard, L1 and SSIM loss. The optimizer of choice was Adam, with a learning rate of 0.0001. Validation loss was used to decrease the learning rate by 10 after 5 stale epochs, and to perform an early stopping after 11.

6.2 Evaluation metrics

We use five most commonly used benchmark datasets DUTS-TE, DUT-OMRON, PASCAL-S, ECSSD and HKU-IS to test the model. To evaluate the performance, we use MAE (Mean Absolute Error), maximum F-score, and S-measure (Structure-measure) as suggested in many SOTA approaches [31, 18, 45, 21, 30] to be fairly good metrics for this task. Also, we compare our resulting model with others on model size.

6.2.1 MAE

MAE [7] is the average of the absolute differences between ground-truth pixel values and the predicted probability values of the pixel being salient. It can be defined as follows

$$MAE = \frac{1}{H * W} \sum_{y=1}^{H} \sum_{x=1}^{W} |P(x,y) - G(x,y)|$$

Here P denotes the probability map of pixel of x width and y height and G - corresponding pixel of ground truth map. This metric looks at the percentage of erroneously labeled pixels relative to the image size, treating each error equally.

6.2.2 Max F-measure

Max F-measure is the average of the maximum F-measures computed for each image using a Precision-Recall curve. To build a Precision-Recall curve, we need to compare two pixel-wise binary masks. Since the saliency prediction map is continuous, we need to binarize it first, which is a separate important step in the research. There are several approaches to this, including SaliencyCut [8], which is based on GrabCut [35], an iterative process that uses Gaussian mixture models and graph cut for foreground extraction. Other approaches include mean-shift segmentation followed by averaging the saliency of the segments and using adaptive thresholding to binarize them [2].

We use a simpler approach [5] which takes a fixed threshold in the range 0-255 and creates a binary mask for each. By computing precision and recall at each threshold, we get a curve that shows the range of the performance of the model. Since neither of the two metrics is extensive enough, we can use a weighted harmonic mean of the two called F-measure, which prioritizes precision over recall by setting a coefficient β^2 to 0.3. The formula goes as follows

$$F_\beta = \frac{(1 + \beta^2) * Precision * Recall}{\beta^2 * Precision + Recall}$$

The maximum F-measure over the curve can be used to evaluate the model performance on the image. Averaged over the dataset, max F-measure is a metric that shows efficiency.

6.2.3 S-measure

Since previous measures both only care about overall pixel-wise efficiency, there is a need for a measure that will evaluate the structure of the predicted object area. For many applications of salient object detection, it is important to preserve the structure of the object. We will use a measure proposed by [12] called Structure-measure (or Structure Similarity Measure) which consists of two parts: region-aware structure similarity measure and object-aware.

The structure similarity of separate regions of the object is measured using an approach based on SSIM index [44], which was discussed here. The formula is as follows

$$Sr = \sum_{k=1}^{K} * w_k * SSIM(k)$$

K is the number of blocks the saliency mask is divided into. For each block, SSIM is computed and multiplied by a weight w_k that is determined by the amount of

ground truth object pixels that the block covers (to give importance to the blocks that contain the object).

As for the object-aware measure, it compares the global distributions of the salient object and the background. Since in a high-quality prediction map, these two regions' distributions will be relatively uniform, i.e. the foreground pixels should all have relatively equally predicted probabilities of being non-salient, and the object - salient. To assess the dispersion of the probabilities of the salient area as well as the foreground, it uses the coefficient of variation, i.e. the ratio of the standard deviation to the mean. Also, it compares the contrast between the salient and non-salient region by measuring the closeness of the mean probabilities of the two, using a formula based on the part of [44] responsible for luminance. By adding the two components, the formula for the similarity of the foreground, or salient area, is defined like this

$$O_{FG} = \frac{2 * \mu_{xFG}}{(\mu_{xFG})^2 + 1 + 2\lambda * \sigma_{xFG}}$$

The formula is the same for the background, except its pixels are subtracted from 1, since it is a complement to the salient area. O_{FG} and O_{BG} are then assigned weights according to the percent of image they occupy

The general formula, where α is set to 0.5 to treat both object-aware and region aware structure similarity equally, is

$$S = \alpha * S_o + (1 - \alpha) * S_r,$$

6.3 Results

We compare our results with several SOTA approaches in table 6.1 and 6.2. Overall, quantitatively, our best model is UNet++ with an EfficientNet encoder. There are two versions presented, differentiated only by the split of training and validation data. Since they gave very close and contradicting results on different datasets, we could not pick a winner. Usually, in MAE and maxF our two best results - $U++^{85/15}_{eff}$ and $U++^{80/20}_{eff}$ lie between the U^Net and U^Net_{small}, and in S-measure, which is for structure, our approaches could not reach the desired efficiency.

Datasets		DUTS-TE(5019)			DUT-OMRON(5168)			HKU-IS(4447)		
Model	Size(Mb)	F_m	MAE	S_m	F_m	MAE	S_m	F_m	MAE	S_m
BASNet	348.5	.860	.047	.853	.805	.056	.836	.928	.032	.909
U^2Net	176.3	.873	.044	.861	.823	.054	.847	.935	.031	.916
U^2Net_S	4.7	.852	.054	.847	.813	.060	.837	.928	.037	.908
$U++_{mob}$	29.1	.854	.052	.843	.807	.066	.805	.921	.042	.889
U_{mob}	27.7	.851	.053	.846	.802	.067	.805	.920	.041	.891
U_{eff}	27.7	.868	.049	.857	.814	.066	.815	.926	.041	.896
$U++^{85/15}_{eff}$	29.1	.871	.048	.859	.815	.066	.815	.929	.039	0.9
$U++^{80/20}_{eff}$	29.1	.872	.048	.862	.821	.066	.821	.928	.040	.898

TABLE 6.1: The results of several SOTA approaches and our approaches on bigger datasets. The last two approaches are both UNet++ with EfficientNet, but their train/val split was different, and they showed contradicting results on different datasets.

Qualitatively, it seems that $U++^{85/15}_{eff}$ is better, even when looking at pictures from DUTS-TE, where it is not superior. When doing qualitative comparison, there are cases for each model, where it does best, which shows that they are very close to

Datasets		ECSSD(1000)			PASCAL-S(850)		
Model	Size(Mb)	F_m	MAE	S_m	F_m	MAE	S_m
BASNet	348.5	.942	.037	.916	.856	.076	.838
U^2Net	176.3	.951	.033	.928	.859	.074	.844
U^2Net_S	4.7	.943	.041	.918	.849	.086	.831
U++$_{mob}$	29.1	.933	.050	.897	.844	.075	.835
U$_{mob}$	27.7	.933	.049	.900	.842	.074	.835
U$_{eff}$	27.7	.943	.044	.906	.848	.071	.842
U++$_{eff}^{85/15}$	29.1	.943	.044	.908	.851	.069	.847
U++$_{eff}^{80/20}$	29.1	.943	.046	.905	.850	.071	.845

TABLE 6.2: The results of several SOTA approaches and our approaches on smaller datasets. The last two approaches are both UNet++ with EfficientNet, but their train/val split was different, and they showed contradicting results on different datasets.

each other in their approach. However, it seems that MobileNet based approaches on average create clearer boundaries of the object, however the prediction itself is usually inferior to EfficientNet based. Also, even though U_{eff} seems to perform the worst on most images, there are cases when it is the best instead. $U++_{eff}^{80/20}$, which is quantitatively best on the DUTS-TE and DUT-OMRON, usually creates more false positives than $U++_{eff}^{85/15}$ or U_{eff}, and seems to be the only one that sometimes makes no confident predictions on a map at all.

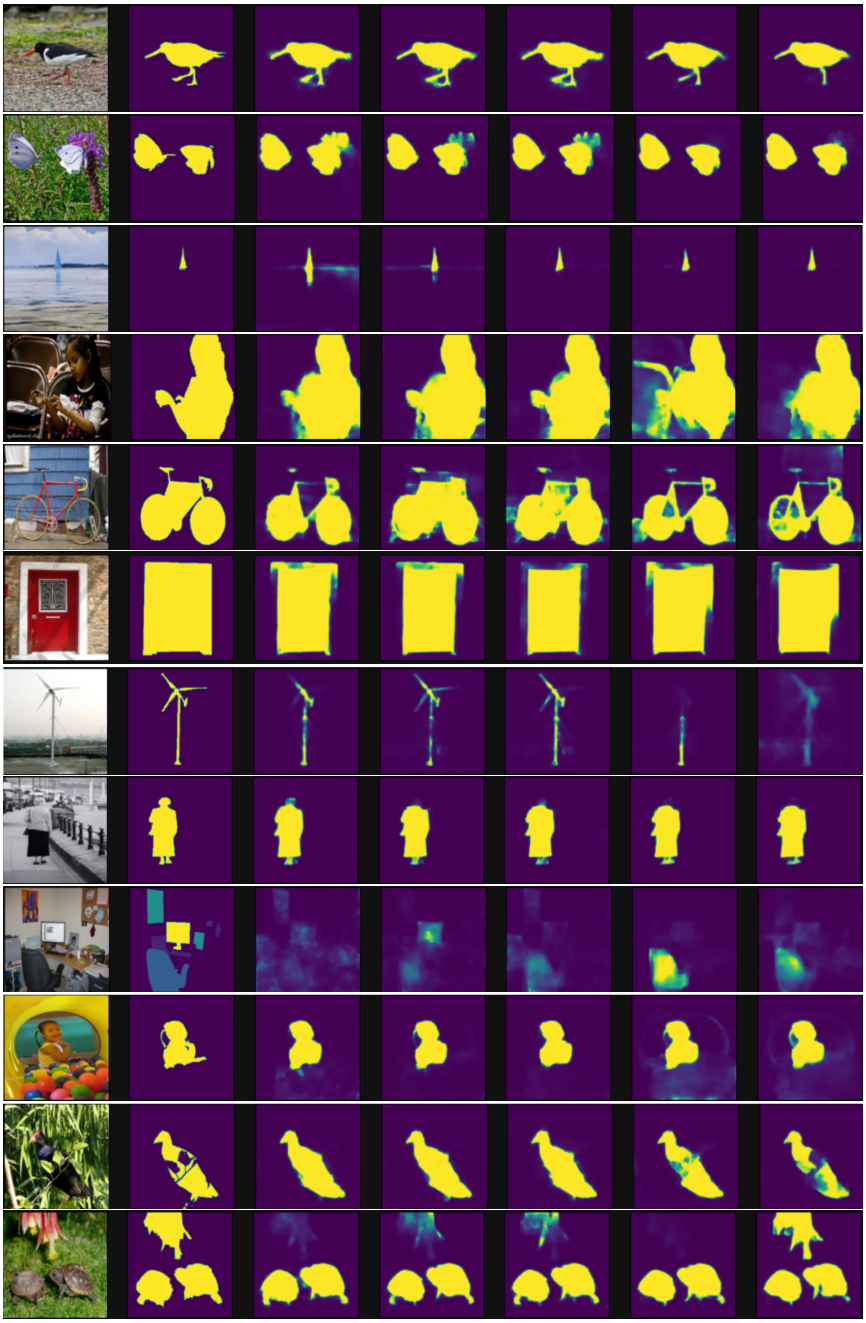

FIGURE 6.1: Our results on various datasets. From left to right: image, gt mask, U_{eff}, $U++^{85/15}_{eff}$, $U++^{80/20}_{eff}$, U_{mob}, $U++_{mob}$

Chapter 7

Conclusion and Future work

7.1 Conclusion

In this paper, we presented the well-known problem of salient object detection. The most popular approaches, methodologies and features used in solving this task were discussed, as well as the wide range of available data resources and some problems that come with them. We proposed an approach to solving this task that is fast, compact and achieves comparable results while being very simple and utilizing well-known and researched architectures. Pre-trained backbones that are used for general feature extraction proved to be very beneficial to the task. Different variations in our approach came very close in quantitative comparison, but it is clear from the visual examples that they behave somewhat differently. Overall, we achieved comparable results to some recent SOTA approaches that create a more complex algorithm than simply utilizing proven architectures for this specific task.

7.2 Future work

There are some areas where we could improve upon what we did, or go onto a next step in salient object detection.

7.2.1 Binarization of the prediction

To finish the binary segmentation of images into salient objects and the background, we have to choose the best way to transform the prediction map, which was created in our approach, into a binarized mask. There are several possible ways to do it, which were touched upon briefly here. We will try the SaliencyCut [8] approach as well as search for other methods.

7.2.2 S-measure based loss and weighted loss

Also, we discussed SSIM index here and Structure-measure here, both used to detect the structure similarity of the prediction and the ground truth. Since we received did not receive high enough results when evaluating the model on Structure-measure, even though a SSIM loss was applied to assist the model, it seems that it is not sufficient enough of a loss. Structure-measure builds upon SSIM for the specific task of saliency detection, so it would be interesting to apply it as a loss to train the model and see if it will bring better results.

Also, it could be beneficial to try a weighted loss like in [18] that gives more importance to the edges of the objects to make the model perform better in terms of finding thin subparts of the object or overall, get better at knowing where the object's boundary is.

7.2.3 Object saliency percentage measure relative to the image

Perhaps the most interesting idea for future work was discussed here. Since the binary representation in most current salient object datasets presents some constraints on the saliency measure of the objects, it would be interesting to give the network soft ground-truth masks. This might help utilize the salient segmentation in further applications such as image cropping, because we will have more inclusive information as to what can be cropped and what can stay when there is more than one salient object in the image. Currently, the soft ground-truth masks do not exist, as far as we know, but an attempt can be made to create them out of DUT-OMRON data or by creating a new dataset.

Bibliography

[1] Radhakrishna Achanta et al. “Frequency-tuned salient region detection”. In: *CVPR 2009*. 2009.

[2] Radhakrishna Achanta et al. “Frequency-tuned salient region detection”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition*. 2009, pp. 1597–1604. DOI: 10.1109/CVPR.2009.5206596.

[3] A. Andrushia and R. Thangarajan. “Saliency-Based Image Compression Using Walsh–Hadamard Transform (WHT)”. In: Jan. 2018, pp. 21–42. ISBN: 978-3-319-61315-4. DOI: 10.1007/978-3-319-61316-1_2.

[4] Edoardo Ardizzone, Alessandro Bruno, and Giuseppe Mazzola. “Saliency Based Image Cropping”. In: vol. 8156. Sept. 2013, pp. 773–782. ISBN: 978-3-642-41180-9. DOI: 10.1007/978-3-642-41181-6_78.

[5] Ali Borji et al. “Salient Object Detection: A Benchmark”. In: *IEEE Transactions on Image Processing* 24.12 (2015), pp. 5706–5722. DOI: 10.1109/TIP.2015.2487833.

[6] Alexander Buslaev et al. “Albumentations: Fast and Flexible Image Augmentations”. In: *Information* 11.2 (2020). ISSN: 2078-2489. DOI: 10.3390/info11020125. URL: https://www.mdpi.com/2078-2489/11/2/125.

[7] Zuyao Chen et al. “Global Context-Aware Progressive Aggregation Network for Salient Object Detection”. In: (2020). DOI: 10.48550/ARXIV.2003.00651. URL: https://arxiv.org/abs/2003.00651.

[8] Ming-Ming Cheng et al. “Global Contrast Based Salient Region Detection”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 37 (2015), pp. 569–582.

[9] Céline Craye, David Filliat, and Jean-François Goudou. “Environment exploration for object-based visual saliency learning”. In: *2016 IEEE International Conference on Robotics and Automation (ICRA)* (2016), pp. 2303–2309.

[10] J. Deng et al. “ImageNet: A large-scale hierarchical image database”. In: *2009 IEEE Conference on Computer Vision and Pattern Recognition* (2009), pp. 248–255.

[11] Mark Everingham et al. “The Pascal Visual Object Classes (VOC) challenge”. In: *International Journal of Computer Vision* 88 (June 2010), pp. 303–338. DOI: 10.1007/s11263-009-0275-4.

[12] Deng-Ping Fan et al. “Structure-Measure: A New Way to Evaluate Foreground Maps”. In: *2017 IEEE International Conference on Computer Vision (ICCV)*. 2017, pp. 4558–4567. DOI: 10.1109/ICCV.2017.487.

[13] Qibin Hou et al. “Deeply Supervised Salient Object Detection with Short Connections”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 41.4 (2019), pp. 815–828. DOI: 10.1109/tpami.2018.2815688. URL: https://doi.org/10.1109%2Ftpami.2018.2815688.

[14] Md. Amirul Islam et al. “Salient Object Detection using a Context-Aware Refinement Network”. In: *BMVC*. 2017.

[15] Huaizu Jiang et al. “Salient Object Detection: A Discriminative Regional Feature Integration Approach”. In: *International Journal of Computer Vision* 123 (2013), pp. 251–268.

[16] Mathias Kraus, Stefan Feuerriegel, and Asil Oztekin. “Deep learning in business analytics and operations research: Models, applications and managerial implications”. In: *European Journal of Operational Research* 281 (Sept. 2019). DOI: 10.1016/j.ejor.2019.09.018.

[17] Jason Kuen, Zhenhua Wang, and Gang Wang. “Recurrent Attentional Networks for Saliency Detection”. In: *CoRR* abs/1604.03227 (2016). arXiv: 1604.03227. URL: http://arxiv.org/abs/1604.03227.

[18] Min Seok Lee, WooSeok Shin, and Sung Won Han. *TRACER: Extreme Attention Guided Salient Object Tracing Network*. 2021. DOI: 10.48550/ARXIV.2112.07380. URL: https://arxiv.org/abs/2112.07380.

[19] Guanbin Li and Yizhou Yu. “Visual Saliency Based on Multiscale Deep Features”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2015.

[20] Yin Li et al. “The Secrets of Salient Object Segmentation”. In: *CVPR, 2014, IEEE Conference on*. 2014.

[21] Jiang-Jiang Liu et al. “A Simple Pooling-Based Design for Real-Time Salient Object Detection”. In: *IEEE CVPR*. 2019.

[22] Nian Liu and Junwei Han. “DHSNet: Deep Hierarchical Saliency Network for Salient Object Detection”. In: *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* (2016), pp. 678–686.

[23] Jonathan Long, Evan Shelhamer, and Trevor Darrell. “Fully convolutional networks for semantic segmentation”. In: June 2015, pp. 3431–3440. DOI: 10.1109/CVPR.2015.7298965.

[24] David Lowe. “Distinctive Image Features from Scale-Invariant Keypoints”. In: *International Journal of Computer Vision* 60 (Nov. 2004), pp. 91–. DOI: 10.1023/B:VISI.0000029664.99615.94.

[25] Gellért Máttyus, Wenjie Luo, and Raquel Urtasun. “DeepRoadMapper: Extracting Road Topology from Aerial Images”. In: *2017 IEEE International Conference on Computer Vision (ICCV)* (2017), pp. 3458–3466.

[26] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. “V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation”. In: *2016 Fourth International Conference on 3D Vision (3DV)*. 2016, pp. 565–571. DOI: 10.1109/3DV.2016.79.

[27] Frank Moosmann, Diane Larlus, and Frederic Jurie. “Learning saliency maps for object categorization”. In: (May 2006).

[28] Frank Moosmann, Diane Larlus, and Frederic Jurie. “Learning saliency maps for object categorization”. In: (May 2006).

[29] Roozbeh Mottaghi et al. “The Role of Context for Object Detection and Semantic Segmentation in the Wild”. In: June 2013. DOI: 10.13140/2.1.2577.6000.

[30] Xuebin Qin et al. “BASNet: Boundary-Aware Salient Object Detection”. In: *2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 7471–7481. DOI: 10.1109/CVPR.2019.00766.

[31] Xuebin Qin et al. “U2-Net: Going deeper with nested U-structure for salient object detection”. In: *Pattern Recognition* 106 (Oct. 2020), p. 107404. DOI: 10.1016/j.patcog.2020.107404.

[32] Rodolfo Quispe and Helio Pedrini. “Improved Person Re-Identification Based on Saliency and Semantic Parsing with Deep Neural Network Models”. In: *Image and Vision Computing* 92 (Sept. 2019). DOI: 10.1016/j.imavis.2019.07.009.

[33] O. Ronneberger, P.Fischer, and T. Brox. “U-Net: Convolutional Networks for Biomedical Image Segmentation”. In: *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*. Vol. 9351. LNCS. (available on arXiv:1505.04597 [cs.CV]). Springer, 2015, pp. 234–241. URL: http://lmb.informatik.uni-freiburg.de/Publications/2015/RFB15a.

[34] Paul Rosin. “A simple method for detecting salient regions”. In: *Pattern Recognition* 42 (Nov. 2009), pp. 2363–2371. DOI: 10.1016/j.patcog.2009.04.021.

[35] Carsten Rother, Vladimir Kolmogorov, and Andrew Blake. “"GrabCut": interactive foreground extraction using iterated graph cuts”. In: *ACM SIGGRAPH 2004 Papers* (2004).

[36] Mark Sandler et al. “Inverted Residuals and Linear Bottlenecks: Mobile Networks for Classification, Detection and Segmentation”. In: *CoRR* abs/1801.04381 (2018). arXiv: 1801.04381. URL: http://arxiv.org/abs/1801.04381.

[37] Jianping Shi et al. “Hierarchical Image Saliency Detection on Extended CSSD”. In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 38.4 (2016), pp. 717–729. DOI: 10.1109/TPAMI.2015.2465960.

[38] Mingxing Tan and Quoc V. Le. “EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks”. In: *CoRR* abs/1905.11946 (2019). arXiv: 1905.11946. URL: http://arxiv.org/abs/1905.11946.

[39] T. N. Vikram, M. Tscherepanow, and B. Wrede. “A saliency map based on sampling an image into random rectangular regions of interest”. In: *Pattern Recognit.* 45 (2012), pp. 3114–3124.

[40] Tadmeri Vikram, Marko Tscherepanow, and Britta Wrede. “A random center surround bottom up visual attention model useful for salient region detection”. In: Jan. 2011, pp. 166–173. DOI: 10.1109/WACV.2011.5711499.

[41] Lijun Wang et al. “Learning to Detect Salient Objects With Image-Level Supervision”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2017.

[42] Linzhao Wang et al. “Saliency Detection with Recurrent Fully Convolutional Networks”. In: *ECCV*. 2016.

[43] Tiantian Wang et al. “Detect Globally, Refine Locally: A Novel Approach to Saliency Detection”. In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 3127–3135.

[44] Z. Wang, E.P. Simoncelli, and A.C. Bovik. “Multiscale structural similarity for image quality assessment”. In: *The Thrity-Seventh Asilomar Conference on Signals, Systems Computers, 2003*. Vol. 2. 2003, 1398–1402 Vol.2. DOI: 10.1109/ACSSC.2003.1292216.

[45] Jun Wei et al. “Label Decoupling Framework for Salient Object Detection”. In: *CoRR* abs/2008.11048 (2020). arXiv: 2008.11048. URL: https://arxiv.org/abs/2008.11048.

[46] Yichen Wei et al. "Geodesic Saliency Using Background Priors". In: *ECCV*. 2012.

[47] Zhe Wu, Li Su, and Qingming Huang. "Cascaded Partial Decoder for Fast and Accurate Salient Object Detection". In: *CoRR* abs/1904.08739 (2019). arXiv: 1904.08739. URL: http://arxiv.org/abs/1904.08739.

[48] Jianxiong Xiao et al. "SUN database: Large-scale scene recognition from abbey to zoo". In: June 2010, pp. 3485–3492. DOI: 10.1109/CVPR.2010.5539970.

[49] Pavel Yakubovskiy. *Segmentation Models Pytorch*. https://github.com/qubvel/segmentation_models.pytorch. 2020.

[50] Chuan Yang et al. "Saliency Detection via Graph-Based Manifold Ranking". In: June 2013, pp. 3166–3173. DOI: 10.1109/CVPR.2013.407.

[51] Chuan Yang et al. "Saliency Detection via Graph-Based Manifold Ranking". In: *2013 IEEE Conference on Computer Vision and Pattern Recognition*. 2013, pp. 3166–3173. DOI: 10.1109/CVPR.2013.407.

[52] Jimei Yang and Ming-Hsuan Yang. "Top-Down Visual Saliency via Joint CRF and Dictionary Learning". In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39.3 (2017), pp. 576–588. DOI: 10.1109/TPAMI.2016.2547384.

[53] Lu Zhang et al. "A Bi-Directional Message Passing Model for Salient Object Detection". In: June 2018, pp. 1741–1750. DOI: 10.1109/CVPR.2018.00187.

[54] Xiaoning Zhang et al. "Progressive Attention Guided Recurrent Network for Salient Object Detection". In: *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2018), pp. 714–722.

[55] Zongwei Zhou et al. "UNet++: A Nested U-Net Architecture for Medical Image Segmentation". In: July 2018.

[56] Fabio Zund et al. "Content-aware compression using saliency-driven image retargeting". In: Sept. 2013, pp. 1845–1849. ISBN: 978-1-4799-2341-0. DOI: 10.1109/ICIP.2013.6738380.