National College of Ireland

Synthetic Defocus Generation using Deep Learning

MSc Research Project
MSc. Data Analytics

Rutvik Mendjoge
Student ID: 20127855

School of Computing
National College of Ireland

Supervisor: Dr. Catherine Mulwa

National College of Ireland
Project Submission Sheet
School of Computing

Student Name:	Rutvik Mendjoge
Student ID:	20127855
Programme:	MSc. Data Analytics
Year:	2021
Module:	MSC Research Project
Supervisor:	Dr. Catherine Mulwa
Submission Due Date:	16/08/2021
Project Title:	Synthetic Defocus Generation using Deep Learning
Word Count:	6590
Page Count:	23

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:	
Date:	16th August 2021

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST:

Attach a completed copy of this sheet to each project (including multiple copies).	☐
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	☐
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	☐

Assignments that are submitted to the Programme Coordinator office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Synthetic Defocus Generation using Deep Learning

Rutvik Mendjoge
20127855

Abstract

Synthetic Defocus (Bokeh) is a common photographic technique for displaying in-focus objects. It focuses on the main subject of a picture while defocusing on all other objects. DSLR cameras are mostly used for this purpose. To capture bokeh, a smaller aperture is required. Many cellphones can now shoot this image as a result of technical advancements. The method used in this study is to achieve synthetic defocus on existing images. Using deep learning techniques, the concept can be accomplished. To have a comprehensive analysis of the image, it is necessary to examine all of the factors, therefore the idea of Deep Neural Networks is emphasized. DeepLab, Mask-RCNN, and Xception are three Deep Learning Algorithms that were used on the EBB Data set. Three evaluation metrics, such as PSNR, SSIM, and MSE, were used to evaluate these algorithms. DeepLab excelled in edge detection with smooth edges and excellent PSNR values when compared with the state of the art algorithms.

1 Introduction

Bokeh is a common photographic technique that emphasizes an image's subject. It provides the sense that the in-focus topics are more important than the rest of the image. That is why the human eye is drawn to Bokeh, or Depth of Field. A large lens on a DSLR camera is typically used to produce this bokeh effect. The aperture increases with the size of the lens. The focal length of the camera is used to calculate the subject's focus length. The higher the Depth of Field, the longer the focal length. A lens with a bigger aperture may capture more light in a photograph. Aperture, focal length, lens distance from the ground, and object distance are all variables that allow the human eye to discern between the subject's foreground and background.

Due to breakthrough technologies such as twin cameras and multi-image mounting, new smartphone cameras can compete with traditional cameras in many ways. When you consider how portable cell phones are, it's no surprise that they've become quite popular among amateur documentary, wildlife, and street photographers. And, on occasion, certain seasoned photographers and journalists carry the device in their wallets for paid jobs. Because of their small image sensor, smartphones fail to blur the background in photos, resulting in an almost limitless depth of field, rendering the background almost as clear as the subject.

Most smartphones have a 4 mm focal length. Since the focal length is the distance between the lens and the image sensor, there can't be any more. The depth of field is enhanced with a shorter focal length, making everything in the picture crisp. 4 mm is an extremely small focal length. In full-frame cameras, smaller than 25 mm is uncommon

(Cristofalo and Wang; 2017). These 4 mm are so tiny that a large aperture (e.g. f/1.6) on the smartphone wouldn't make much of a difference. Even so, the backdrop wouldn't be particularly hazy. Photographing the subject up up and personal is one technique to make an out-of-focus background look natural.

Another approach to accomplish this is to use computational methods. Synthetic segmentation is a method for distinguishing foreground from background. Many academics have used Image Processing and Deep Learning to try to tackle this challenge. Literature Review discusses previous work on these two approaches. Deep Learning techniques includes the use of sementic segmentation algorithms. The objective here is to expand the use of huge datasets for in-depth analysis of various types of pictures. The dataset includes photographs taken with and without the bokeh effect (Qian et al.; 2020). The goal is to use both types of pictures to train the deep learning model and obtain the desired result. In research methodology and specifications, a more thorough set of objectives and methods is described. Various research on bokeh have been done to yet, however a few aspects have still to be investigated.

There are several implementations on this topic, but it lacks in few domains. i) No Precise Edge Detection. ii) Unable to locate the objects in an image, hence end up blurring the subject in an image. iii) High execution time. These problems are addressed and improved in this paper by implementation of three deep learning algorithms such as DeepLab, Mask-RCNN and Xception. It's very hard to match the efficiency and consistency of images captured by traditional DSLR cameras. However, newer smartphones has almost matched the quality of the synthetically defocused (Bokeh) images (Luo, Li, Lin, Chen, Lee, Choi, Yoo and Polley; 2020). The approach presented in this paper can be implemented on a mobile device and used to generate bokeh images in post-processing (Singh et al.; 2018). Having photography as a hobby and seeing things and nature as important components of every shot has led to the consideration of other aspects.

1.1 Research Question

RQ: *"How can synthetically defocused images generated by deep learning algorithms (DeepLab, Mask-RCNN and Xception) improve the photography by enhancing the image quality?"*

Generating Synthetic defocus images using Deep Learning is challenging. There are few state of the art Deep Learning algorithms which are successful in generating these images.

Sub-RQ: *"How can synthetically defocused images generated by deep learning algorithms compete with the efficiency and consistency of images acquired with a DSLR camera?"*

To improve the efficiency of the computationally generated bokeh images.

1.2 Research Objectives

Objective 1: Involves investigation and review of literature of synthetic defocus generation research projects from 2011-2020.
Objective 2: Implementation, Evaluation and Results of the segmentation models using deep learning techniques.

Obj 2.1: Implementation, Evaluation and Results of DeepLab.

Obj 2.2: Implementation, Evaluation and Results of Mask-RCNN.

Obj 2.3: Implementation, Evaluation and Results of Xception.
Objective 3: Comparison of Developed Models.
Objective 4: Comparison of Developed Models with the Existing Models.

The following is the format of the paper's structure. Section 2 discusses the models, feature extraction, and Synthetic Defocus Generation using Deep Learning, while Section 3 discusses the methodology approach in the field of Synthetic Defocus Generation. Design Specification is covered in Section 4. Implementation of Deep Learning Algorithms is covered in Section 5. Section 6 contains the evaluation and findings acquired after pre-processing the data, as well as the results gained after training the model. The Conclusion and Future Work are found in Section 7.

2 Related Work

2.1 Computational Photography and Object Detection

The digital camera has ushered in a paradigm change in photography, displacing more than 150 years of technology based on the strange and beautiful photochemistry of silver halide crystals. Surprisingly, the camera has remained fairly same throughout the transition. Imaging labs are working with cameras that don't just digitize a picture but also conduct complex computations on it. Some of the studies are aimed at improving or supplementing conventional photographic techniques, such as extending the dynamic range of a picture or increasing the depth of field (Adams et al.; 2010). Other advancements would allow the photographer to have more control over variables such as motion blur. In most cameras, a patchwork pattern of red, green, and blue filters covers the sensor array, allowing each photo site to receive light in just one band of wavelengths. However, in the finished image, each pixel has all three-color components.

De-mosicing is a computer technique in which signals from neighboring photo sites are interpolated in various ways to give the pixels their hues (Raskar and Tumblin; 2005). A single pixel can include data from several of photo sites. A slicing method, which emphasizes edges and abrupt transitions, is likely to be used by the camera's image-processing computer (Sun et al.; 2021). It may also modify the contrast and color balance before compressing the data for easier storage. Given all of this internal processing, it appears that a digital camera is more than just a passive recorder. It does not capture photographs; rather, it creates them. When the sensor detects a pattern of illumination, it is merely the beginning of the image-making process.

As part of the process, there are so many techniques which needs to be followed in order to capture a perfect click. One of the most important technique when it comes to semantic segmentation is Object Detection (Papageorgiou et al.; 1998). Object detection is a method of recognizing and finding objects in pictures and movies using computer vision. Using this form of identification and localization, object detection may be used to count things in a scene, establish and monitor their precise locations, and appropriately label them. Object localization is the process of drawing a bounding box around one or more objects in an image, whereas image classification is the process of assigning a class label to an image. Object detection is a more challenging operation that combines the two by creating a bounding box around each object of interest in the image and labeling it with a class label. (Ren et al.; 2016) All of these difficulties are referred to as object recognition. As a result, object identification is critical for bokeh photography.

2.2 Edge Detection and Semantic Segmentation

Edge detection is an image processing approach that identifies places in a digital image that have discontinuities, or abrupt changes in image brightness (Wang; 2016). The edges of the picture are the locations where the image brightness changes dramatically. It is a fundamental stage in image processing, picture pattern recognition, and computer vision. Convolution methods come to our rescue when processing very high-resolution digital pictures.

In the evolution from coarse to fine inference, semantic segmentation is a natural step: The source of the problem might be found in classification, which entails creating a forecast for the entire input (Gidaris and Komodakis; 2015). The following stage is detection / localization, which provides not only the classes but also additional information about their geographical position. Finally, semantic segmentation accomplishes fine-grained inference by generating dense predictions inferring labels for each pixel, resulting in each pixel being labeled with the class of the object or region it surrounds. Unlike classification, where the end output of the extremely deep network is all that matters, semantic segmentation necessitates not just pixel-level discrimination but also a technique to project the discriminative features learned at various stages of the encoder onto the pixel space (Zhang et al.; 2018). As part of the decoding procedure, several methods use various mechanisms.

2.3 Literature review on Existing Models

To produce weight maps that blend the original picture with multiple smoothed replicas of the input image, the author (Dutta; 2021) recommends employing a depth-estimation network. The background blurring in the produced photographs differs from the ground truth image, despite the fact that the proposed network provides a perceptually acceptable bokeh appearance. To generate alternate representations of smoothed images, disk blur kernels can be employed instead of gaussian blur kernels. A depth estimation network predicts the matching weight maps for the synthesized bokeh picture, which is created as a weighted sum of the input image plus a number of differentially smoothed pictures. The biggest blur kernel utilized in this article, on the other hand, determines the suggested network's maximum blur effect. Using more and larger blur kernels, we can get a powerful blur effect.

One alternative to using Neural Networks is to create a user-friendly interface based on semi-automatic object segmentation and 3D modeling algorithms that can conduct accurate depth labeling interactively. Furthermore, to achieve the narrow focus look, it uses a real-time gather-based DOF rendering approach that has been modified to mimic the bokeh effect. The depth of field may be changed and the depth of the focal plane assigned in real-time using the built-in user interface and the updated DOF rendering process to create a shallow focus image with bokeh effect. (Hu et al.; 2013) To achieve real-time efficiency, a gather-based DOF rendering methodology was used, which overcomes the bokeh simulation limitation of the traditional gather-based method. They got the findings without using any Neural Networks, therefore it may be claimed that this approach is inefficient, because it may fail if we send photos with difficult lighting circumstances.

There are many papers which have used the state of the art Deep Learniing Algorithms One of the model is (Luo, Peng, Xian, Wu and Cao; 2020). Examine how a single all-in-focus image may realistically portray bokeh. To produce bokeh effects, exist-

ing computational bokeh rendering techniques include a basic flat backdrop blur. They calculated the depth of an image using VGG16. Unet16 is then utilized to calculate the defocus. The Resnet50 network is used to speed up the inference process. As a result, the rendering effects differ from the real-life bokeh seen on DSLR cameras. They propose utilizing a multi-stage network to learn shallow depth-of-field from a single bokeh-free image to address this problem.

The network is made up of four modules: defocus estimation (Zhuo and Sim; 2011), radiance, rendering, and upsampling. The four modules are designed to teach students how to learn global and local information around the edges of in-focus objects of various sizes. Experiments have shown that this method may provide an unique bokeh effect in difficult conditions. This multi-stage network was trained to provide outputs ranging from multi-channel probabilistic defocus maps to single-channel defocus maps by integrating defocus estimations into the network and using just bokeh photos as supervision. As a result, the aesthetic efficiency of synthetic bokeh is increased. The model also includes a radiance module that converts image intensity into scene radiance, allowing them to use physics to generate bokeh. Using the aforementioned methodology, the quality of bokeh photographs may be significantly enhanced. With this architecture, a PSNR of 23.58 is achieved. The Imagenet data set is used to train and test the model. This system has a bad performance due to its low PSNR value. To validate the modules and demonstrate their effect on the output of the proposed network, extensive visualizations and ablation studies are provided. In the future, they'll look at employing dynamic filters for kernel selection and modifying the defocus estimation model to learn single-channel defocus maps without qualifying parameters, which will reduce training time.

(Purohit et al.; 2019) To distinguish between the task of drawing between large leading and regional areas, as well as their relative depth, the model is augmented by pre-trained protrusion segments and depth evaluation modules. A pyramid link module decodes back a densely linked encoder in the proposed network. The PSNR value is 24.74, and the SSIM is 0.89. This network is equipped with previously trained highlighted region segmentation and depth estimation modules since this function needs the separation of large foreground and background areas, as well as their relative depth. A dynamic filter (DDDF) synthesis module that immediately integrates the desired impact into the input picture boosts the network's power significantly. The network placed second in the AIM 2019 bokeh effects competition, demonstrating its ability to create realistic bokeh effects.

Introducing a new 3D reconstruction method (Synthetic Aperture) that generates depth maps from brief video sequences captured with conventional cameras without the need of multi-lens lenses, active sensors, or the photographer's purposeful movements. In this work, the authors (Wadhwa et al.; 2018) explore the features of unwanted motion and present a technique for retrieving 3D knowledge from image sequences. It's the first practical method for rebuilding a three-dimensional structure from a series of short video samples. Even in the presence of deviations, the depth of the random point is shown to be a good initialization of the cost function of changing beams for modest changes in relation to the reference view and comparable camera locations. Because the image consistency computation is noisy, it considers utilizing the distance relationship between pixels to regulate the depth graph, and the resultant depth graph appears considerably better than using simply the connection between adjacent pixels. Because the image consistency calculation is noisy, it considers using the distance relationship between pixels to control the depth graph, and the resulting depth graph looks much better than using just the link between nearby pixels.

While stereo algorithms have received a lot of attention, they're rarely thought of in terms of creating defocused images. This method outperforms prior stereo algorithms in terms of defocus while being quicker than comparable solutions. The author (Barron et al.; 2015) proposed producing a disparity map from a stereo pair as a method for synthetically defocusing a picture. Despite its simplicity, the approach generates defocused pictures that appear better than comparable state-of-the-art technologies. The method relies on a one-of-a-kind mechanism for integrating optimization concerns within standard splat-blur-slice data structures, which are frequently employed for simple bilateral filtering.

When it comes to editing photos and movies, an image mat is a must-have. Traditional picture carpet cutting techniques requiring interaction, such as trims and strokes, cannot be used in real time on a mobile phone. Finally, mobile phones are working on an autonomous portrait animation system based on a fast deep carpet that can accomplish real-time darkening at a rate of 15 frames per second and requires no input. (Zhu et al.; 2017) In this post, the author suggests a quick and easy way to obtain carpet photos and movies. According to this unique perspective, cleaning a raw segment mask can be faster and more accurate than standard non-imaging approaches. They compared the components of the proposed system to Deeplab and PSPN, two of the most popular semantic segmentation networks. The Light Dense Network (LDN) improves performance substantially, while the Feathering Block (FB) decreases gradient error (Grad. Error) and mean squared error significantly in the suggested method (MSE).

3 Synthetic Defocus Methodology Approach and Design Specification

3.1 Synthetic Defocus Methodology Approach

Knowledge discovery in databases (KDD framework) is a common project implementation technique that also helps us give a systematic approach to project planning. This process helps in extracting the knowledge from the data set. Figure 1 shows the Synthetic Defocus Methodology Approach. It is based on the application of specific principles, with the first stage being the selection of a data set or subset of data from which knowledge must be extracted. The second step is to pre-process the data and make it suitable to perform the necessary operations, the third step is to remove noise or clean the data which is also known as Exploratory Data Analysis, and the fourth step is Data Transformation, which helps join and transform the data for data mining algorithms. This step is also known as reduction because only the variables that are relevant to the goal or task are selected. The pattern evaluation or interpretation phase, which helps utilize business intelligence tools to extract information and may be used to construct a presentation to acquire insights and patterns from the data set, is the next principle step, which is an essential element of the KDD architecture. The KDD model was employed in this research because it is highly efficient and works well with image-based data.

3.1.1 Project Understanding and Data Gathering

Understanding the project is required before taking the appropriate measures. The first step is to acquire information. We utilized the EBB data collection, which is publicly

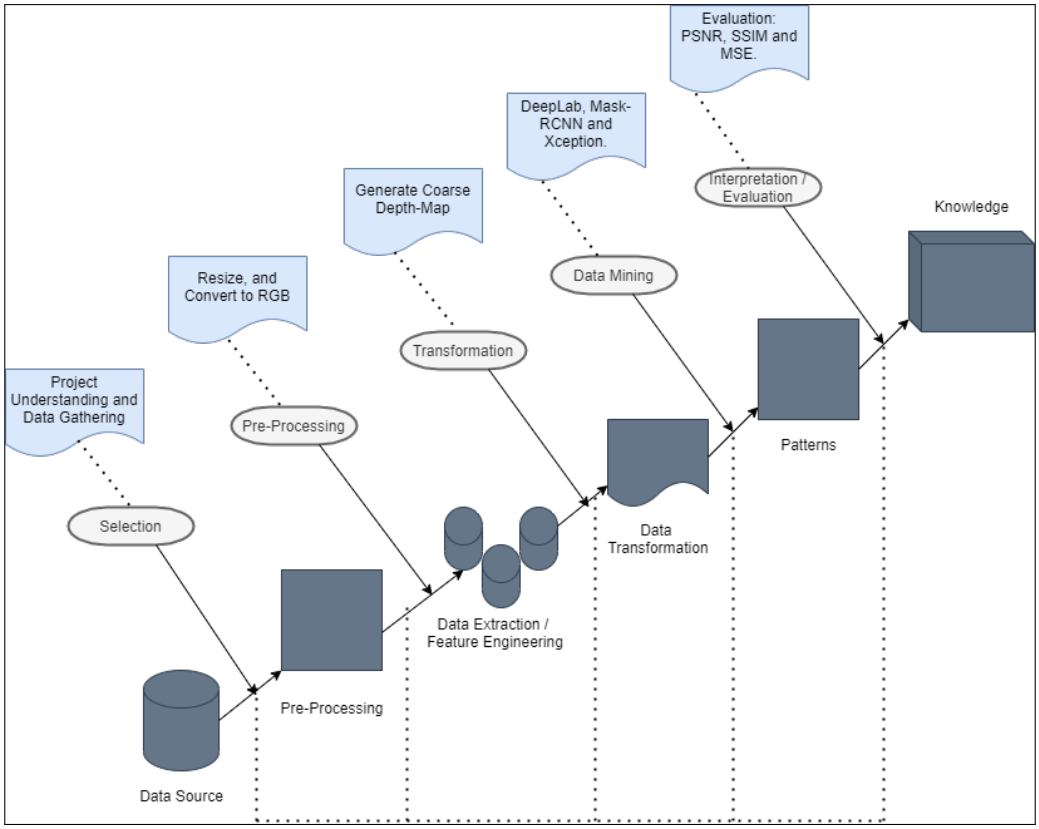

Figure 1: Synthetic Defocus Methodology Approach

available on the ETH Zurich website, for our research. More than 10000 pictures were captured using a high-quality DSLR camera in this data set (Canon 7D).
[1]

Figure 2 represents the sample data set. The images were taken by adjusting the camera's aperture size. Two photographs of the same scenario were taken to demonstrate the difference between wide and narrow Depth of Field. A wide aperture (f/1.8) produces a dramatic Bokeh Effect, but a small aperture (f/16) produces a sharp shot. The photographs were taken during the day in a variety of locations, with varying lighting and weather circumstances. The photographs were taken in automated mode, with the default parameters used throughout the collecting process.

3.1.2 Data Pre-Processing

When it comes to deep neural networks, data preparation is essential. Publicly available data isn't always in good enough shape to utilize straight away. As a result, image down-sampling is required before sending images to the convolutional layers. In this situation, the data we had was practically ready to utilize, but a few minor tweaks were required. The photos were first converted to RGB format and then scaled to the required dimensions. Furthermore, data down-sampling is largely dependent on the model we will employ in the future. Each model's input size might differ. As a result, different sizes and shapes of input pictures were employed for three models.

[1] `http://people.ee.ethz.ch/~ihnatova/pynet-bokeh.html#dataset`

Figure 2: Sample Data

Because the collected picture pairs were not perfectly aligned, they were matched first using SIFT key-points and the RANSAC technique. The resultant pictures were then cropped to the intersection portion and down-scaled to a final height of 1024 pixels. 200 picture pairings were set aside for testing from the resultant 10,000 photos, while the remaining 4.8 thousand photo pairs can be utilized for training and validation. This collection contains pictures at a resolution of roughly 1024×1536 pixels. Finally, we used MegaDepth's Megadepth model to generate a coarse depth map for each wide depth-of-field image. These maps may be layered directly with the input pictures and used by the trained model as extra direction.

3.1.3 Setting up the Virtual Environment / Data Transformation

To successfully run this project, several packages, libraries and frameworks are used. First of all, Anaconda is used to create the virtual python environment. Later on, specific to the model the packages, libraries and frameworks like TensorFlow, PyTorch and CUDA are installed. The specific versions of all the necessary packages are explained in detail in the Configuration Manual.

3.1.4 Data Mining

Data Mining / Modelling techniques are used to train the model and estimate values while adapting to business needs, then verifying and testing them. EBB data set (Everything is better with Bokeh) is used by several researchers to generate Depth of Field (Bokeh)

images correctly. Many algorithms were tried and tested on this data set. In this paper, DeepLab, Mask-RCNN and Xception these models were implemented. Out of those, DeepLab outperformed other models by a slight margin.

3.1.5 Evaluation Metrics

Evaluation metrics are used to determine how well a model performs in relation to specific parameters and to compare it to other models. Three assessment measures were utilized in this project: PSNR (Peak Signal to Noise Ratio), SSIM (Structural Similarity Index), and MSE (Mean Signal to Noise Ratio) (Mean Squared Error).

The PSNR block calculates the peak signal-to-noise ratio between two pictures in decibels. This ratio is used to compare the quality of the original and compressed images. The better the quality of the compressed or reconstructed image, the higher the PSNR. Moreover, there is no correct value for MSE. The closer the value to zero, the better the model is. SSIM is the Structural Similarity Index of an image. The SSIM index that results is a decimal number between 0 and 1, with 1 being the only value that can be reached in the event of two identical sets of data, indicating complete structural similarity. There is no structural resemblance if the value is 0. The results got after the implementation is further discussed in the Evaluation section.

4 Design Specification

The Two-Tier Architecture that this project followed is seen in Figure 3. The Client Layer and the Business Layer, often known as the Data Layer, are the two levels that make up the system. Clients might be photographers or regular individuals who will serve as the model's image source. First, the data is acquired, and then the pictures are processed for feature extraction. The modified data is then supplied to each of the three models separately for model-specific pre-processing. DeepLab, Mask-RCNN, and Xception are the three models utilized here. DeepLab functioned admirably under all conditions.Three assessment criteria are taken into consideration once the model has been trained using the EBB data set for testing and validation purposes. The three measures utilized to evaluate and compare three models on the given data set are PSNR (Peak Signal to Noise Ratio), SSIM (Structural Similarity Index), and MSE (Mean Squared Error).

5 Implementation of Synthetically Defocused Models

5.1 Introduction

Implementing the generation of synthetic defocus (Semantic Segmentation aka Bokeh) from the small aperture images of the three models are discussed in this section. With respective models, several architectures are used based on the existing architectures and networks.

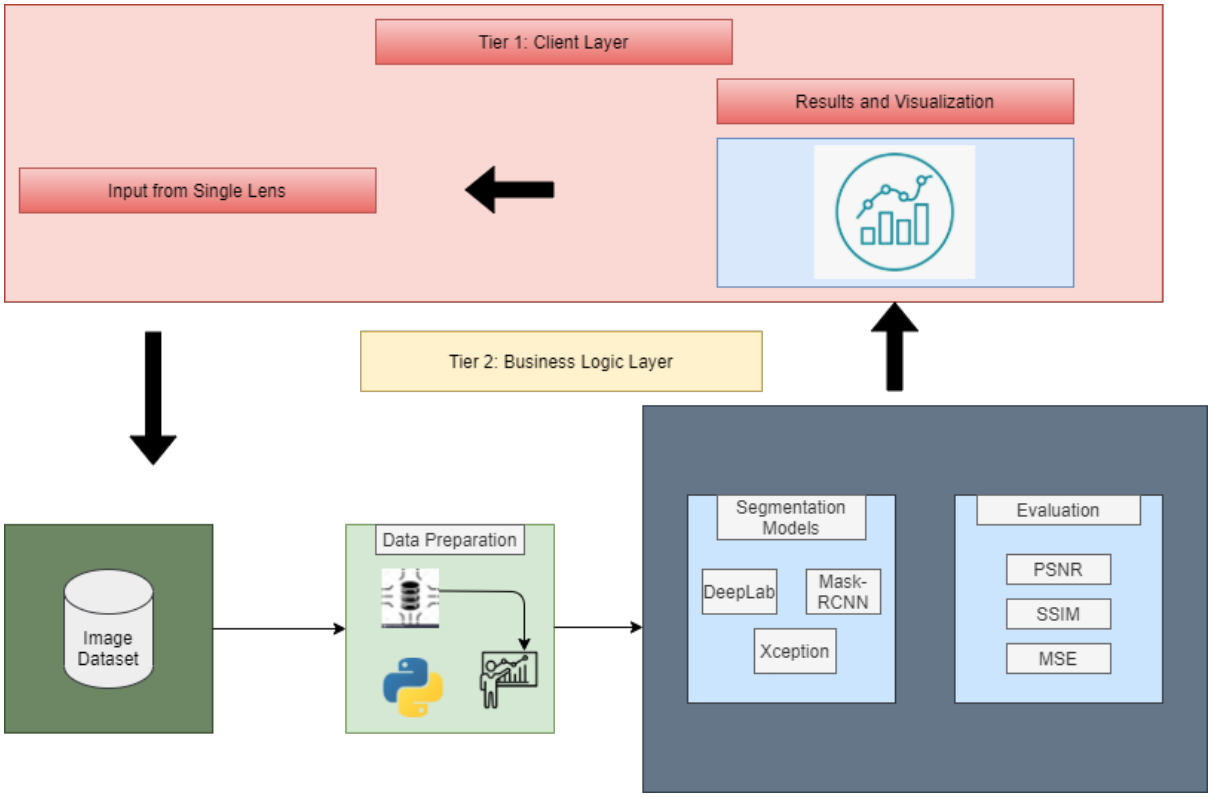

Figure 3: Process Flow

5.2 Implementation of DeepLab

DeepLab is a semantic segmentation architecture which is shown in Figure 4, used to separate foreground from the background of the image. Basically it uses a Convolutional Neural Network with few advances which generates semantic maps for the each input image. DepLab uses Atrous Covolution, an improved ASSP module which enables batch normalization and provides image level features. Moreover, it solves the problem of CRF (Convolution Random Field). This model uses VGG as its backbone network. The backbone refers to the network which takes as input the image and extracts the feature map upon which the rest of the network is based. Output of the backbone network is the input to the DeepLab network.

An encoder is made up of tightly linked modules that effectively handle the problem of vanishing gradients and enhance feature propagation while decreasing model complexity significantly. Furthermore, we convert information from pixel to channel space, which improves the receptive field for all of the encoder's blocks. This also has the additional benefit of decreasing the computational cost.

5.2.1 Atrous Convolution

The very first block in the encoder is the Atrous Convolution. Convolution is used to extract relevant characteristics from an input stream. The formula to calculate Atrous Convolution is shown in the Figure 5. Convolution may be achieved using a variety of different filters in image processing. Each filter type aids in the extraction of various elements or features from the input picture, such as horizontal, vertical, and diagonal

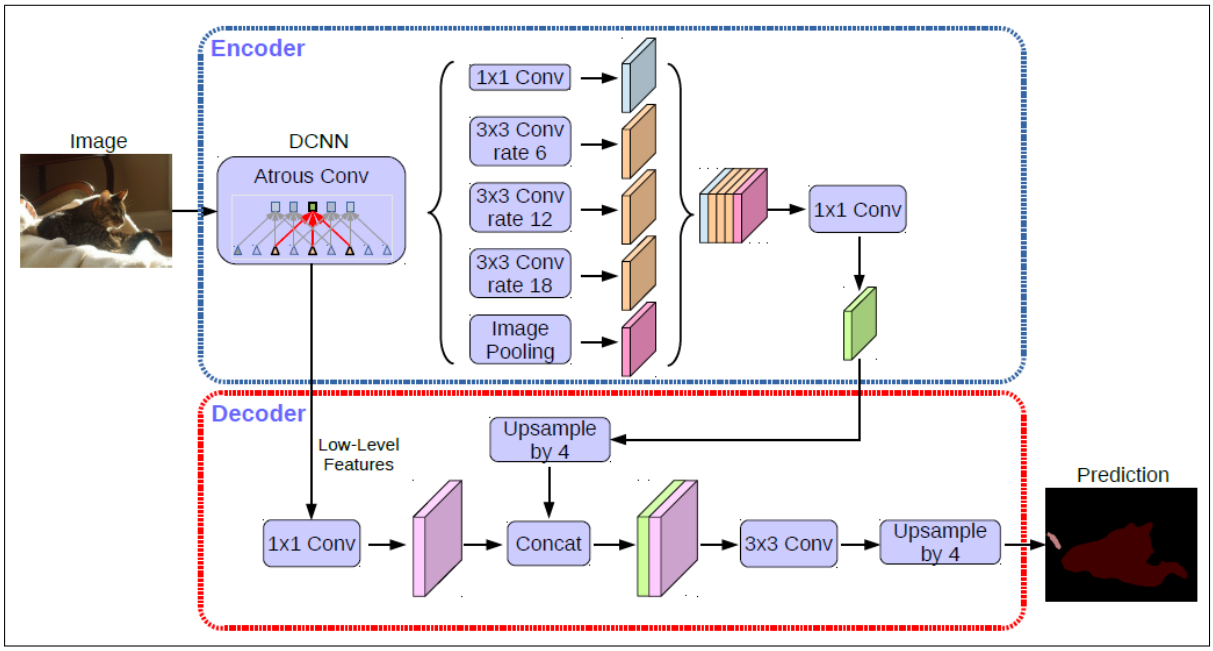

Figure 4: DeepLab Architecture

edges. Similarly, various characteristics are retrieved by convolution utilizing filters whose weights are automatically acquired during training in a Convolutional Neural Network. After then, all of the collected characteristics are combined to generate recommendations. Convolution has a few advantages, including weight sharing and translation invariance. Convolution also considers the spatial connection between pixels. These might be particularly useful in a variety of computer vision applications, as these frequently require detecting objects in which some components have certain spatial relationships with other components.

$$(F *_l k)(\boldsymbol{p}) = \sum_{\boldsymbol{s}+l\boldsymbol{t}=\boldsymbol{p}} F(\boldsymbol{s})k(\boldsymbol{t})$$

Figure 5: Atrous Convolution calculation

Atrous Convoutions (Dilated) are first introduced by this paper. Dilated convolutions, on the surface, appear to inflate the kernel by introducing gaps between the kernel pieces. The l (dilation rate) extra argument specifies how much we want to expand the kernel. Figure 6 accurately depicts the Atrous Convolution with varying l. Although implementations vary, l-1 spaces are generally added between kernel components. The kernel size for l = 1, 2, and 4 is shown in the diagram below. This model makes use of Atrous convolution which increases the dilation rate (l) gradually resulting in the exponential growth of the effective receptive field. Moreover, this convolution systematically increases the multi-scale contexual information without loosing the information. Hence, by using Atrous Convolution, we can see a significant increase in the accuracy of the model rather than the state of the art VGG (Backbone Network) Network.

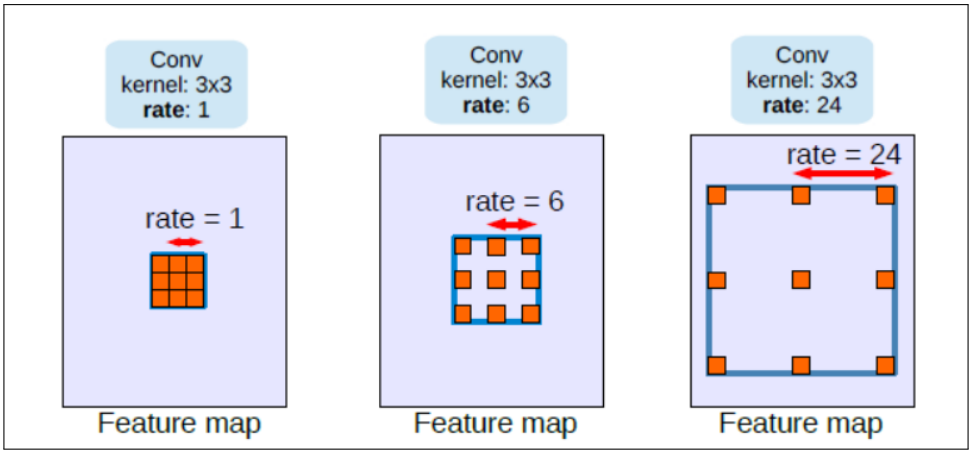

Figure 6: Atrous Convolutional with different rates of l

5.2.2 Atrous Special Pyramid Pooling (ASSP)

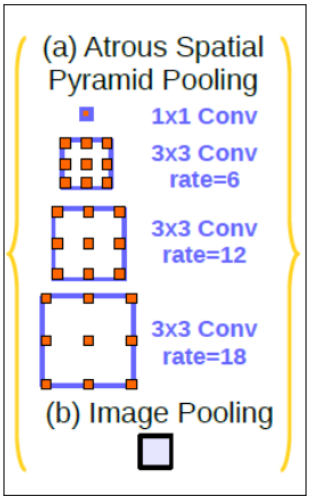

Figure 7: Atrous Special Pyramid Pooling

The use of ASPP is justified since it has been discovered that as the sample rate increases, the number of valid filter weights decreases. When the output stride is 16, the pyramid consists of one 11 convolution and three 33 convolutions with rates of (6, 12, 18). It is shown in the Figure 7. It features 256 filters in total, as well as Batch Normalization. The features from all the branches are then concatenated and sent through another 11 convolution before being generated by the final 11 convolution.

These enhancements make it easier to extract dense feature maps for long-range contexts. This enhances performance on segmentation tasks by increasing the receptive field exponentially without diminishing or losing the spatial dimension. This model was implemented with the help of TensorFlow. While Training, L1 Construction loss and Adam Optimizer were used. Initial learning rate was 0.0001. The code is written in Python and uses standard TensorFlow packages. OpenCV was used to pre-process the data. Results and Evaluation metrics of this model will be analysed in the Evaluation section.

5.3 Implementation of Mask-RCNN

Mask R-CNN is a Faster R-CNN extension in essence. Because it is quicker, R-CNN is widely used for object detection tasks. For a given image, it returns the class label and bounding box coordinates for each item in the image. The Mask R-CNN architecture is built on the base of faster R-CNN. For each item in a given picture, Mask R-CNN will provide the object mask as well as the class label and bounding box coordinates. Faster R-CNN begins by utilizing a ConvNet to extract feature maps from the images. After running these feature maps through a Region Proposal Network, the probable bounding boxes are returned (RPN).

Then, on these candidate bounding boxes, we apply a RoI pooling layer to make all of the candidates the same size. Finally, the ideas are sent to a fully connected layer, which classifies and outputs object bounding boxes.

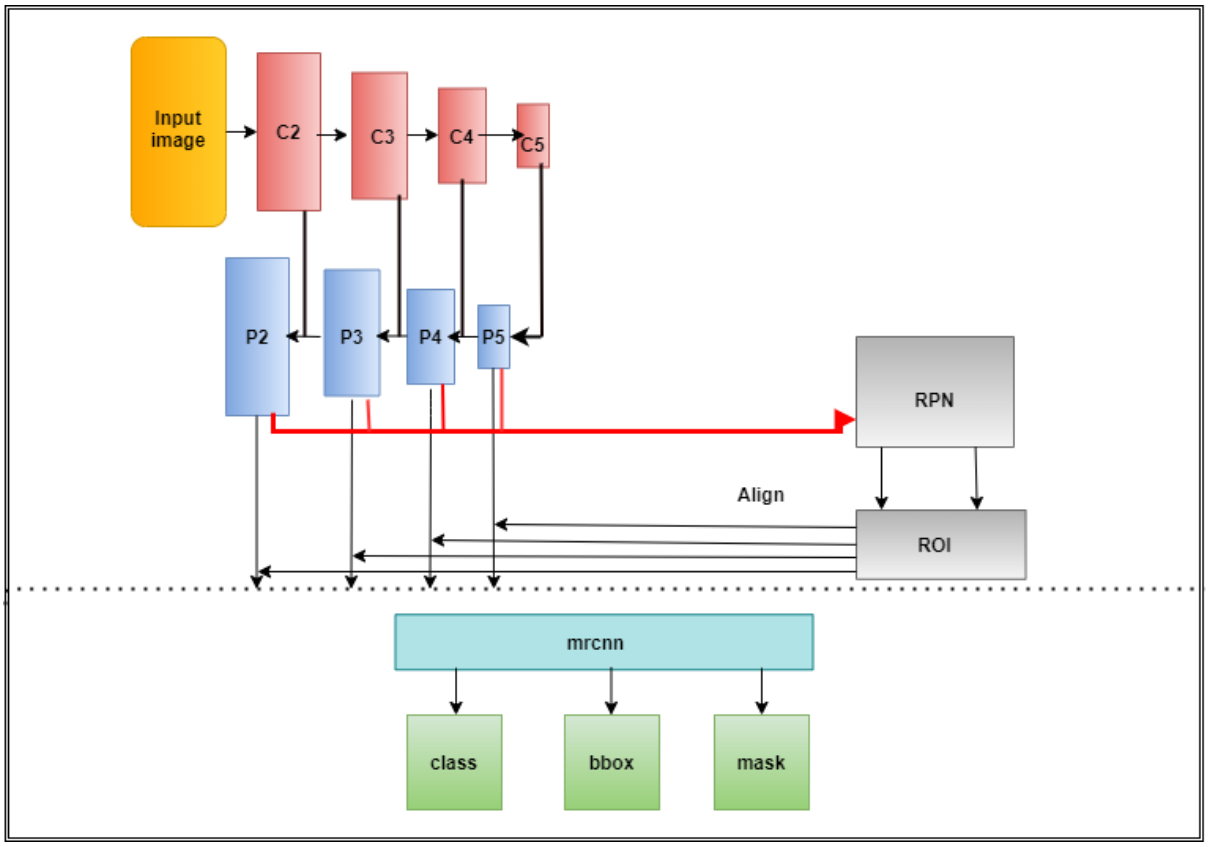

Figure 8: Mask-RCNN Architecture

Mask-RCNN Architecture is shown in the Figure 8. To extract features from the pictures in Mask R-CNN, we employed the ResNet architecture (Backbone Network). So, the first step is to use the ResNet architecture to extract features from a picture. These characteristics serve as an input to the following layer. We now apply a region proposal network to the feature maps produced in the previous phase (RPM). This essentially forecasts whether or not an object is present in that area (or not). We receive those areas or feature maps that the model predicts will include some item in this stage. Because the areas acquired by the RPN may have different forms, we use a pooling layer to transform all of them to the same shape. The class label and bounding boxes are then predicted by passing these areas through a fully connected network. The procedures are nearly

identical to how Faster R-CNN works up to this point. The difference between the two frameworks now comes into play. Furthermore, Mask R-CNN creates the segmentation mask. To do so, we first calculate the region of interest in order to shorten the calculation time. We used the ground truth boxes to calculate the Intersection over Union (IoU) for all of the projected areas. IoU can be calculated as,

IoU = Area of Intersection / Area of Union

If the value of IoU is greater than or equal to 0.5 then we consider that area as region of interest, otherwise that region is neglected. For all the regions, same procedure is applied and at the end those regions with IoU value greater than 0.5 are selected. Once IoU for two bounding boxes are calculated, we move forward to Segmentation Mask. Once the IoU is calculated, segmentation masks for each object in that region are generated. It returns a mask of size 28 X 28 for each region which is then scaled up for inference.

Moreover, segmentation maps are calculated using OpenCV package. A segmentation map is a planar division. Each section in the picture symbolizes an object or a specific location.In addition to that, used Gaussian Blur algorithm with the help of OpenCV here to blur the image. Gaussian blurring, median blurring, averaging blurring, and bilateral filtering are the four major approaches used to produce blurring effects using OpenCV. The application of convolutional processes on the picture using a filter is common to all four approaches (kernel). Between the four blurring techniques, the values of the applied filters change.

This model has already be trained million times on the COCO 2017 data set. Hence, the training phase can be omitted. We are using PyTorch framework for Mask-RCNN implmentation, it's quite easy to set up and use. The results and evaluation metrics are further discussed and compared in the Evaluation section.

5.4 Implementation for Xception

Francois Chollet proposes the Xception Model. Xception is an expansion of the Inception Architecture that uses depthwise Separable Convolutions to replace the conventional Inception modules. The structure of Xception is based on ResNet, however instead of the convolutional layer, Depthwise Separable Convolution is used. When a non-linear activation function is not employed, the convergence process is faster and the accuracy is greater between the 3x3 convolution for learning spatial correlation and the 1x1 convolution for learning inter-channel correlation. When compared to VGG-16 and ResNet in terms of ImageNet classification performance, Xception outperformed them both.

Entire flow of the Xception Architecture is depicted in Figure 9. Two convolutional layer blocks are followed by a ReLU activation in the entering flow. The number of filters, filter size (kernel size), and strides are all detailed in the diagram. Separable convolutional layers are also available. Max Pooling layers are also available. When there are more than one step, the strides are indicated as well. There are also Skip connections, in which the two tensors are combined using 'ADD'. In each flow, it also indicates the form of the input tensor. For example, we start with a 299x299x3 image and end up with a 19x19x728 image following the entrance flow. Similarly, this picture depicts the image size, various layers, number of filters, filter shape, kind of pooling, number of repetitions, and the possibility of adding a fully linked layer at the end for the Middle and Exit flows. Batch normalization is also applied to all Convolutional and Separable Convolutional layers.

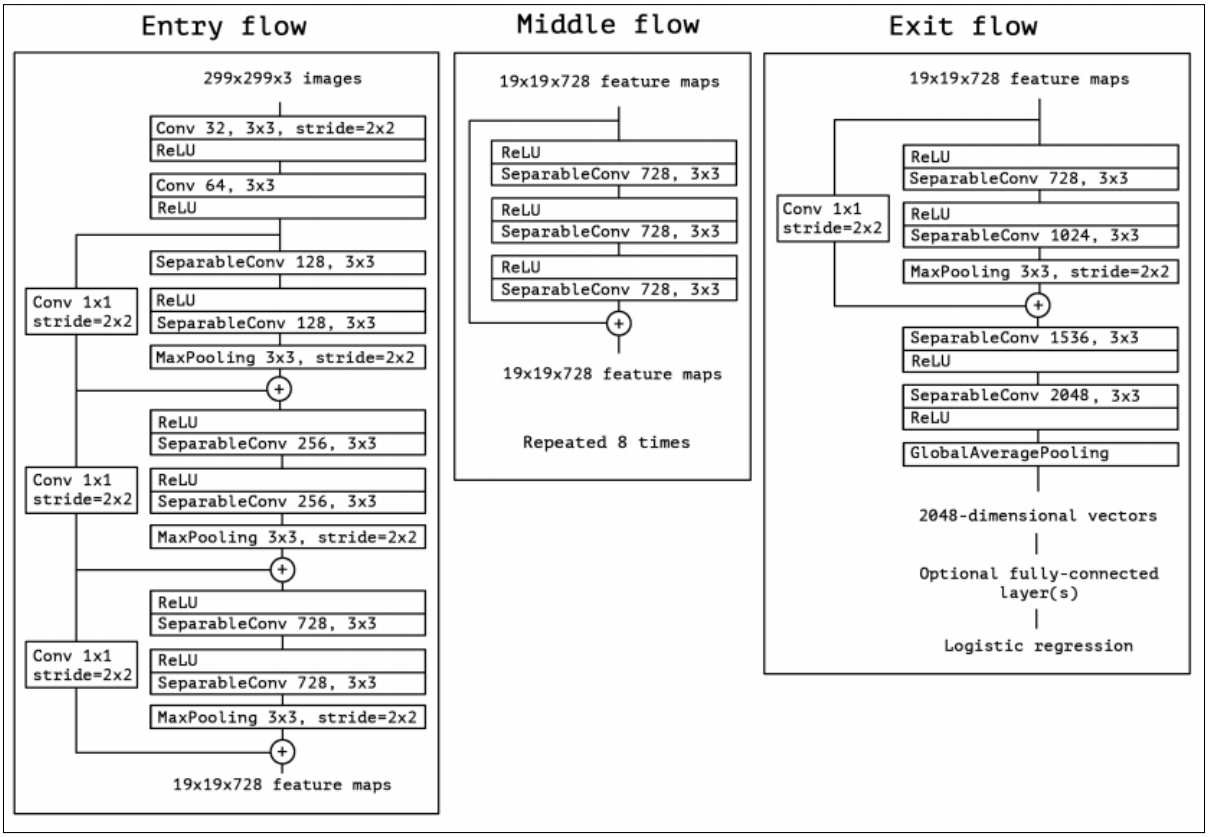

Figure 9: Xception Architecture

Depthwise separable convolutions operate with kernels that can't be factored into two smaller kernels, unlike spatial separable convolutions. A depthwise separable convolution, like the spatial separable convolution, divides a kernel into two distinct kernels that perform two convolutions: depthwise and pointwise. To convert a 12x12x3 picture into an 8x8x3 image, a depthwise separable convolution will need three kernels. We have three 5x5x1 kernels that move 8x8 times in depthwise convolution. There are 4,800 multiplications in 3x5x5x8x8. We have 256 1x1x3 kernels that move 8x8 times in the pointwise convolution. That's 49,152 multiplications of 256x1x1x1x3x8x8. The network can handle more data in less time by performing fewer calculations.

This model was implemented using TensorFlow and VGG pre-trained Model as a backbone netwrok. Matplotlib and Numpy packages are used to do the pre-proessing. Gaussian Blur is a very popular methodology by which whole image can be blurred. Gaussian Blur was used to blur the respected image. In image processing, Otsu's approach is an adaptive thresholding method for binarization. It can determine the best threshold value for the supplied image by examining all threshold values (from 0 to 255). According to all the research have been carried out so far on the Google's Xception model, it is supposed to work well than others. Surprisingly, DeepLab model with VGG backbone network performed slightly better than others.

6 Evaluation and Results of Developed Models

6.1 Experiment 1: Evaluation of DeepLab

Figure 10: DeepLab Results

On the EBB data set, the model was tested and verified, and the results are average PSNR, SSIM, and MSE scores obtained on 10 sample pictures. According to the sample test results, the average PSNR score is 33.86. The better the reconstructed bokeh pictures are, the higher the PSNR score. The PSNR score is calculated by enlarging and downscaling the original picture. The PSNR score is calculated by comparing the input picture with the synthetically defocused output image. The PSNR score indicates that the model generates bokeh pictures with precision edge detection, however the MSE number is not that low. The MSE value averages about 100. In comparison to previous techniques, the SSIM value is approximately 0.75, which is decent.

Results from the DeepLab Model are shown in the Figure 10. The photos on the left side of the diagram are input images, while the ones on the right side are Bokeh images. DeepLab clearly performs well in the semantic segmentation challenge. Edge detection

in the DeepLab model, on the other hand, is a little more exact in the image above. The borders of the object's shirt and coat were a bit faded towards the ending of the first model. With accurate edge detection, DeepLab was able to correct it.

6.2 Experiment 2: Evaluation of Mask-RCNN

Figure 11: Segmentation Map

The Figure 11 shows the segmentation map for a test image. There is only one class in this image, hence the object which is localized will be represented by 255 pixel value. The more the objects in an image more the classes will be detected in an image.

In the Figure 12, we can see that the wooden bar is blurred out in the first photograph. It did a good job of locating all of the males on board. The object localisation and bokeh effect are good in the other two models, however edge detection appears to be smoothing around the edges of a picture.

When compared to the DeepLab model, the Mask-RCNN model performed roughly similar. The average PSNR score was 33.35, which is higher than the rest of the models. The average MSE was about 89, with an average SSIM score of 0.78.

6.3 Experiment 3: Evaluation of Xception

The Xception model has performed admirably, as demonstrated in the Figure 13. In each image, we can observe that powerful edge detection has been applied. Because the depth-of-field is significantly narrower, crisp pictures are produced. A MSE of 153.12 is observed with an average PSNR score of 26.87. The SSIM number should be considerably closer to 1, but instead we obtain a 0.60.

A segmentation Map for an image is shown in the Figure 14. It has identified in total 2 classes in an image which are 'man' and 'boat'. After the segmentation map is generated, gaussian blur is used and generated the final Bokeh image.

Overall, DeepLab model with VGG backbone network and Mask-RCNN model performed exceptionally well in all three metrics.

Figure 12: Mask-RCNN Results

6.4 Discussion and Comparison of models with the existing models

The performance of the three models presented in this study in comparison to current models is given in Table 1. The EBB data collection was used to train the majority of the current models. As a result, this comparison is correct.

When comparing the three models to the current models, we can observe that they all have excellent PSNR ratings. These are the average scores of ten validation photos, not a single image. Among the current models, PyNET and DDDF are the two models that have been implemented and have good PSNR ratings. DeepLab and Mask-RCNN, on the other hand, outperform current models, demonstrating that they can be implemented and utilized to create superior Depth-of-Field pictures.

All of the pictures are shown together in Figure 15. The output of each model is displayed after the original pictures. It is obvious that the DeepLab model, performs

Figure 13: Xception Results

better than the others visually. In comparison to the DeepLab Model, the Mask-RCNN model comes close. However, the Xception model still need further refinement.

As seen in Table 1, it is evident to say that DeepLab and Mask-RCNN performed well and we got a PSNR score of 33.86 and 0.75 for SSIM. The above mentioned objectives have been implemented successfully and models are generating the desired outcomes. This project has given me immense knowledge about how CNN's work with different frameworks like TensorFlow and PyTorch.

Figure 14: Segmentation Map

Table 1: Evaluation Metrics

Model / Author	**PSNR**	**SSIM**
DeepLab	**33.86**	0.75
Mask-RCNN	**33.35**	0.78
Xception	**26.87**	0.60
DDDF	24.74	**0.89**
Xiong et al.	23.93	0.89
Airia-bokeh	23.58	0.87
PyNET	23.28	0.87
Dutta et al.	22.14	0.87

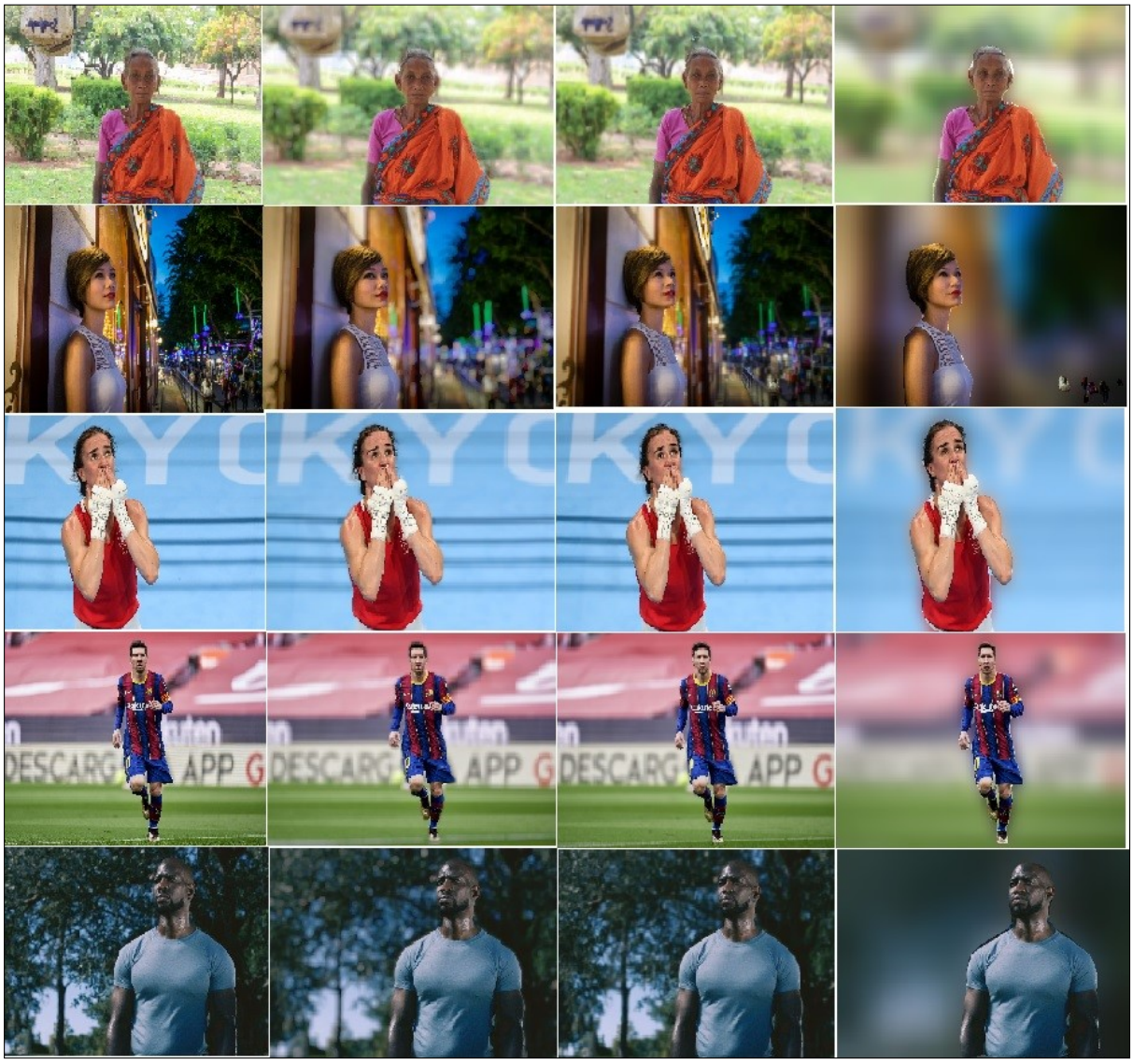

Figure 15: Comparison of Exiting photos with Each model output

7 Conclusion and Future Work

The primary goal of this research was to study and research about the Synthetic Defocus Generation using Deep Learning. The research question, sub-research question and all the objectives were met successfully. All of the models performed well but DeepLab performed better amongst all. The research also helped to determine how the Deep Learning Algorithms helped to match the efficiency and consistency of the images captured by the DSLR. In the completion of Research Objectives 2, a customized KDD approach was used to implement the research. To determine whether model works better, a comparitive analysis of both models have been conducted. Mask-RCNN and DeepLab are the two models which performed exceptionally well, hence it is safe to say that these two algorithms are deployable and ready to utilise.

In future, these exact models can be implemented in a high performance computing machine where we can pass huge data sets to train the model and then more accurate results can be expected.

8 Acknowledgement

Dr. Catherine Mulwa's assistance and supervision made it feasible to carry out the study endeavor. She was invaluable in helping me finish my assignment and motivating me to do so. I'd also want to express my thanks to my family members for their financial and moral support during my Master's program.

References

Adams, A., Talvala, E.-V., Park, S. H., Jacobs, D. E., Ajdin, B., Gelfand, N., Dolson, J., Vaquero, D., Baek, J., Tico, M. et al. (2010). The frankencamera: an experimental platform for computational photography, *ACM SIGGRAPH 2010 papers*, pp. 1–12.

Barron, J. T., Adams, A., Shih, Y. and Hernández, C. (2015). Fast bilateral-space stereo for synthetic defocus, *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 4466–4474.

Cristofalo, E. and Wang, Z. (2017). Out-of-focus: Learning depth from image bokeh for robotic perception, *arXiv preprint arXiv:1705.01152* .

Dutta, S. (2021). Depth-aware blending of smoothed images for bokeh effect generation, *Journal of Visual Communication and Image Representation* **77**: 103089.

Gidaris, S. and Komodakis, N. (2015). Object detection via a multi-region and semantic segmentation-aware cnn model, *Proceedings of the IEEE international conference on computer vision*, pp. 1134–1142.

Hu, M.-C., Tseng, Y.-N., Wu, J.-L., Kuo, C.-H., Hsiao, Y.-M. and Hua, K.-L. (2013). Real-time depth of field rendering with bokeh effect, *2013 IEEE International Symposium on Consumer Electronics (ISCE)*, IEEE, pp. 99–100.

Luo, C., Li, Y., Lin, K., Chen, G., Lee, S.-J., Choi, J., Yoo, Y. F. and Polley, M. O. (2020). Wavelet synthesis net for disparity estimation to synthesize dslr calibre bokeh

effect on smartphones, *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 2407–2415.

Luo, X., Peng, J., Xian, K., Wu, Z. and Cao, Z. (2020). Bokeh rendering from defocus estimation, *European Conference on Computer Vision*, Springer, pp. 245–261.

Papageorgiou, C. P., Oren, M. and Poggio, T. (1998). A general framework for object detection, *Sixth International Conference on Computer Vision (IEEE Cat. No. 98CH36271)*, IEEE, pp. 555–562.

Purohit, K., Suin, M., Kandula, P. and Ambasamudram, R. (2019). Depth-guided dense dynamic filtering network for bokeh effect rendering, *2019 IEEE/CVF International Conference on Computer Vision Workshop (ICCVW)*, IEEE, pp. 3417–3426.

Qian, M., Qiao, C., Lin, J., Guo, Z., Li, C., Leng, C. and Cheng, J. (2020). Bggan: Bokeh-glass generative adversarial network for rendering realistic bokeh, *European Conference on Computer Vision*, Springer, pp. 229–244.

Raskar, R. and Tumblin, J. (2005). Computational photography, *ACM SIGGRAPH 2005 Courses*, pp. 1–es.

Ren, S., He, K., Girshick, R. and Sun, J. (2016). Faster r-cnn: towards real-time object detection with region proposal networks, *IEEE transactions on pattern analysis and machine intelligence* **39**(6): 1137–1149.

Singh, N., Kumar, M., Mahesh, P. and Sarkar, R. (2018). Depth aware portrait segmentation using dual focus images, *2018 IEEE International Conference on Multimedia and Expo (ICME)*, IEEE, pp. 1–6.

Sun, B., Lin, W., Sha, H. and Su, J. (2021). Gnetseg: Semantic segmentation model optimized on a 224mw cnn accelerator chip at the speed of 318fps, *arXiv preprint arXiv:2101.10444* .

Wadhwa, N., Garg, R., Jacobs, D. E., Feldman, B. E., Kanazawa, N., Carroll, R., Movshovitz-Attias, Y., Barron, J. T., Pritch, Y. and Levoy, M. (2018). Synthetic depth-of-field with a single-camera mobile phone, *ACM Transactions on Graphics (ToG)* **37**(4): 1–13.

Wang, R. (2016). Edge detection using convolutional neural network, *International Symposium on Neural Networks*, Springer, pp. 12–20.

Zhang, H., Dana, K., Shi, J., Zhang, Z., Wang, X., Tyagi, A. and Agrawal, A. (2018). Context encoding for semantic segmentation, *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 7151–7160.

Zhu, B., Chen, Y., Wang, J., Liu, S., Zhang, B. and Tang, M. (2017). Fast deep matting for portrait animation on mobile phone, *Proceedings of the 25th ACM international conference on Multimedia*, pp. 297–305.

Zhuo, S. and Sim, T. (2011). Defocus map estimation from a single image, *Pattern Recognition* **44**(9): 1852–1858.