Fetal Head Segmentation Using Ultrasound Images

Mohit Jain (mohitj2)
University of Illinois at Urbana Champaign
Champaign, Illinois
mohitj2@illinois.edu

1. Introduction

The aim of this project is to attempt to take in an ultrasound image of the womb and to segment the fetal head from the rest of the image.

The detection of the fetus head is done manually by the clinicians and doctors. But, in 3rd world countries there may be a shortage of trained clinicians who can efficiently segment the fetus from the rest of the image and produce a report on the same.

The project aims to solve this problem for the clinicians and the patients so that the exact image of the fetus can be detected which makes it easier for the clinicians to produce an extensive image for the same.
We implemented both the unsupervised learning and the supervised learning methods to detect and segment out the fetus from the rest of the image.

2. Dataset

This dataset [6] contains a set of about 1000 images of the ultrasounds of the womb. All the said images have the fetus and the surrounding organs of the mother. The average image size is about 800 by 540 pixel for all the images in the dataset.
The dataset that I will be using is sourced from Zenodo and is given at-
https://zenodo.org/record/1327317#.XmqJcS2ZPAI

3. Brief description of the solution approach

We propose a novel methodology of implementing deep learning technologies for the purpose of image segmentation and recognition from 2-D fetal ultrasound images.

Currently the only way to deal with fetal head segmentation depend on patients and clinician observations. The radiologists need to decipher the nearness of embryo head in the ultrasound image manually, which can now and again lead to misinterpretation.

The application of this project is to make an endeavor to computerize this activity. It is made to facilitate the heap on the radiologists and relieve the worry of the patient of miss estimation. We accept a 2D ultrasound image as an input and section the fetal head from it.

Our program will ask the client to choose the 2-D fetal picture and when the client chooses the picture ,our program will apply the described method on image and will resize it (224x224) for the model and in like manner will segment the fetal head out from the picture.

4. Literature survey

4.1 Pixelwise Segmentation of Uterine Convolutional Neural Networks.

[1] The initial segment of the framework is a product that is fit for looking at the pictures taken from the uterus wall, and section them to figure out which part of the photos show the fundus They have introduced a methodology of utilizing completely convolutional neural network to fragment pictures about the uterus divider. They implemented the model using FCNN models. The best suggested model was prepared through 140 epochs and were able to achieve an accuracy of about 92%. In spite of the fact that they have not investigated all the current methodologies, for example, K-intends to fragment out the Region of Interest in the image which in their case was the uterus wall. Dataset in their case was constrained as the quantity of pictures utilized were very less. They ought to have run all the greater number of epochs in order to improve the accuracy of the model. An elective method to expand the exactness is to by using various data augmentation methods like,

flipping, pivot, scaling, crop, transpose, shading transformations, and so on.

4.2 Research on Semantic Segmentation of High-resolution Remote Sensing Image Based on Full Convolutional Neural Network

This paper the dataset was a set of high- resolution remote sensing images. The authors applied Fully Convolutional Network for the purpose of semantic segmentation, the method was combined with matrix expansion technique to optimize the convolution. The paper achieves an accuracy of over 85% showing that the tuned Fully Convolutional Network combined with matrix expansion along with sufficient training can result in an efficient semantic segmentation model which can work well with a high-resolution remote sensing dataset.

4.3 Fused Convolutional Neural Network for White Blood Cell Image Classification

[3] This paper presents a fused Convolutional Neural Network to classify white blood cell images. The approached followed here is one that has a combination of 5 convolutional layers, 3 pooling layers out of which two layers are max pool layer and one layer is average pool layer and a fully connected convolutional network with single hidden layer. The model shows promising results as it has good speed for a high accuracy model predictor. But the main issue here is that the model predicts many other White blood cells as “Neutrophil Class”. Hence the model requires modifications to make it more robust to False positive and False Negatives.

4.4 Diagnosis of skin diseases using Convolutional Neural Networks

This paper presents a way to use Convolutional Neural Network for the purpose of diagnosing skin diseases. The dataset that they used has a classification of five diseases. The feature extraction was done using Convolutional Neural Network. The accuracy produced by the model is good, but the accuracy could be limited by the small size of the dataset and that there was only small amount of image for each of the five diseases to be classified.

5. Approach used

There is extremely limited work on ultrasound pictures since ultrasound pictures have a great deal of commotion which isn't easy to work with, exceptionally hard to manage yet can prompt erroneous, some of the time wrong outcomes. The subject of fetus recognition has been taken up on account of two reasons, right off the bat there is very less work in the field. Also, the present ways to deal with fetal head division depend on patients and clinician watchfulness. The radiologists need to decipher the nearness of baby head in the ultrasound picture physically, which can in some cases lead to distortion. Our implementation is made to facilitate the heap on the radiologists and alleviate the worry of the patient of misdiagnosis.

5.1 K- Means algorithm

K means is a clustering algorithm that uses iterative methods to form clusters of the data points. Here we have tried to implement image segmentation using K means algorithm.

It was applied to 12 images from the dataset with k=2, but since everything inside the ultrasound image is either tending towards black or white color, so the algorithms is not able to cluster fetus properly rather it is creating 2 cluster : one representing white tone and other representing black tone.

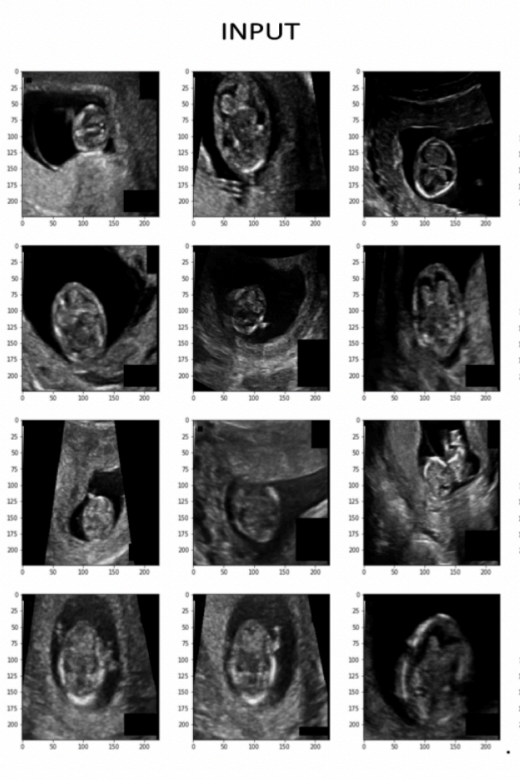


Figure 1 Input image to K- Means

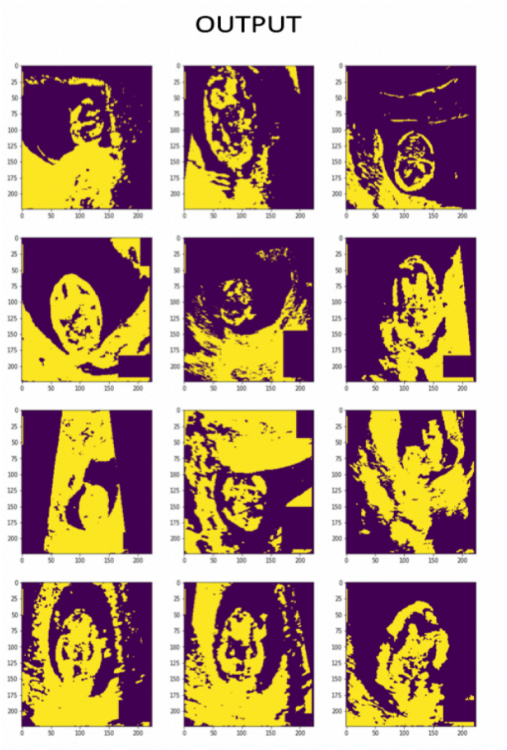


Figure 2 output from K-Means

The output of the K means Algorithm is not able to segment the image into 2 clusters one for the fetus head and the other for the rest of the image. We tried adjusting the number of clusters i.e. the value of k for the implementation but the results were still not viable. Also, the results form k>2 would not make sense as there are only 2 parts of the image that we need to get segmented. First is the image of the fetus head and the second is the rest of the leftover pixels of the image

5.2 Fuzzy C Means Algorithm

The Fuzzy C Means Algorithm is similar to K means algorithm but, the main difference here is that in case of K means one datapoint can only be a part of only one cluster whereas in the case of Fuzzy C Means Algorithm a single datapoint can be a part of more than one cluster.

We tried to implement this method considering that this may be able to segment the image as it can filter the edge pixel of the fetus as a part of both the fetus and the rest of the image.

Results from the Fuzzy C Means Algorithm is kind of similar to the results which we got from the K-means algorithm.

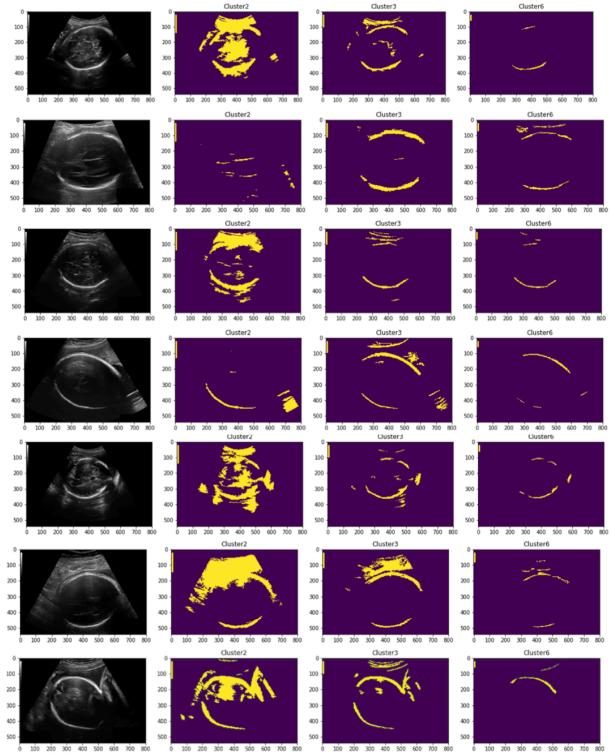


Figure 3 Results from Fuzzy C-Means

Similar to K means algorithm we tried adjusting the number of clusters. However, the results were still not viable.

5.3 Contour detection

After Applying the clustering methods, we tried to implement a contour detection to segment out the fetal head from the rest of the image.

The output from contour detector are as follows-

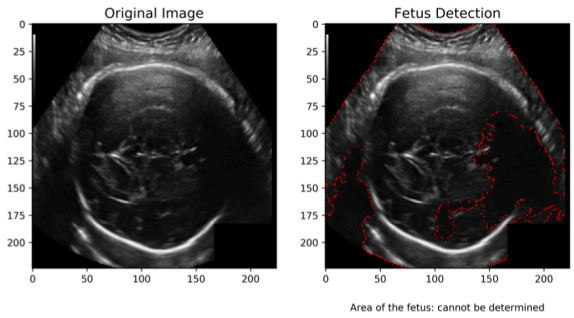


Figure 4 Results from Contour Detector

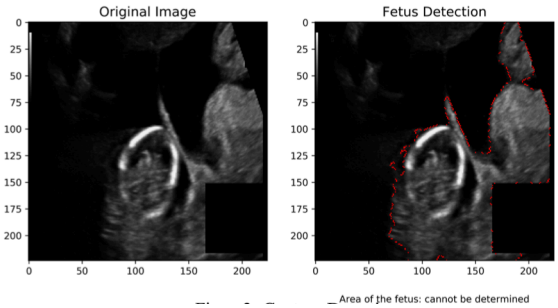


Figure 5 Results from Contour Detector

5.4 Fully Convolutional Network

After trying the unsupervised learning methods, we moved to using the supervised learning method in order to get better results.

We perform the implementation in steps: -

5.4.1. Importing all the library needed for the programs.
We have imported the following library:
NumPy: -to implement computation with arrays.
keras.model: -to load and train the model.
TensorFlow: - to import backend for Keras.
os: -for the purpose of directory explore.
cv2: -for the purpose of image processing.
VGG16: - to load and implement VGG16.
Matplotlib. pyplot: -for Plotting images.
Keras. Optimizer: - to use Adam optimizer.
Tkinter: To create the GUI.

5.4.2. Loading the dataset :

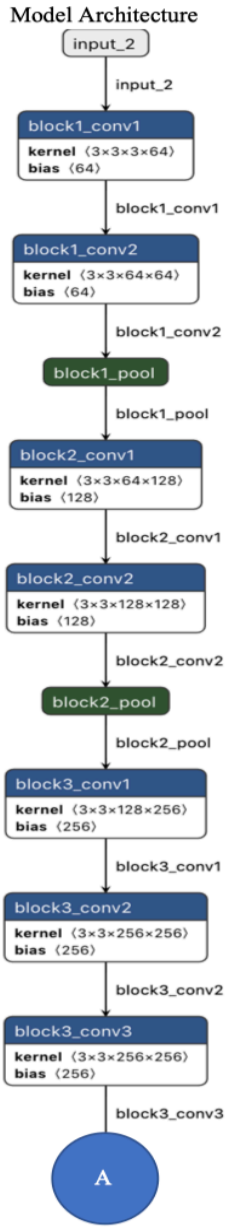
We have used google colab for implementing and training of our model, our dataset was downloaded from the link that was mentioned in the dataset section of the report. To load the dataset on colab, we created a folder containing all the ultrasound images and its annotations (marking of the fetal head) on the drive, and were fetching our dataset from the drive, every time the model was trained.

5.4.3. Image data pre-processing:

For training data files, we have used minimal image processing techniques were used this is so because Convolutional Neural Network automatically performs auto-feature Extraction.

To get started with the data processing, the first step was to resize the images. At first all the image so the dataset was of different sizes. But for the algorithm to work properly we need to have a uniform size of the image.
Hence every image was converted to 224 x 224 x 3 dimension. After the resizing was completed, we did the normalization of the data so as to avoid the overflow in the gradient calculations during the back propagation.
Then using the OpenCV methods, we created a mask where all the area under annotation part is white and the rest is marked as black. After which encoding was applied such that the white pixel of the image was marked as 0 and the black pixels were marked as 1, in an attempt to segregate the white and the black pixels of the image.

5.4.4 Model-



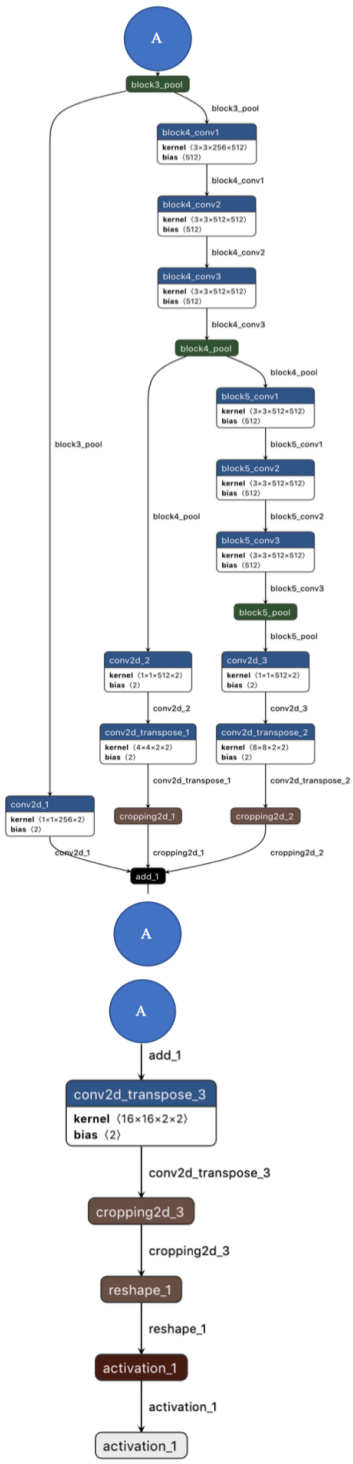


Figure 6 Model Architecture

The above graph represents the model architecture of the whole system. This was generated by using an opensource platform called Neutron [5].

The model here is the FCN-8 with some changes made in order to design a favorable model for our specific application.

We made use of the first 18 layers of VGG-16 which were divided into 5 blocks. Initially we used the pretrained weights of the ImageNet in order to save time on the learning curve.

After the five blocks, as we have resized all the image to 224x224 we used deconvolutional layers to revert the image back to 224x224. At the final layer there are 50176 number of pixels in our image. For every pixel in the image the neuron predicts the probability of the pixel being in class 1 or class 0.

5.4.5 Training the model

We experimented with the batch size of the training data and it was observed that the batch size of 5 give an optimum result for the specific application on our training model. The optimizer used here is adadelta. The loss function used is categorical cross-entropy. The main reason for using adadelta is that it gives best results in the minimum number of epochs.

We then used keras.fitgenerator() to train the model on the go as well as to call our batch generator.

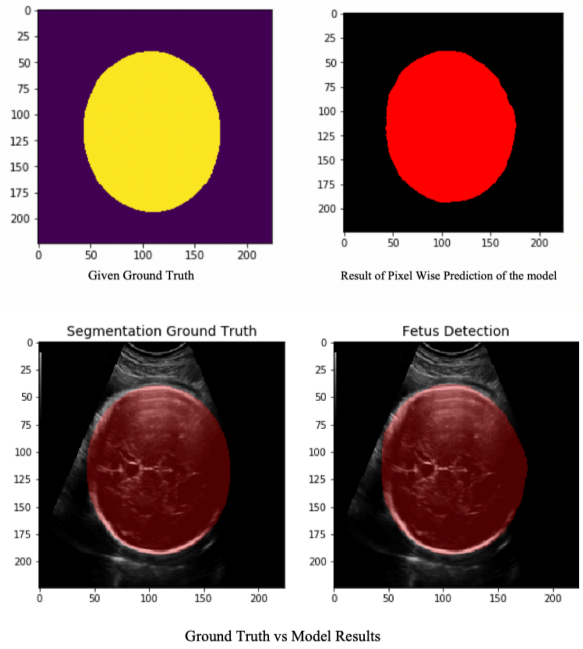


Figure 7 Ground Truth vs Model Prediction

5.4.6 Custom testing and predicting the images

The prediction method model.predict() was used which churns out 50176x2 output. Then this output from the function is processed to make a mask of red color and we resize it to 224x224x3. After the resizing is done, we plot the output superimposed with the initial image so that the

results can be viewed clearly.
The above figure shows the result of the ground truth side by side with the output generated by our model.

5.4.7 Creating a GUI based tool for Easy access of the model

After the model was ready, we created a very simple Graphic user interface to run the program using the tkinter library in Python.
The GUI has the image of the ultrasound and has two buttons one to select the image for the testing and the other to start the testing. The first button is to select the image that you want to do the fetus detection on. This will let you to the browse window of your computer. The second button is the actual test button. This was done to make the process of testing really easy for the user.

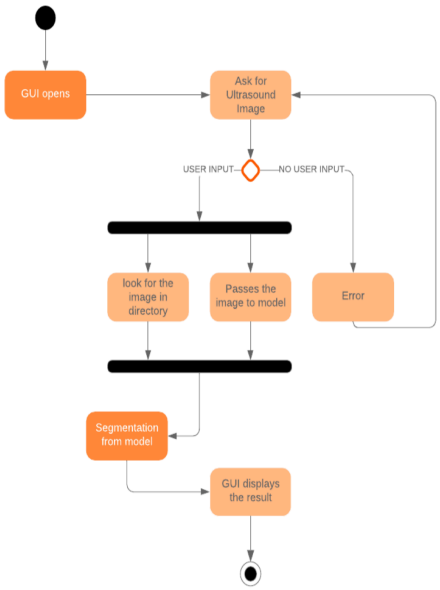


Figure 8 Block Diagram for GUI

6. Results and from Fully Convolutional Network

We have used two method in an attempt to evaluate the model-

1- Per Pixel Accuracy (PPA)
2- Jaccard Index.

By using the per pixel accuracy, which is the average of all the pixels, we achieved an accuracy of 97% for the dataset. By using the Jaccard index we were able to get an accuracy of about 0.92 where 1 is considered the optimal accuracy of Jaccard index.

The sample output of the data from the Fully Convolutional Network are as follows-

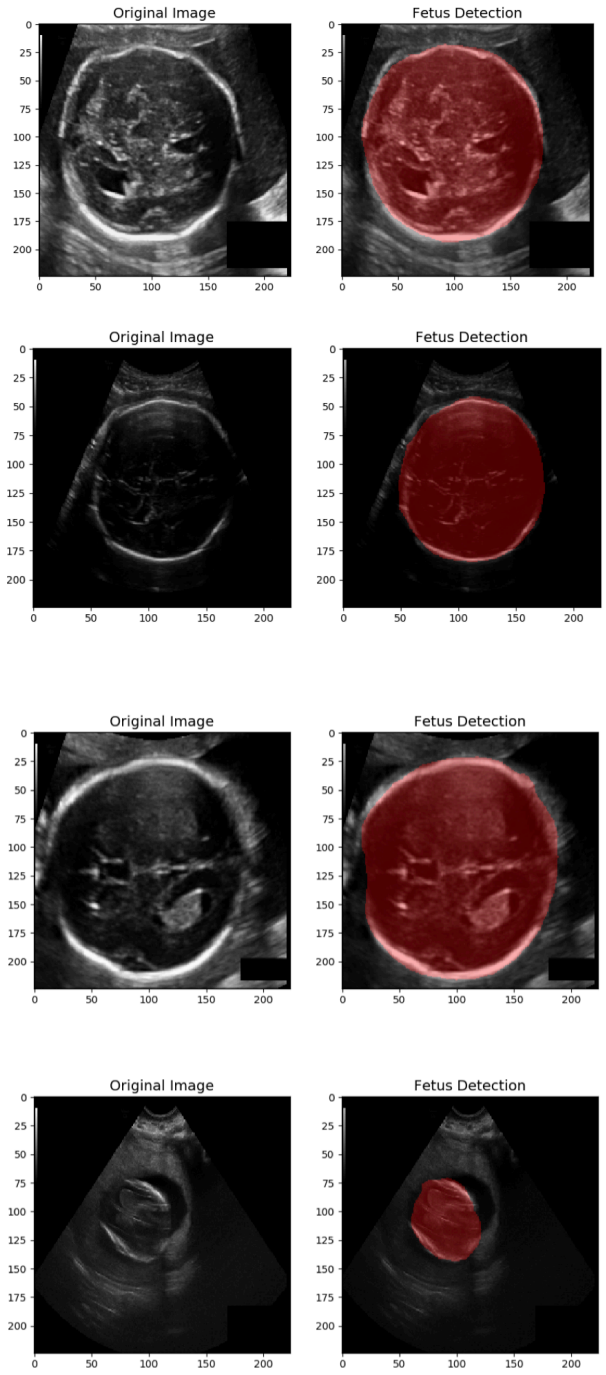


Figure 9 Output from FCN

7. Limitations of the solution –

The application works good for fetal head segmentation we did not have a large dataset of ultrasound images without the fetus. Otherwise we could have implemented the method of Logistic Regression to first find out that whether the image has fetus or not.

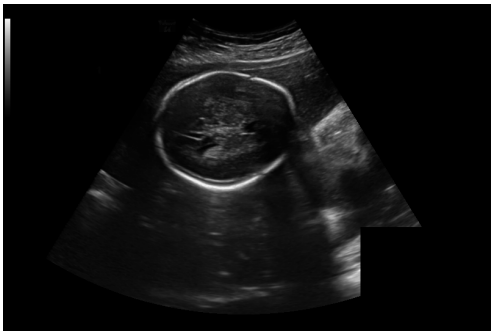
8. Conclusion and future work

8.1 Conclusion

After trying out many methods of image segmentation including methods like k-means, fuzzy c-means, contour detection and FCN compiling the results from these methods. We can easily conclude that the results from FCN are the only results that are viable for the purpose of image segmentation. The results from K means were acceptable to some extend and Contour Detection did not produce the desired results.
Whereas the FCN was able to achieve a Per Pixel Accuracy (PPA) of 97% and Jaccard Index of 0.92.

8.2 Future Work

The work here can be extended to find the area or the circumference of the fetus in the image. This will help the radiologist to estimate the age of the fetus.
However, we did not proceed with this concept because the dataset had varying size of the ultrasound image which is likely due to the different images produced by the different brand model and the result of human error. This is evident in the images below.



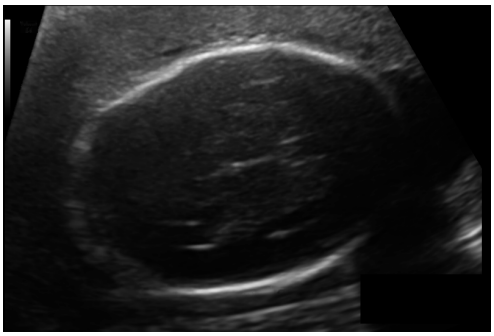


Figure 10 Comparison of Image variations

Also, the model can be extended to organ detection, like kidney stones, gall bladder stone, etc. This will help in automating a lot of medical applications.

9. Statement of Individual Contribution-

We did all the work as a team. The contribution of the team members is as follows-
Mohit Jain(mohitj2)- 0.5 or 50%
Shounak Ray(sray10)- 0.5 or 50%

References

[1] P. Burai and B. Harangi, "Pixelwise segmentation of uterine wall in endoscopic video frame using convolutional neural networks," Proceedings of the 10th International Symposium on Image and Signal Processing and Analysis, Ljubljana, 2017, pp. 95-98. Frobnication revisited. Journal of Foo, 13(1):234–778, 2003.

[2] X. Fu and H. Qu, "Research on Semantic Segmentation of High-resolution Remote Sensing Image Based on Full Convolutional Neural Network," 2018 12th International Symposium on Antennas, Propagation and EM Theory (ISAPE), Hangzhou, China, 2018, pp. 1-4.

[3] P. P. Banik, R. Saha and K. Kim, "Fused Convolutional Neural Network for White Blood Cell Image Classification," 2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC), Okinawa, Japan, 2019, pp. 238-240.

[4] [4] J. Rathod, V. Waghmode, A. Sodha and P. Bhavathankar, "Diagnosis of skin diseases using Convolutional Neural Networks," 2018 Second International Conference on Electronics, Communication and Aerospace Technology (ICECA), Coimbatore, 2018, pp. 1048-1051.

[5] https://github.com/lutzroeder/netron

[6] https://zenodo.org/record/1327317#.XmqJcS2ZPAI