Segmentation using CNN for Display Distinction of Dialysis Machines

Pavan Chowdary Cherukuri[1], Kishan Reddy Raghunath[1]

***Abstract*— The aim of this project is to distinguish the dialysis machines which has a screen for a robot to interact with. We aim to use segmentation techniques to identify dialysis machines by using deep neural networks. We would be implementing the transfer learning using PyTorch on an already trained U-Net model that segments display devices and then use its learning to segment dialysis machines and its screen. RCNN and RRCNN based on U-Net models can be implemented for training deep architecture and better feature representation but for the available task currently, we believe the U-Net model is good for segmentation [1]. A pipeline was designed to segment the dialysis machine screens, initially object detection was performed on the input image and the pixels detected were cropped and sent to the segmentation model. Yolov5 object detection model has been trained on a set of custom made dataset to detect the dialysis machines. Later, a U-Net model has been trained initially on COCO dataset that specifically consists screens (TV's, monitors) so that the parameters of this trained model can be used to train a second model of a custom created dataset of dialysis machines (here the outputs of Yolo). The loss values of the trained models have been plotted accordingly with the number of epochs, and these results were observed to be in optimal range. To compare the models, different architectures, ResNet50 for object detection and Transformer based ResUNET architecture for segmentation were also trained. The Yolov5 trained model is observed to detect dialysis machines with 0.9116 mean average precision and 0.813 recall. The UNET model was observed to have average dice coefficient of 0.899, precision of 0.8573 and precision of 0.9023.**

I. Introduction

The COVID-19 pandemic, in 2020, caused hospitals to be overloaded with patients. In this situation, immunocompromised patients were at high risk of contagion. An example of this was dialysis patients, who faced various inconveniences when they had to attend medical facilities. So we propose a mobile robot equipped with a manipulator to interact with the dialysis machine and can be operated remotely so that risks of contagion can be minimized by reducing contact with the health care workers. For this reason, the segmentation of dialysis machines and segmenting of the screen on the dialysis machine need to be performed. Thresholding and histogram-based methods are the classical methods by which segmentation can be performed. Histogram methods are very efficient techniques where a histogram is computed from all the pixels in the image, but it is difficult to identify significant peaks and valleys in the image. Thresholding converts a gray-scale image to a binary image and a threshold value is selected to categorize into k groups or clusters (for K-means) applying Euclidean to calculate the least distances on the other hand choosing k manually, scaling with a number of dimensions, and clustering of outliers is a downside of this technique.

Other methods of segmentation include Partial differential equation-based methods, Variational methods, Trainable segmentation, etc. We are implementing transfer learning on a pre-trained U-net model. Unlike most deep learning algorithms that rely on a mass of data to obtain good results, U-Net is a unique network model for image segmentation. Due to the implementation of transfer learning for a custom-created dataset of dialysis machines, it requires very less images for post-transfer learning network training to get high performance compared to the large and tedious amounts of the initial learning [2]. Labeling the dataset requires an expert in this field which is expensive and requires a lot of effort and time. Sometimes, different data transformation or augmentation techniques are applied for increasing the number of labeled samples available. In the case of semantic segmentation, the image backgrounds are assigned a label and the foreground regions are assigned a target class [3]. It will be easier to highlight the dialysis machine and its screen as two different classes over the background of the diagnostic room scene.

II. Related Work

Our present work is the extension of work by Hassam Khan Wazir and Kapila [4], in an attempt to interact with Dialysis machines remotely, here, the robot is mounted on a fixed platform and the camera visualizes the screen and fiducial markers are used to get the position of the screen.

Over the years, a lot of image segmentation of TV displays and monitors has been performed, for various reasons, be it in the production of monitors, OCR systems, etc [5, 6]. As mentioned in the previously, segmentation can be done by various methods, but problems do exist such as not being able to locate the screen automatically, unable to rectify the perspective projection, not able to work on a non-backlight screen with uneven illumination and not able to adapt to changes of screen type.

Efforts have been made to optimize the segmentation by implementing FCNN techniques as such made by Arbelaez et al. [5]. The segmentation of phone screens is being done by Otsu thresholding techniques previously by Liu Meiju [7], the results of these works show an improved algorithm with optimized inspection accuracy, improved segmentation efficiency, and reduced running time Arbelaez et al. [5], Liu Meiju [7].

The implementation of a Recurrent Convolutional Neural Network (RCNN) and Recurrent Residual Convolutional Neural Network (RRCNN) based on U-Net models has

[1]New York University, Brooklyn, NY 11201, USA {pc3088, krr9721}@nyu.edu

improved training deep architecture and made feature accumulation in medical image segmentation better. These techniques are used in cases of low data sets such as that of internal organs, blood vessel segmentation, etc in the medical field [1]. Since the data set for the segmentation of dialysis machines can be created and limited, the use of such recurrent models on U-Net makes the system redundant and time-consuming to run the models.

III. Method

Deep learning has allowed the field of computer vision to develop rapidly, hence the use of deep learning architectures has been taken into consideration for this purpose. Since this is a unique kind of problem to solve and the segmentation of the dialysis machine screens is too specific, transfer learning is going to be used. The initial idea was to perform image segmentation to segment the input image into two classes: one class with a dialysis machine, and another class with a screen inside of the dialysis machine. This approach had several flaws and is complex. First, segmenting a class within a class is quite an uphill task and secondly, while performing segmentation of screens, there might be situations where the wrong screen is segmented. Since the base model is trained for generalized TVs and monitors, there might be a risk of segmenting other screens instead of only dialysis machine screens. For this, we have come up with a pipeline by training various models. So, the idea is to perform object detection initially for the given input image. Then the object detection algorithm detects if any dialysis machines are present in the given input image. Then the detected pixels would be cropped from the image and then sent to the segmentation pipeline. The segmentation model segments just the screens. The models and the architectures used will be discussed in the following sections.

A. *Object detection*

For the object detection part, after pre-processing of the data, the dataset was trained on an Object detection model with model architecture Yolo (You only look once) v5. Yolo uses a single neural network to simultaneously predict multiple bounding boxes and class probabilities for those boxes. The architecture is based on a modified version of the Darknet architecture, which uses residual connections and a global average pooling layer to improve performance. The model is trained on a large dataset of annotated images and can be fine-tuned for specific tasks. Overall, YOLO v5 is a powerful tool for object detection and can be applied on specific tasks, such as dialysis machines in our case. The single-stage Yolo V5 architecture is divided into three parts: backbone, neck and head. These layers will be a combination of different convoluted neural networks, and has different functions like the backbone extracts features, neck generates pyramids and the model head performs detection. Yolo v5 has leaky ReLu and sigmoid functions as the activation functions and stochastic gradient descent optimizer. Yolo returns three outputs of the detected objects, the detected classes, their bounding boxes and their objectness score. This

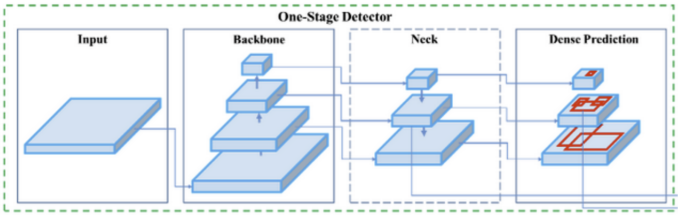

Fig. 1. Single stage detection Architecture 11 [8]

model uses binary cross entropy (BCE) loss to compute the class loss and objectness loss, and it uses complete intersection over union loss to compute the location or box loss. In this situation there won't be a class loss as the model can't predict wrong classes since there is only one class dialysis machines. A general single stage architecture (YOLO) has been visualized in the figure 1.

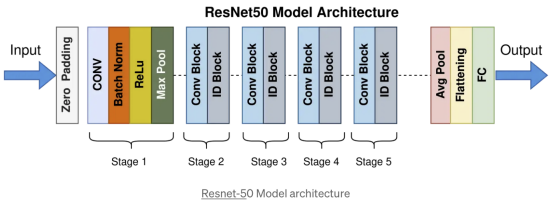

Fig. 2. Single stage detection Architecture 13 [9]

In addition to Yolo, we initially have trained the model on ResNet50 architecture with pretrained weights. ResNet50 is also one of the widely used object detectors, hence this architecture was initially tested out. 50 layer deep architeture of Resnet50 with multiple convoluted blocks can be seen from figure 2. Mean square error loss was used for this model. The results for this model were not as satisfactory as Yolo.

B. *Pre-training a UNet model*

Transfer learning can be defined as the improvement of learning in a new task through the transfer of knowledge from a related task that has already been learned [10]. To achieve this, a model needs to be pre-trained on a large dataset (should generally be more than the specific dataset that is being required , so that the weights and the other parameters can be initialized. **UNET** architecture has been used for this purpose. UNET uses a Fully Convolutional Network model, and it is already an established architecture in medical imaging for segmentation. It is also spreading its wings widely in almost all the computer vision fields, UNET provides better performance for Segmentation tasks particularly with very few training examples. The U-shaped architecture formed by a contracting path and an expansive path can be seen in the figure below. The UNET consists a total of 23 convolutional layers.

Initially, the dataset collection is performed. COCO dataset has been used for this purpose and this data is pre-processed

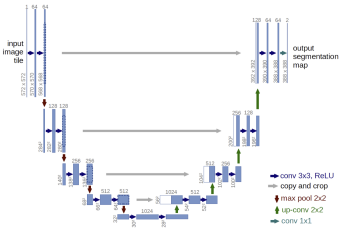

Fig. 3. UNET Architecture [2]

and sent to the model [11]. As seen in the UNET architecture, the model has several convolutional layers, the respective layers were created accordingly. Next, the loss functions were created and the schedulers and optimizers were initialized. Finally the model is trained with a specific number of epochs, batch size and learning rate.

C. *Transfer Learning for a new data set*

Reusing a previously learned model for a different issue is known as transfer learning. Transfer learning is a technique where a machine uses its understanding of one activity to help it generalize about another. Transfer learning allows for quick training development, it eliminates the need to start from scratch and can obtain amazing results with a minimal training dataset. The model that was trained on the screens and monitors of the COCO dataset will be transferred here, whose weights will be used as initial weights for the model with dialysis machines dataset. The dialysis machines dataset that will be used for transfer learning model will be the one given as output by the object detection model. All the parameters and the methods will be discussed more in detail in the Experiments section.

D. *Different architecture for segmentation - TransResUNET*

As per the feedback received, we have also tried the whole process with a different architecture, Transformer based ResU-Net. This is an encoder-decoder based architecture, as per [12] it is believed that this architecture should give superior results compared to the other architectures like UNET. The architecture of TransResUNET can be seen from figure [4]

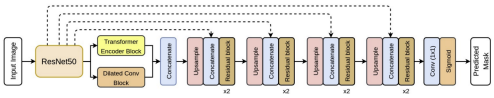

Fig. 4. UNET Architecture [12]

IV. Experiments

A. *Data Preprocessing for Object Detection*

Scraping for Data creation: It is the process of extracting data from websites or other sources. To collect the images of dialysis machines, a python Script has been written that uses the Google Chrome web driver to auto-download the images from the Google images website. The code downloads the images from the source location so that original high-quality images can be obtained instead of the low-pixelated thumbnails. Images collected are in two divisions, one just the dialysis machines in the image frame and the other having dialysis machines in the hospital environment which included patients, beds, and recliners along with the machine in the image frame as a dataset.

Cleaning the Data: Once the images are downloaded, although the right keywords are used to search, junk images are included and need to be eliminated manually. In this process, the scraping procedure has been carried out by searching multiple keywords for multiple iterations to get the required amount of dataset. A total of 321 images were collected for the object detection process. After this procedure, all the images are resized to 256x256 resolution so that all the images are of uniform size and the training of the model is quicker.

Annotating images for object detection: The collected 321 images which contained dialysis machines need to be labeled for training purposes. This process included identifying the dialysis machines in the images and marking their region with a rectangular outline (bounding box) and naming the object. To perform this process LabelImg has been used. LabelImg is an open-source graphical image annotation tool that is used to label object bounding boxes in images. While Annotating the images, if images contained multiple dialysis machines all such regions in the frame are annotated. These annotations are saved in ".txt" for all the individual images in the format supported by Yolo. LabelImg has a set of 15 default classes and the dialysis machine was added as an additional class to the separate ".txt" file classes file. The ".txt" file contained the class of the object and details about the position of the bounding box i.e., x, y, width, and height. x and y correspond to the coordinates that are calculated relative to the center grid cell of the image. These files and images serve as input to the training of the Yolo model. After creation of the dataset, to augment the dataset and feed it to the object detection model, the images were augmented and split into, train, validation and test data by using 'RoboFlow' API. The images and labels were augmented to 900, with varied brightness, exposure, random rotations of -45 to +45 degrees and horizontal flips.

B. *Object detection results*

The Yolo model was trained for a total of 100 epochs with a batch size of 16. The model was trained using google collab's GPU. It took 12 minute 17 seconds to train on images after data augmentation. The loss vs epochs graph can be seen below.

As seen from the graphs, the train box loss value decreases to 0.02874, train objectness loss decreases to 0.017, and validation box loss value decreases to 0.031 and validation objectness loss decreases to 0.028. Taking a look at the metrics, the major metrics the model, the final precision of the model is 0.9158, final recall is 0.813, mean average precision is 0.9116. The by seeing the metrics and losses, it can be deduced that the model performs greatly, and there is

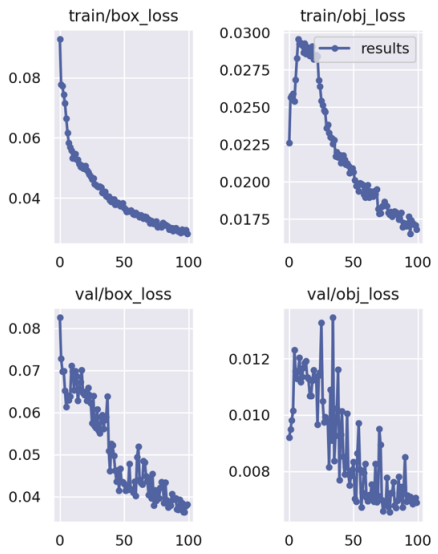

Fig. 5. Train and Validation box, objectness loss

no case of over fitting too. The output of the object detection model for a test inference is displayed in Figure 7.

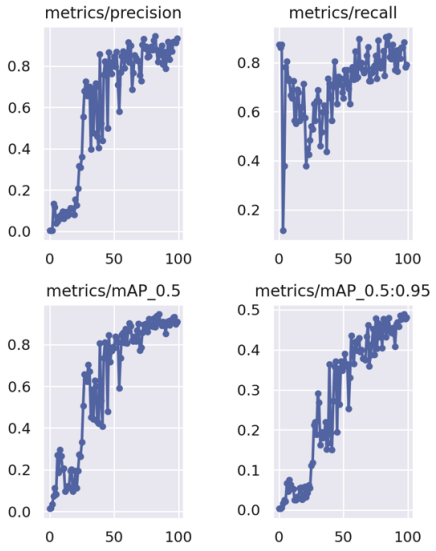

Fig. 6. Train and Validation box, objectness loss

The train and validation loss plot versus the number of epochs for Resnet50 can also be seen in Figure 8. It can be seen that the loss values became stagnant after few number of epochs, the bounding box loss didn't decrease for this model, as expected. Change in hyperparameters would have resulted in a better model, but the Yolo architecture already gave the required results, hence more effort was not made in tuning this model.

C. *Data preprocessing for segmentation - Base model*

The COCO dataset, published by microsoft has been used for this purpose. There are more than 120k images in this Dataset, out of which around 4800 images (particularly

Fig. 7. Yolo output image visualization

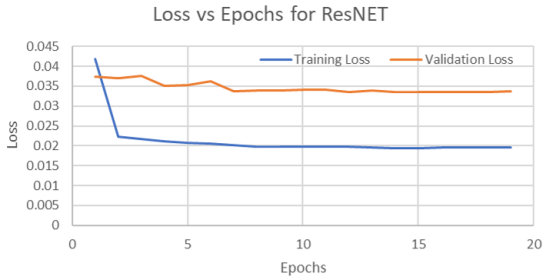

Fig. 8. Resnet50 loss plots

screens) have been used for training the model. These image consists of two datasets- 'train dataset' and 'validation dataset' (which helps to evaluate how well the model makes predictions on new data). These images are initially converted to the size 256*256 pixels for faster computation. Each dataset also has ground truth in the form of annotated images, but these have been converted to ground truth masks (specifically binary masks). This data is therefore, converted to tensors, reshaped as required and is given as an input to the model. The input and output images can be seen in Figures 4 and 5.

Fig. 9. Input image

D. *Segmentation using UNET architecture - Base model*

Training parameters and device specifications for the base model: The model has been trained for **'25'** epochs, with a batch size of **'32'** and an initial learning rate of **'1e-4'**. NYU HPC with single core and 16GB RAM has been used

Fig. 10. Ouput mask image

Fig. 12. Loss values for new configuration

for training this model, it can also be trained using Google Collab as well. Deep learning tensor library **Pytorch** and supporting libraries like **PIL** have been used.

Optimizer and scheduler: An optimizer must be used to modify the attributes of the model like learning rate, weights and other parameters. For this purpose adam optimizer has been used from **torch.optim** package since it has few parameters to tune and a faster computation time when compared to the other optimizers.

It is necessary to decrease the learning rate in any deep learning models as the number of epochs increases, a learning rate scheduler should be used for this purpose. **StepLR** scheduler from **torch.optim** package has been used for this purpose.

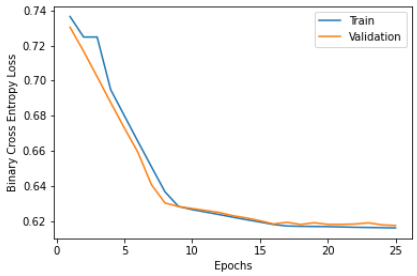

Fig. 11. Bce vs epoch plot

Results: There were two losses in the UNet architecture model, one binary cross entropy loss and dice loss (1-dice coefficient), previously we have plotted bce loss vs epochs as seen in Figure 11 and assumed the model was well trained. But later we have figured out that by changing learning rate and scheduler parameters, the model can be made even better. Now, the learning rate was initialized to 1e-5 and step LR scheduler was used to reduce the learning rate every 5 epochs. This model tends to be even better than the previous one. The total minimum loss (bce loss+dice loss) for the model with learning rate 1e-4 was 0.736846 for the validation data. We will take a look at the total loss values for this configuration in the Figure 12.

It can be seen that the total validation loss has decreased from 1.244 to 0.537 which indicates the decrease in loss was more, and the final loss was less than the final total loss for the previous configuration.

E. Data preprocessing for segmentation - Transfer learning

Data Collection: The outputs of the Yolo detection algorithm serves as the input to the segmentation procedure. The detected images with the bounding boxes are cropped using OpenCv using the x, y, width, and height of the bounding boxes created on the detected images similar to that of the annotation performed for Yolo dataset preparation. The images in this phase are only of dialysis machines cropped to the bounding box size and resized to 256x256 resolution to maintain the uniformity of all the images in the dataset. Due to wrong detection and bounding boxes are falsely created and these images need to be removed manually. The dataset created at this point contains images of only Dialysis machines.

Annotation to Binary: The creation of a dataset for Segmentation requires binary images as mask images, this is created using VGG Image Annotator (VIA), here all the prepared images are loaded and since the screen is rectangular in shape, rectangles are drawn marking the position of the screen on the dialysis machines of all 342 images. The output of this process is a JSON file containing information about the location of the drawn rectangles. The JSON file has to be processed to get the mask images in black and white, white being the region of interest and the rest of the image is marked black in color to perform this, python code has been utilized this reads all the images and the data related to it from the JSON file and new images and masks are created in png format. These images serve as input to the segmentation training algorithm. The resulting images and masks were augmented together using the 'Albumentations' to a total of 1296 images of which 800 were considered as train dataset and remaining as validation dataset. The brightness and exposure were changed for the augmented images, random 90 degree rotations were given and also the images were flipped horizontally.

F. Segmentation using UNET architecture - Transfer learning

The final best base model was saved and is used for transfer learning. The UNet model was trained again on the new preprocessed dataset and the results were plotted. Hyperparameters were varied and the results for two learning rated "1e-5" and "1e-6" were plotted in Figure 13 and 14.

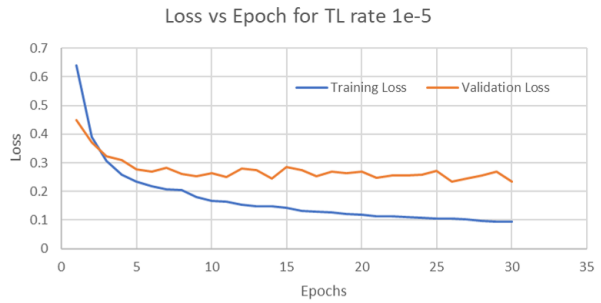

Fig. 13. Loss vs epoch plot for transfer learning

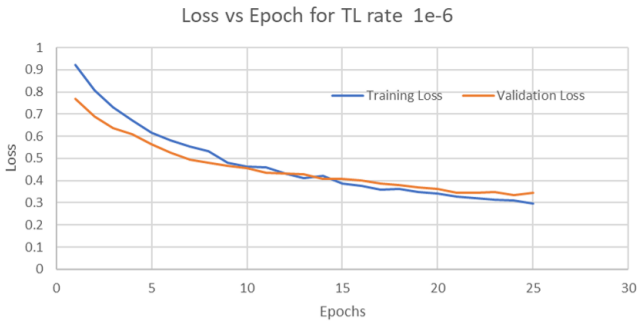

Fig. 14. Loss vs epoch plot for transfer learning

It can be seen from the figure 13 that the validation loss started at 0.45 with a learning rate of 1e-5 and ended up at 0.24. The point to be noted from this was since the learning rates for the base model and the transfer learning model were same, the validation loss almost picked up where it left in the previous model. It is to be further noted that due to the transfer learning we have achieved a decrement in the loss values (validation particularly) to 0.24, this wouldn't have been the case if there was no transfer learning involved. Also, due to the increase in learning rate to 1e-6, the loss values were increased, even the final losses and the model failed to converge, hence was forced to stop at 25 epochs. The importance of tuning hyper parameters can be understood from these plots. Metrics: Coming to the metrics, the final average dice coefficient for the test dataset for the model with learning rate '1e-5' which is our best model is, average Dice coefficient: 0.899 Recall: 0.8573 - Precision: 0.9023, which is pretty good for the segmentation model.

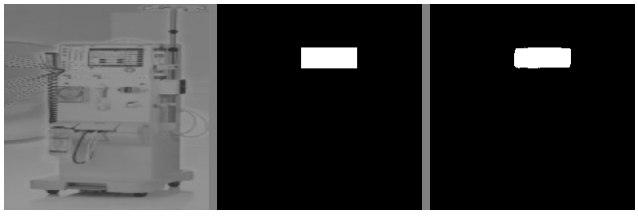

Fig. 15. Segmentation image; Ground truth; Predicted mask

G. Segmentation using TransResUNET architecture

For comparing the results of the UNET architecture, TransResUNET architecture was used. All the other parameters were same for this model as in UNET including the dataset, scheduler, optimizer and the best learning rate which was observed as 1e-5 for the UNET architecture was used here. This model also took almost the same time to get trained as UNET, but from observing the loss values from Figure 16 and 17, it can be observed that this model has less loss values. Coming to the metrics, it does slightly better with average values of test dataset of Dice coefficient: 0.9033, recall: 0.8731, and precision: 0.9293. It can be observed that this model works slightly well compared to the UNET architecture.

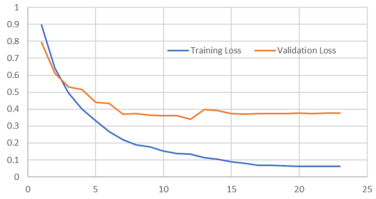

Fig. 16. Loss vs epoch plot for TransResUNET Basemodel

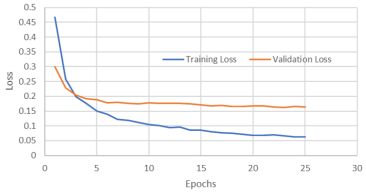

Fig. 17. Loss vs epoch plot for transferred TransResUNET model

V. Conclusions

To conclude the work done, there were multiple models trained for object detection and segmentation. The results were compared with each other and discussed. Even though the results look promising, there is still scope to improve, considering the scope and workforce available for this project, more work on tuning the hyper parameters and re-training the models, would have fetched far more wonderful results. Coming to the limitations of the above proposed pipe line, the problem with this is the image was being forced into multiple models, getting resized and cropped in the way. This would be sort of a disadvantage because when the original image will be retrieved the pixel values of the segmented masks might no be as accurate as expected. For this one mitigation is to train another neural network like an autoencoder, to retrieve image better. In future, we would want to still continue working on this project, like make the models even better. We would also want to create a specific dataset for the dialysis machines simulated in Gazebo and perform the transfer learning to the new dataset from the best obtained models.

CONTRIBUTION

Pavan: Initialized parameters, trained the U-Net model, Yolo v5, cropped images, augmented datasets, pre processed dataset for segmentation and Trans ResU-Net and tuned the hyperparameters.

Kishan: Created the New dataset, made bounding boxes, ground truth masks for segmentation , trained Resnet50, plotted the results.

REFERENCES

[1] C. Y. T. M. T. Md Zahangir Alom, Mahmudul Hasan and V. K. Asari, "Recurrent residual convolutional neural network based on u-net (r2u-net) for medical image segmentation." 1, 2

[2] P. F. Olaf Ronneberger and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," 2015. 1, 3

[3] Q. T. Z. W. Z. R. Jiawei Pan, Deyu Zeng, "Eu-net: A novel semantic segmentation architecture for surface defect detection of mobile phone screens," *IET Image Processing*, 2022. 1

[4] S. M. C. Hassam Khan Wazir, Christian Lourido and V. Kapila, "A covid-19 emergency response for remote control of a dialysis machine with mobile hri," *Frontiers in Robotics and AI*, 2021. 1

[5] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "A method to locate and recognize lcd display screen using fcn," 2019. 1

[6] A. P. Ruan Belem, Caio Cruz, "Automated video monitor screen extraction using semantic segmentation and cnn," 2020. 1

[7] G. X. Z. J. Liu Meiju, Zhuang Rui, "Application of improved otsu threshold segmentation algorithm in mobile phone screen defect detection," 2020. 1

[8] "Yolo." [Online]. Available: https://iq.opengenus.org/yolov5/ 2

[9] "Resnet." [Online]. Available: https://towardsdatascience.com/the-annotated-resnet-50-a6c536034758 2

[10] C. Y. T. M. T. Md Zahangir Alom, Mahmudul Hasan and V. K. Asari, *Transfer Learning, Handbook of Research on Machine Learning Applications*. 2

[11] "Common objects in context." [Online]. Available: https://cocodataset.org/#home 3

[12] M. B. R. M. U. B. P. D. J. P. Nikhil Kumar Tomar, Annie Shergill, "Transresu-net: Transformer based resu-net for real-time colonoscopy polyp segmentation," *eess.IV*, 2022. 3