Turkish Journal of Computer and Mathematics Education *Vol.12 No.6 (2021), 3475-3487*
Research Article

Effective Deep Learning Model to Identify Harmful Bacteria in Blood Samples for Healthcare Application

Dr. G. Naveen Sundar[1], Dr. D. Narmadha[2], PriyaElizabeth Jose[3],AswathiNair[4]
[1,2]Assistant Professor, Department of CSE, Karunya Institute of Technology Sciences, Coimbatore
[3,4]UG student, Department of CSE, Karunya Institute of Technology Sciences, Coimbatore

Abstract: Like water, blood is also an important part of human life. It is the basic and most necessary life-sustaining system of most organisms as well as humans. The basic functionality of blood is to nourish the organs and provide essential nutrients required by the system. The contamination of blood has a very harmful and invasion effect in human as well as other organisms' life. To tackle this situation of identifying the contaminants of blood (mainly bacteria and pathogens), we are trying to come up with a deep learning model, which can identify various bacteria and pathogens in the blood samples, thus saving many lives.

Keywords: Bacterial identification in blood sample,Deep Learning, Convolutional Neural Networks, Fully Convolutional Network, U-Net.

1. Introduction

Blood can be termed as an essential body fluid present in humans and other animals, whose basic and important function is to carry all the vital nutrients required to the whole part of the body. The basic question is what would happen if the blood is contaminated? The obvious answer to that question is that our body will show reaction according to the level of contamination. But running all through the lab tests and all may threaten one's life immeasurably. Especially, when the blood is contaminated with bacteria, it causes a serious condition called Septicaemia, which can lead to sepsis and also cause deaths in humans. To help tackle this situation, we are trying to propose an effective healthcare technology that makes use of deep learning algorithms which will help to quickly identify the definitive presence of bacteria or not, telling us the contamination of blood based on a microscopic image.

Neural networks are a model in machine learning which attempts to mirror a particular side of the human brain. They are the basic foundationfor most of the methods of deep learning, which is a part of machine learning that includesseveralchronological layers that data run through to perform classification or pattern analysis. Deep learning has made strides in many areas such as natural language processing, bioinformatics, and computer vision. Because CNN's are trained in data, they form complex connection of node. Convolutional neural networks (CNNs) epitomize a unique subset of neural networks and deep learning.

Over the years, Convolutional Neural Networks have become a popular deep learning algorithm for image and pattern recognition. Though there are many algorithms in deep learning, CNN is the most popular and effective one which will help in the classification based on images. CNN's perform supervised learning as they need to be provided with specific data to analyze. Our model attempts to do a quick estimation of blood samples without any chemical tests, stating if they contain bacteria or not.
We are attempting to fashion a deep learning model that merelyinvolves a microscopic image of the sample to see whether or not contaminants are present, thus improving the comfort with which one can perform basic blood analysis and determine the safety of our loved ones.

2. Related Work

Identifying the blood contaminants has been the basic priority of most researchers in science. Some of these research work led to the development of various technologies and founding, which led to other developments, helping to tackle the overall problems caused due to the contaminated bacteria in blood. The proposed aim is to find the bacteria sample in blood, which is a fluid, our research works went through various types of bacteria identification with different fluids, which includes water also.

Imam Utoyo, RiriesRulaningtyas, Budi DwiSatoto and Eko Budi Khoendori in their research in the identification of gram-negative bacteria in blood samples which uses Convolutional neural network with fine-tuning, used a variant of CNN, that is VGG-16 architecture, which used a combined sample of 2520 images from twoseparate classes. The dimensions of the data were 256x256 pixels and the amount of data used during each phase of training, testing, and validation were 840 imageswhich had a resolution of 96 points per inch, and a depth of 24 bits. The accuracy achieved duringthe training phasewas 99.20% [1].
The researchers of the Clean Water AI project Peter Ma,Shin Ae Hong,Justin Shenk and Sarah Hancreated an IOT device that determines the contamination level of water. The system itself included various hardware component like UP2 Grove IOT Development kit and many other. Along with various software services like Microsoft azure for machine learning training etc. They did this research with yeast instead of doing it with the bacteria. They followed this supervised learning to develop a deep learning algorithm which detects contamination images in real-time scenarios. This is still an ongoing project with various aspects of modifications [2].

Another research related to Rapid identification of pathogenic bacteria using Raman spectroscopy and deep learning promises to identify the species and antibiotic resistance of bacteria. This research promises label-free bacteria detection, identification, and antibiotic susceptibility and also helped to achieve treatment identification accuracies of 99.7% [3].
Another researcher Mohamed Abd Elaziz, Khalid M. Hosny, Ahmed A. Hemedan, and Mohamed M. Darwish used Fractional-order Orthogonal descriptors to identify various bacterial species. The algorithm they used was the Salp swarm algorithm (SSA) as well as teaching-based learning optimization (TLBO) as local operators which helped in the improvement of the exploitation ability of the SSA algorithm and they passed with a good result of accuracy [4].

Researches like KritSriporn, Cheng-Fa Tsai, Chia-En Tsai, and Paohsi Wang who didMalaria Disease analysisby means of Effective Deep Learning Approach used in total 7000 images. These images were trained with different CNN variants like Xception,AlexNet, ResNet-50, VGG-16 and Inception-V3for verification and analysis. And got the best analysis for Xception for 98.86% [6].
Deep Learning Methods for Classification of White Blood Cells for Diagnosing Disease by Muhammed Yildirim, Ahmet Çinar made use of most of the CNN variants like AlexNet, ResNet50, DenseNet201, and GoogleNet. AlexNet- %79.27, ResNet-50 - %78.74, DenseNet201- %77.88 and GoogleNet - %62.93 as result without applying Gaussian Filter and Median Filter [7].
A U-Net Based Fully Convolutional Networks model for Automatic Brain Tumor Detection and Segmentation by Hao Dong, Guang Yang, Fangde Liu, and Yuanhan Mo, helped in the pioneering development of a novel 2D fully convoluted segmentation network which wasbuilt on the U-Net architecture consisted ofcontracting path and an expanding path. There were five convolutional blocks in the downsampling or contracting path. Lastly in their research, a 1×1 convolutional layer wascast-off to minimize the number of feature maps to two that reflect the segmentation of the foreground and background respectively. The network did not invoke a fully connected layer. The model finally achieved an accuracy of 77% [8].

Researchers like Edouard A. Hay, Raghuveer Parthasarathy worked to determine the performance of convolutional neural networks for the identification of bacteria in 3D microscopy datasets which worked on the 3D images bacteria in the gut of the zebrafish have reached a precision of about 75% on Vibrio images and tests on Pseudomonas. They used several sets of test images and compared the performance of convolutional neural networks to Random Forest and the vector support machine. In their experiment, they found that neural network outperforms the feature-based algorithm [10].
In a work for recognition of malaria parasites in dried human blood spots, the researchers were able to obtain the dried blood spots from malaria inspection in 12 regions in Southeastern Tanzania in the year 2018/19. The evaluation taken into consideration 296 individuals, inclusive of 123 showed malaria and 173 negatives.Among the algorithms used, Logistic regression in conjunction with mid-infrared spectroscopy become the best-acting version which performed universal prediction and achieved 92% for predicting infectious (specificity = 91.7%; sensitivity = 92.8%) and 85% for predicting blended infections[11].

Researchers like SigalTrattner and Hayit Greenspan used imaging analysis, along with statistical modeling tools in a general framework for visual array analysis to automatically identify the bacterial types. The proposed

framework mechanicallyanalyzed the visible array information and bestowed recordsapproximately the spot places and categorizations, in addition to phage profiles of the bacterial types[12].

Also, since our dataset is related to the image segmentation problem, we tried to focus on various image segmentation algorithms which use Convolutional Neural Networks. Image segmentation is the procedure of partitioning an image or a virtual image into more than one segments or set of pixels additionally referred to as image objects. In our model, this image segmentation process will be used to identify bacteria, blood cells, and noise.
From our research and findings, we found we have various other frameworks or algorithms out of which DeepLab and U-Net were the most efficient ones for image segmentation problems. If you compare U-Net and DeepLab, we can conclude that U-Net is more efficient than DeepLab because it is built upon the concept of Fully Convolutional Networks (FCN). A Fully Convolutional Networks can effectively learn how to make dense predictions for per-pixel tasks, especially in Biomedical images.
Going through all the journals and researches we found out the best model for image segmentation coming under deep learning model, especially Convolutional Neural Network (CNN) is U-Net.

3.Methodology

The paper entails more than one steps beginning from downloading the dataset from Kaggleand importing it to drive from where we import the contents of the drive to Google Colab. The dataset is of Spirochaeta, a genus of bacteria that is found on animal blood. The dataset consists of 366 dark field microscopy images and manually annotated masks. Subsequently the dataset wasseparated into train and test set, this is 275 and 91 images respectively. The U-Net model was trained on the training dataset with an epoch value of 10and a batch length of 16.

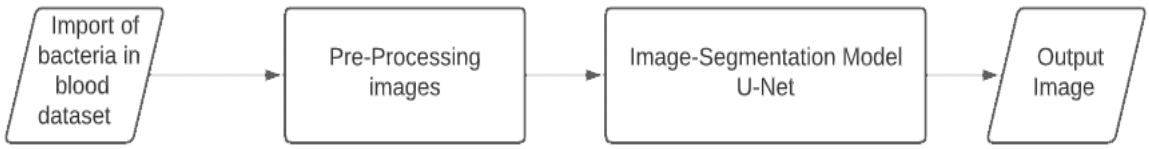


Figure 1.A flowchart for the proposed work

Algorithm

Step 1: Mount the drive onto Colab and check for the working directory if needed.
Step 2: Download the dataset and unzip it. Load the dataset containing two folders 1. Images, 2. Masks
Step3: Do one-hot encoding on the masks by invoking
tensorflow.keras.utils.categorical and the masks from the data/masks folder were one hot encoded and saved to data/ dataForModel/encodedMasks/encodedMasks folder.
Step4: Perform image augmentation to get a bunch of data for training and designing a robust model. Image augmentation is done by performing rotation, flipping etc on the data. This was performed using tensorflow. keras. preprocessing. image. ImageDataGenerator
Step5: Fixing the image width and height as 128x128, in batches of 16 images, 75% of images was used for creating the training set and the rest 25% for validation.
Step6: The train datagen for images and labels was created.
Step7: The training dataset and the validation dataset were created by zipping together the output of the image generator and label generator respectively.
Step8: Creating the U-Net Model consisting of downsampling (encoding) and upsampling(decoding). The downsampling with 5 convolutional blocks with filter size of 3x3 with activation function as Relu and padding kept as same
Step9: The final layer SoftMax activation function as it is a multiclass classification on each pixel problem. The kernel initializer is taken as he_normal to assume normal distributed data.

Step10: The model was compiled by taking the loss as categorical_crossentropy and the optimizer used for optimizing the gradient during training and backpropagation was adam.
Step11: We training three models, that is, 1. With callbacks, 2. Without callbacks, 3. With custom loss and 4. Optimized architecture with custom loss.
Step11: Each model was trained for 10 epochs taking 100 steps per epoch except for the fourth model which was trained for 20 epochs taking 100 steps per epoch.
Step12: Plotting the graph for loss and accuracy for each model after training the model.
Step13: Finally getting the accuracy of each model, and evaluating the best one with more accuracy**.**

3.1. Dataset

The dataset is accessible on Kaggle under the title "Bacteria Detection with Darkfield Microscopy". This 162MB dataset contains 366 images. The images in this dataset are in RGB format with dimensions 128x128. The dataset involves of two types of images, that is dark field microscopy images and manually annotated masks. The dataset is presented in the following weblink:

https://www.kaggle.com/longnguyen2306/bacteria-detection-with-darkfield-microscopy?select=images

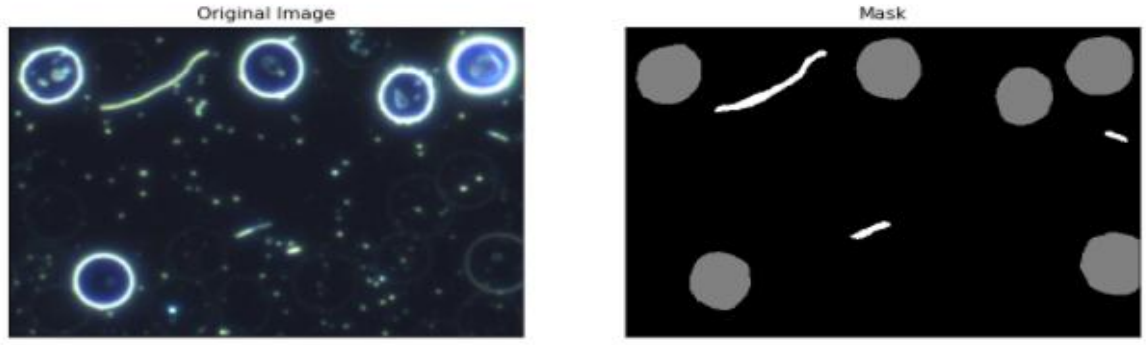


Figure 2. Shows the dark field microscopic images of source image and the annotated masked image

Figure b. consists two images where we have a source image and an annotated masked image. The dataset was gathered and annotated by students as a hands-on experience. The dataset itself is based on multiclass classification, identifying blood and bacteria. We cannot perform a normal classification problem because this dataset tries to solve it as a segmentation problem. The classification itself occurs on each pixel, identifying blood or bacteria. The dataset is of Spirochaeta, a genus of bacteria that is found on animal blood.

3.2 Fully Convolutional Networks

A Fully Convolutional Network is a neural network that performs only convolutional operations along with upsampling. FCN is a CNN without fully connected layers.

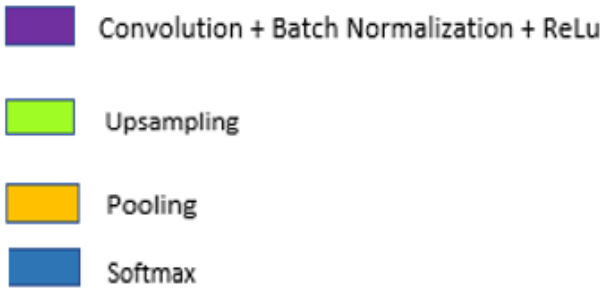


Figure 3. FCN Architecture

3.2.1 Down Sampling

The downsampling consists of repeated application of 3x3 convolutions, as in normal convolutional functions. The main objective of this function is to extract high-level features from the input data and performs a convolutional neural network to change image pixels to pixel categories. Each of these functions is followed by rectified linear unit (ReLu) layer. It helps the model to learn faster and performs better. And finally, a 2x2 max pooling operations with stride 2. This function helps to calculate the maximum or largest value in each feature map. The downsampling performs normal convolutional functions. The only difference is that the model doesn't perform fully connected functions (FC).

3.2.2 Up Sampling

Up- Sampling can be termed as to get the output size larger. Also known as up convolution and transposed convolution. Eachphase in the expanded path contains of an upsampling of the feature map shadowed by a 2x2 convolution. This may also know as up-convolution. It reduces the number of feature channels to half, a concatenation with the correspondingly picked feature map from the downsampled, and two 3x3 convolutions, each followed by a ReLU. At the final layer, a 1x1 convolution is used to get the preferred range of classes.

3.3 U-Net Architecture

U-Net is an architecture that uses the concept of Fully Convolutional Networks, that is convolution without the use of fully connected layers. It is used for semantic segmentation and also has proved to be great in use in Biomedical image segmentation problem. This model consists of the contracting path as well as the expanding path. The contracting path or the downsampling uses the typical convolutional network, while the expanding path or upsampling uses the transposed convolutional method (standard convolutions with modified input feature map).

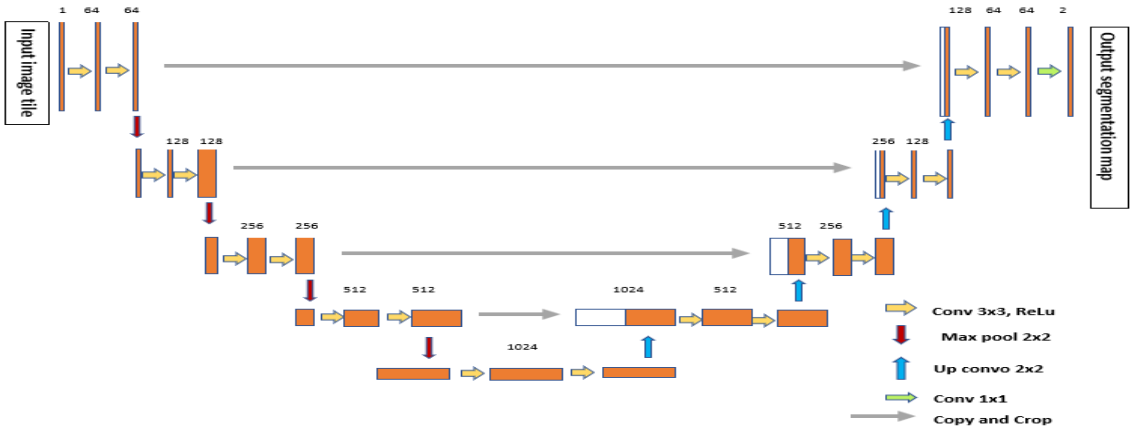


Figure 4. The proposed model

3.3.1 Image Pre-processing

Image pre-processing aims at improving the image data by enhancing the desired features of the image and suppressing the undesired features of the image. The loaded dataset contains two folders, one which contains images and the other which contains masks. The problem was of the image segmentation on the pixel categories as mentioned in the masks.json file. The given dataset was clean and hence not much of the data pre-processing was required. In total there were three categories for the pixels to segment, that is bacteria, blood, and background (noise). Since this is a categorical data, and most of the machine learning algorithm does not use categorical data directly, the masks need one-hot encoding which allows the categorical data to be more expressive.

3.3.2 Image Augmentation

Image augmentation is a method that is used to increase the size of the training dataset by creating modified versions of the image. This technique helps to improve the performance and ability of the model. This includes cropping, padding, horizontal flipping of the available dataset without actually collecting new data. In batches of 16 images, 75% of images were used for creating the training set and the rest 25% images were reserved for validation. The image size was fixed at 128x128. The types of augmentation performed in this model were rotation range, width shift range, shear range, height shift range, horizontal flip, vertical flip, and zoom range.

3.3.3 Model design

The task in hand of the model is to predict the masked labels. The contracting path follows the following steps simultaneously

convo_layer1 -> convo_layer2 -> max_pooling -> dropout_layer(optional)

Since the dropout layer is optional, we are going to train our model without the dropout layer. A dropout is a technique where randomly selected neurons are left during the training part of the model. Also, we will analyze the model with the use of dropout in the later part of implementation. The contracting part of the U-Net model has five convolutional layers, each process constitutes two convolutional layers. These convolutional processes will increase the depth of the image. The max-pooling layer halves down the size of the image. The implementation here uses padding=" same".

The expanding path follows the following steps simultaneously

conv_2d_transpose->concatenate-> convo_layer1 -> convo_layer1

In this portion of the model, the image will be upsized to its original size. To upsize the image, we use a technique called transpose convolution. After this process, the image output of the transpose convolution part is then concatenated with the corresponding image from the contracting path. The reason for combining the images from the contracting as well as expanding path is to combine the information from the previous layers will help to get a more precise prediction.

3.3.4 Model 1: With callbacks

Overtraining happens when a model starts to foresee training examples with high accuracy but cannot generalize to new data, which leads to poor performance in the field. In our case, this overtraining happens because the data is too homogenous. To tackle this situation, we are going for an early stopping method. During training, the model is evaluated on the holdout validation dataset after each epoch. If the performance of the model starts to degrade, then the training process is stopped. A large number of epochs can lead to the overfitting of the model, also too less may result in an underfit model. Early stopping technique helps to specify a random large number of training epochs and breaks the training once the model performance stops improving on a holdout validation dataset. The metrics on which we check the performance of our model is based on accuracy.

Overtraining of the model leads to an overfit or underfit model. Overfitting refers to a model that trains the data too well that by hearts all the data in such a way that it impacts the performance of the model on any new data that is, the model will be designed to fail in any new data. Also, we are taking patience value as 2 as in the training will be stopped for the number of epochs with no improvement. A callback can be termed as a set of functions which is applied in the training stage of the model. This comprises of stopping the training when you reach certain accuracy or loss score and saving the best model in our case as a checkpoint after each successful epoch.

3.3.5 Model 2: Without callbacks

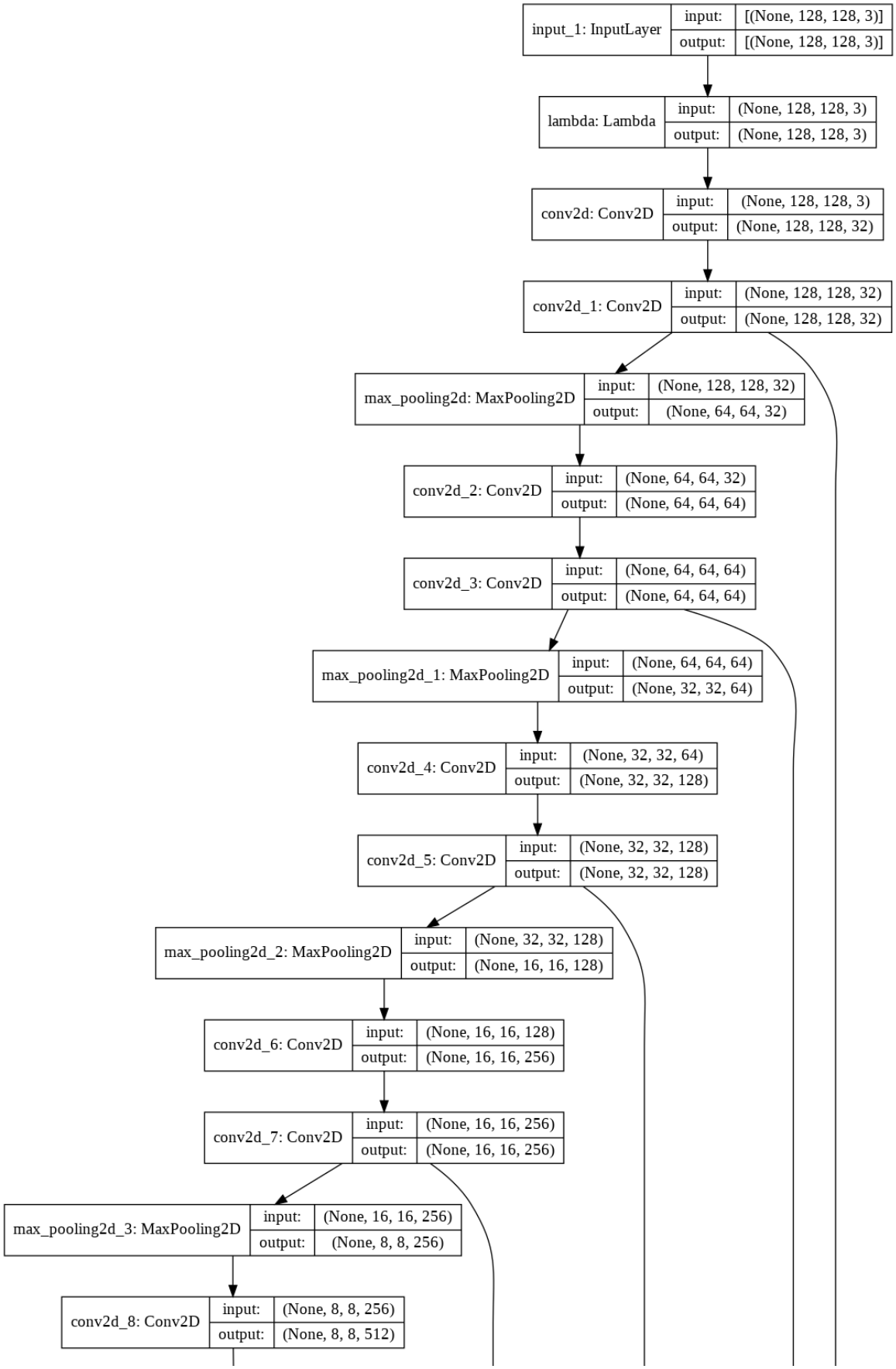
With the help of a callback function and early stopping, one can avoid the two problems in the model, one is an overfit model and the other is an underfit model. Since this model is executing without the callback function, it trains through 20 epochs in total. Since early stopping is considered as a method of regularization to avoid overfit model, without this function, the learner cannot determine if the model begins to overfit or not.

3.3.6 Model 3: With custom loss

Our model here predicts three things that is, bacteria, blood cell, and the background. It so may happen that while predicting, the accuracy of the model is increased due to high weightage in prediction of the background rather than predicting the bacteria and the blood cell. Also, sometimes the accuracy may be increased just due to predicting the background as it has more weightage, and also it may happen that our model may not be able to predict the bacteria and blood itself. To tackle this problem in this model we are creating a loss function. In this, we are assigning each label class with an equal weight of 0. Passing all the images in an iterator, we are capturing all the unique pixel values of the three classes. Then by increasing the count of the unique class labels in every image mask we are updating the weights dictionary with this count. Then we update the final weight by dividing the total count of label classes by the count of respective label classes. This reduces the weight of the background label class while increasing the weight of the blood and bacteria class, thus helping to predict accurately.

3.3.7.Model 4: Optimized architecture with custom loss

In Model 1, 2 and 3 architecture, we had in total four parts of two convolution layer and one max-pooling layer and also one part of two convolutional layer alone in the contracting part. And in the expanding part had a total of four parts of one deconvolution, one concatenation operation and two convolutional layers and also, the same architecture had one convolutional layer as the output image. Like convolution have a deconvolution operation as the reverse of it, we have reverse of max-pooling as Upsampling. In the architecture for the fourth model, we had this same contracting part as the model 1, 2 and 3. But when we are going to the expanding part of the model, instead of using the convolutional transpose or deconvolution, the concatenation operator and two convolutional layers, we are reducing the number of convolutional layers there and are using Upsampling layer. In short, instead of raising the size of the image using a convolutional operation, we are raising it by using an Upsampling layer so as to reduce the operational cost. The expanding path hence consists of nine transpose convolution, four concatenation operation and four Upsampling layer. Also, we are taking the loss as the weight loss calculated in the third model. The filter size used in this model also differs from that of the first 3 models. The first 3 models had the starting filter size as 16, but for this fourth model we are using the filter size as 32.


input_1: InputLayer
input: [(None, 128, 128, 3)]
output: [(None, 128, 128, 3)]
lambda: Lambda
input: (None, 128, 128, 3)
output: (None, 128, 128, 3)
conv2d: Conv2D
input: (None, 128, 128, 3)
output: (None, 128, 128, 32)
conv2d_1: Conv2D
input: (None, 128, 128, 32)
output: (None, 128, 128, 32)
max_pooling2d: MaxPooling2D
input: (None, 128, 128, 32)
output: (None, 64, 64, 32)
conv2d_2: Conv2D
input: (None, 64, 64, 32)
output: (None, 64, 64, 64)
conv2d_3: Conv2D
input: (None, 64, 64, 64)
output: (None, 64, 64, 64)
max_pooling2d_1: MaxPooling2D
input: (None, 64, 64, 64)
output: (None, 32, 32, 64)
conv2d_4: Conv2D
input: (None, 32, 32, 64)
output: (None, 32, 32, 128)
conv2d_5: Conv2D
input: (None, 32, 32, 128)
output: (None, 32, 32, 128)
max_pooling2d_2: MaxPooling2D
input: (None, 32, 32, 128)
output: (None, 16, 16, 128)
conv2d_6: Conv2D
input: (None, 16, 16, 128)
output: (None, 16, 16, 256)
conv2d_7: Conv2D
input: (None, 16, 16, 256)
output: (None, 16, 16, 256)
max_pooling2d_3: MaxPooling2D
input: (None, 16, 16, 256)
output: (None, 8, 8, 256)
conv2d_8: Conv2D
input: (None, 8, 8, 256)
output: (None, 8, 8, 512)

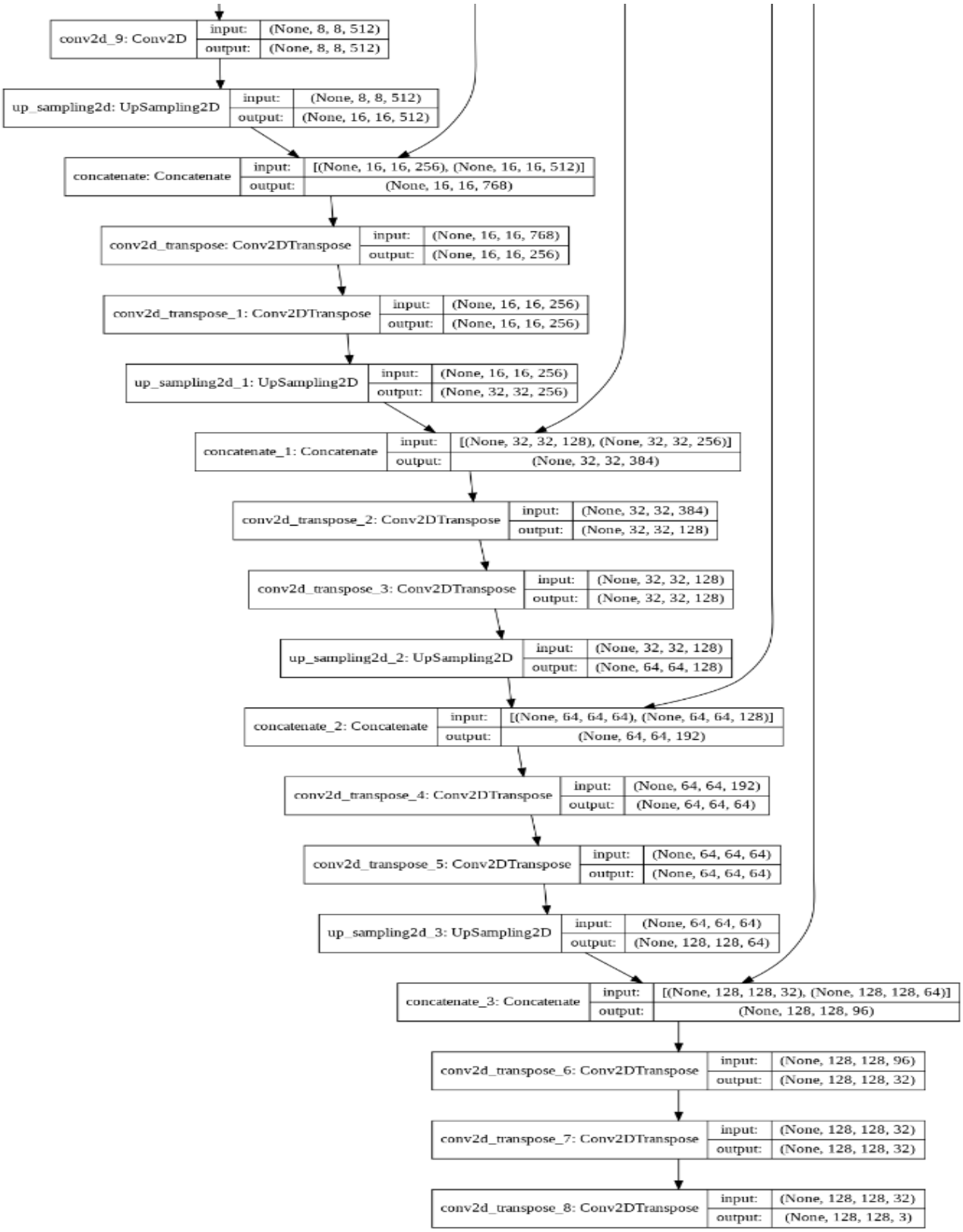


Figure 5. Model 4 graph

4. Experimental Model and Discussions

During the development of the project on Bacteria identification in blood samples, four models were trained and tested with dataset consisting of 366 images, in which 275 of them were used for training and 91 were used for validation sets. The same method has been used for the data preprocessing step for all three models. The only difference is that in the first model we have a callback function for early stopping, in the second model we don't have the concept of early stopping, while in the third model we created a loss function to upsurge the accuracy of the model. Performance metric used to analyze the model was Accuracy. Accuracy can be termed as the measure or the fraction of pixels in the image which were correctly classified. Below we have a table showing the performance of each model in terms of accuracy metric.

	Model	Accuracy
0	Model1 with callbacks	93.18251013755798
1	Model2 without callbacks	95.99214792251587
2	Model3 with custom loss	96.5478777885437
3	Optimized architecture with custom loss	96.60601615905762

Figure 6. Shows the value of performance measure in terms of accuracy of each model

During the training of all the four models, the training correctness and training loss ofModel 1 were 92.41% and 22.92% individually. The validation accuracy achieved by Model 1 is 93.18% whereas the validation loss is 24.06%. Similarly, for Model 2, the value for training accuracy achieved is 96.62% and training loss is 9.81%. The validation accuracy and loss obtained by Model 2 is 95.99% and 16.17% respectively. The training accuracy and training loss of Model 3 were 96.58% and 3.31% respectively. The validation accuracy achieved by Model 3 is 96.54% whereas the validation loss is 5.75%. Model 4 outperforms Model 1 (with callbacks), Model 2 (without callbacks) and Model 3 (custom loss function) by achieving training accuracy and loss as 96.78% and 3.21% respectively. Model 4 achieved the validation accuracy of 96.60% and loss is 3.75%. Model accuracy and model loss graph of each model is depicted in figure g, h, i and j.

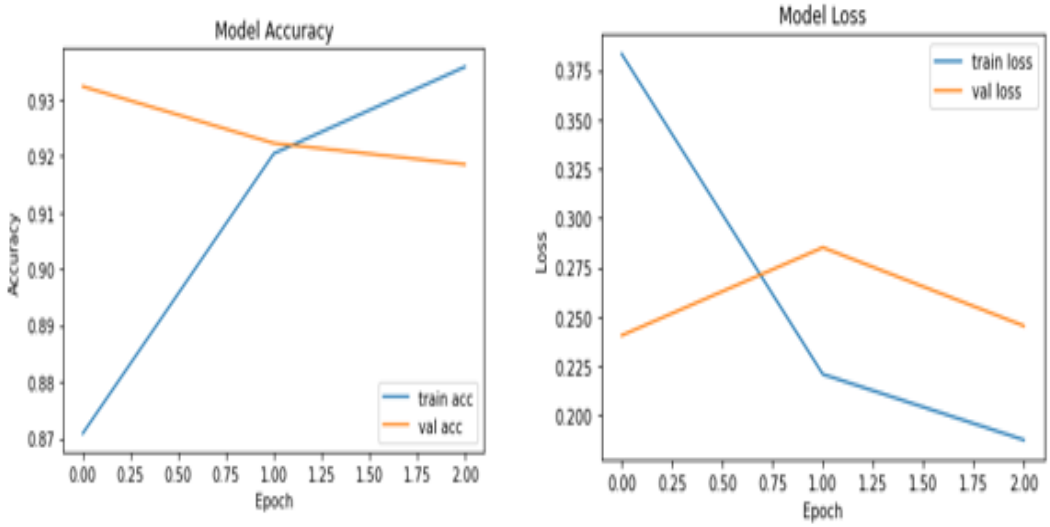


Figure 7. Model 1 accuracy and loss graphs

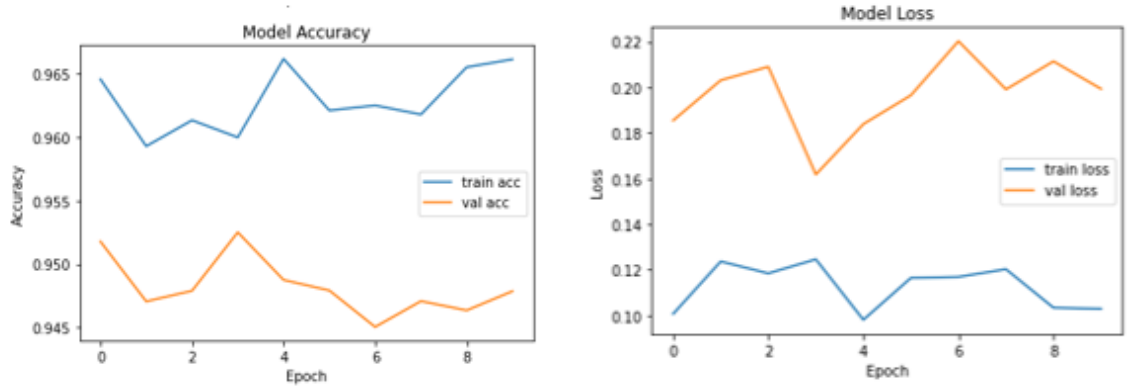


Figure 8. Model 2 accuracy and loss graphs

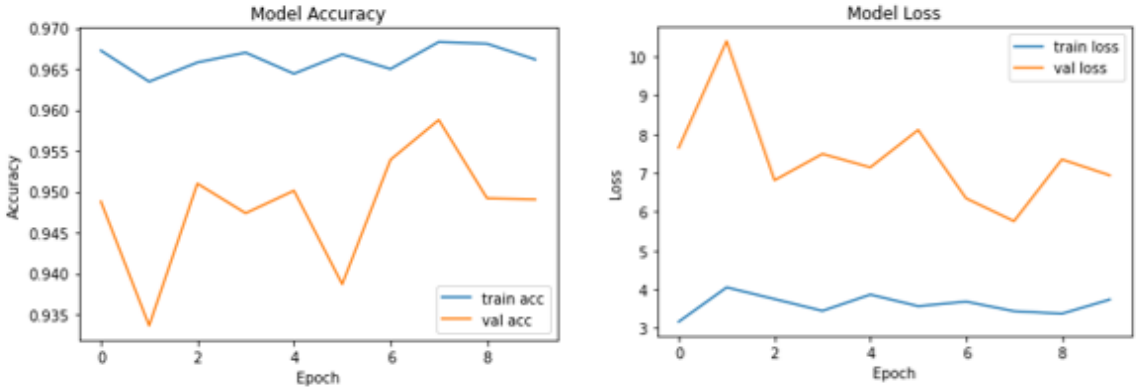


Figure 9. Model 3 accuracy and loss graphs

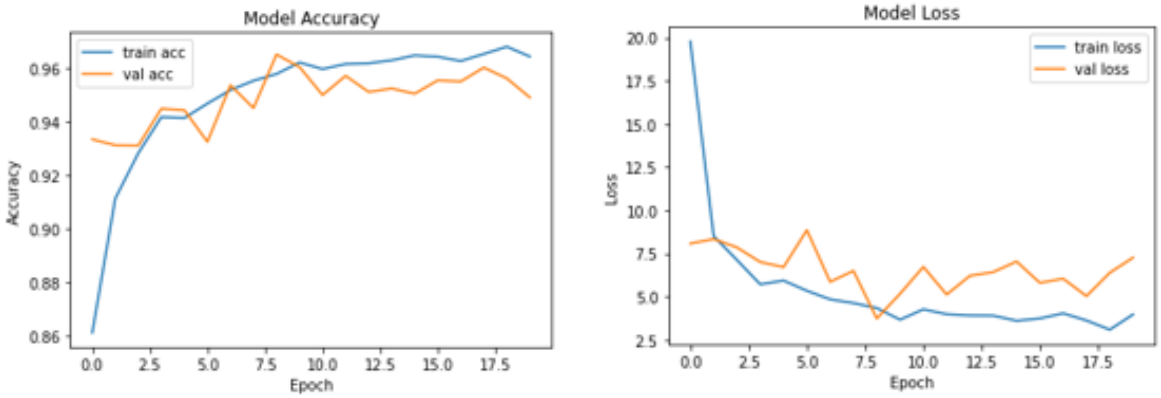


Figure 10. Model 4 accuracy and loss graphs

Table 1. Shows the related works and their accuracy

PAPER NUMBER	AREA	PROCEDURE	ACCURACY	OUR OUTCOME
[8]	Automatic Brain Tumor Detection and Segmentation	U-Net Based Fully Convolutional Networks	77%	96.60%
[7]	Classification of White Blood Cells	Deep Learning Methods (AlexNet, ResNet-50, DenseNet201 and GoogleNet)	AlexNet- %79.27, ResNet-50 - %78.74, DenseNet201- %77.88 GoogleNet - %62.93	96.06%
[10]	Identification of bacteria in 3D microscopy datasets	Convolutional neural networks	~ 75%	96.60%
[11]	Detection of malarial parasites in dried human blood spots	Using mid-infrared spectroscopy and logistic regression analysis	For predicting mixed infections= 85% Predicting single infections=92%	96.60%

5. Conclusion and Upcoming Works

This study extends a high-yielding neural network for image segmentation problem. Model 3 which creates a loss function for the imbalanced dataset comes with high validation accuracy with 96.60% and loss percent of 3.75%. The proposed model will be very useful in the field of microbiology, especially for the image segmentation problem. This model will be very much helpful in the early detection of bacteria in blood samples which will be time and cost-effective. A great number of blood sample images can be processed very speedily to deliver accurate diagnosis, thus helping patients and saving many lives. Since this model helps in the identification of a single bacteria, that is Spirochaeta, future works can also include other strains of microscopic bacteria or pathogens.

The future work of the author of this paper aims at obtaining higher prediction based on accuracy and minimizing the loss by the inclusion of the dropout layer and by using the U-Net++ model for better accuracy. Since this dataset was kind of imbalanced, the complete performance of the model can be enhanced with the use of a larger and balanced dataset.

References

[1] Satoto, B. D., Utoyo, I., Rulaningtyas, R., &Khoendori, E. B. (2020). An improvement of Gram-negative bacteria identification using convolutional neural network with fine tuning. *Telkomnika*, 18(3), 1397-1405.

[2] Nehal, S. A., Roy, D., Devi, M., & Srinivas, T. (2019). Highly sensitive lab-on-chip with deep learning AI for detection of bacteria in water. *International Journal of Information Technology*, 1-7.

[3] Ho, C. S., Jean, N., Hogan, C. A., Blackmon, L., Jeffrey, S. S., Holodniy, M., ... & Dionne, J. (2019). Rapid identification of pathogenic bacteria using Raman spectroscopy and deep learning. *Nature communications*, *10*(1), 1-8.[4] ASOC 106504 Improved recognition of bacterial species using novel fractional-order orthogonal descriptor

[4] Razzak, M. I., Naz, S., &Zaib, A. (2018). Deep learning for medical image processing: Overview, challenges and the future. *Classification in BioApps*, 323-350.
[5] Sriporn, K., Tsai, C. F., Tsai, C. E., & Wang, P. (2020). Analyzing Malaria Disease Using Effective Deep Learning Approach. *Diagnostics*, *10*(10), 744.
[6] Yildirim, M., &Çinar, A. (2019). Classification of White Blood Cells by Deep Learning Methods for Diagnosing Disease. *Revue d'IntelligenceArtificielle*, *33*(5), 335-340.
[7] Dong, H., Yang, G., Liu, F., Mo, Y., & Guo, Y. (2017, July). Automatic brain tumor detection and segmentation using u-net based fully convolutional networks. In *annual conference on medical image understanding and analysis* (pp. 506-517). Springer, Cham.
[8] Ronneberger, O., Fischer, P., &Brox, T. (2015, October). U-net: Convolutional networks for biomedical image segmentation. In *International Conference on Medical image computing and computer-assisted intervention* (pp. 234-241). Springer, Cham.
[9] Hay, E. A., & Parthasarathy, R. (2018). Performance of convolutional neural networks for identification of bacteria in 3D microscopy datasets. *PLoS computational biology*, *14*(12), e1006628.
[10] Mwanga, E. P., Minja, E. G., Mrimi, E., Jiménez, M. G., Swai, J. K., Abbasi, S., ... & Okumu, F. O. (2019). Detection of malaria parasites in dried human blood spots using mid-infrared spectroscopy and logistic regression analysis. *Malaria journal*, *18*(1), 1-13.
[11] Trattner, S., Greenspan, H., Tepper, G., &Abboud, S. (2004). Automatic identification of bacterial types using statistical imaging methods. *IEEE transactions on medical imaging*, *23*(7), 807-820.