Automated Blood Cell count from Blood Smear Images for COVID-19 Diagnosis

Parshin Shojaee *
Noushin Omidvar*
parshinshojaee@vt.edu
Omidvar@vt.edu
Virginia Tech
Blacksburg, VA, USA

1 Problem Statement & Aims

The COVID-19 pandemia has to date reached more than 156 million confirmed cases (probably a much higher number of infected), and almost 3.27 million deaths. We all know that the recent outbreak of the COVID-19 is an urgent global concern. To deal with it, healthcare professionals need to make rapid and accurate decisions on the COVID-19 diagnosis, treatment, and isolation needs. Therefore, many researchers have been recently attracted to provide techniques that can improve the mentioned decisions.The current gold standard test for COVID-19 diagnosis is amplification of viral RNA by (real time) reverse transcription polymerase chain reaction (rRT-PCR). However, it presents known shortcomings such as long turnaround times (3-4 hours to generate results), potential shortage of reagents, false-negative rates as large as % 15-20 , the need for certified laboratories, expensive equipment and trained personnel. Thus there is a need for alternative, faster, less expensive and more accessible tests. A recent work has shown that a simple blood test might help to reduce the false-negative rRT-PCR tests [1]. Blood tests also can be used in developing countries and in those countries suffering from a shortage of rRT-PCR reagents and/or specialized laboratories as an inexpensive and available alternative to identify potential COVID-19 patients. However, performing regular blood tests also requires medical experts.

To leverage the advantage of blood analysis and reduce the cost of performing blood tests, researchers are recently working on the automated blood-based COVID-19 diagnosis tests. In this study, the blood sample image of each patient will be considered as a new testing sample, and the probability of being diagnosed with positive COVID-19 (diagnosis label) will be the class predicted for that test sample. Figure 1 shows a graphical representation of the pipeline of this project. Based on this figure, it can be observed that we will utilize two data sets (*i*) data set of blood cell smear images with annotations (rectangular box and labels) for blood cells (WBC, RBC, and Platelets) and labels for the WBC sub-types (Neutrophil, Monocyte, Lymphocyte, Eosinophil); and (*ii*) data set of COVID binary classification having features of blood cell and sub-types count. In this project, we have three main tasks as follows: (1) Detection of blood cell objects (WBC, RBC, Platelets) in the collected smear image; (2) Classification of WBC sub-types based on the cropped sub-images from detected box for each WBC; and (3) COVID-19 Diagnosis based on the features collected from task (1) and task (2).

*Both authors contributed equally to this research.

2 Data Description

In this project, we will exploit two data sets. The first data set would be the combined version of two data sets: https://github.com/Shenggan/BCCD_Dataset and https://www.kaggle.com/paultimothymooney/blood-cells that contain a total of 17,092 annotated and labeled images of blood cells. In this data set, each blood cell object (WBC, RBC, and Platelets) are annotated with a rectangular ground truth box and corresponding labels. Moreover, in this data set, the cropped pictures of each WBC object is also labeled as following four groups: Neutrophil, Monocyte, Lymphocyte, Eosinophil. The size of the images is 540×960 pixels, in format of jpg. The second dataset http://zenodo.org/record/3886927#.YFqiLkhKjVp is consisted of routine blood-test results performed on 1,925 patients on admission to the ED at the San Raffaele Hospital (OSR) from February 19, 2020, to May 31, 2020. For each sample, COVID-19 diagnosis was determined based on the result of the molecular test for SARS-CoV-2 performed by RT-PCR on nasopharyngeal swabs. The response of each COVID-19 test data sample takes a binary value $\{0, 1\}$ in case the COVID-19 test result is $\{negative, positive\}$, respectively. Table 3 represents the available features in this data set.

3 Data Pre-Processing

- Since we are using a 300 variant of single shot multibox detection for blood dection (SSD300) model for cell detection in images of task 1, input images to the model have to be (1) transformed into float tensors with size $3x300x300$; and (2) normalized to ImageNet images' RGB channels.
- For task 2 (classification of WBC sub-types), the data was highly imbalanced, i.e., some sub-types have large number of samples while others don't. Therefore, to

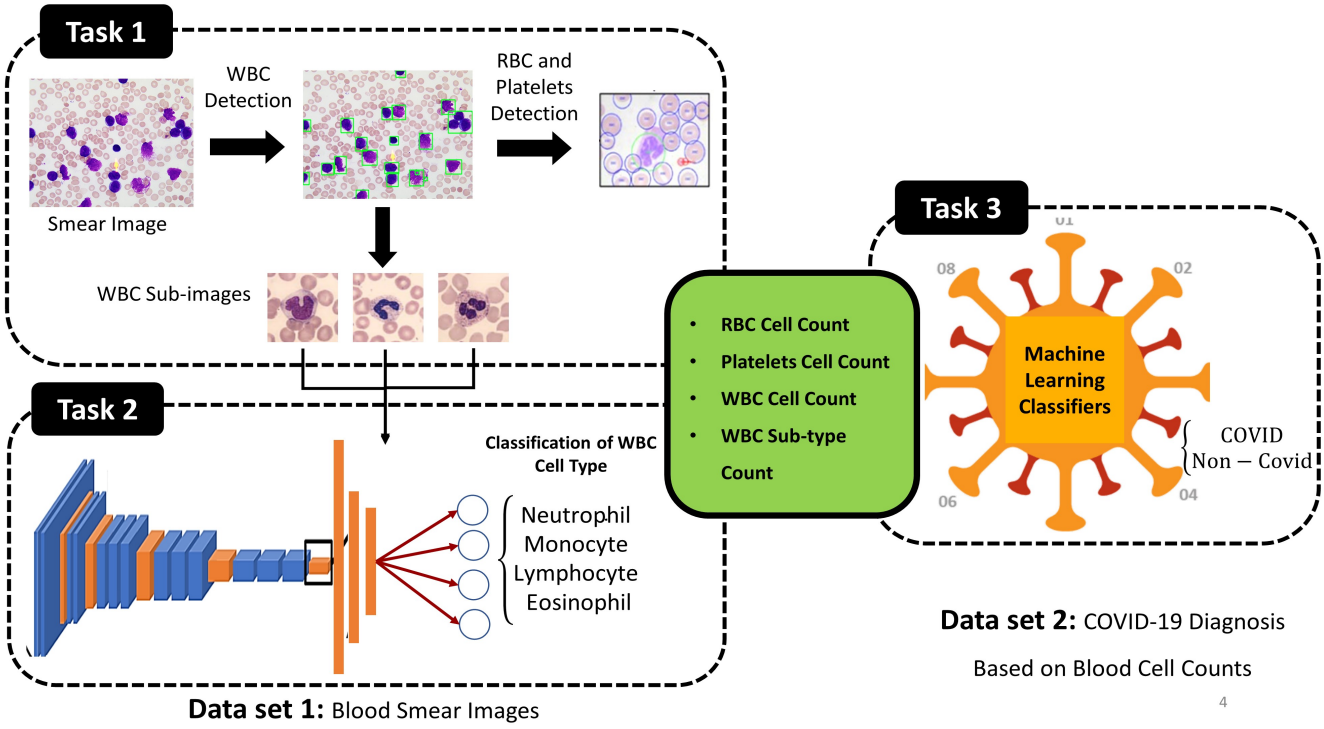

Figure 1. Graphical Representation of pipeline of this project

make the data set balanced, we needed to augment it by adding rotation, mirroring, random cropping, and shearing so that all the WBC sub-types will have the same number of image samples. Th data augmentation will increase the number of all data samples being imported to the CNN model, and make the train and validation data more balanced.

- In task 2, before importing images to the deep neural networks (DNNs), we also need to normalize the blood cell images by scale of 255.

Table 1. Feature Descriptions

Feature	Type
Gender	Categorical
Age	Categorical
Leukocytes (WBC)	Numerical(continuous)
Red Blood Cells (RBC)	Numerical(continuous)
Platelets	Numerical(continuous)
Neutrophils	Numerical(continuous)
Lymphocytes	Numerical(continuous)
Monocytes	Numerical(continuous)
Eosinophils	Numerical(continuous)
Basophils	Numerical(continuous)

4 Model Building

In this project, we will perform three main tasks. Each task and its corresponding models will be elaborated in the following paragraphs.

Task 1: The objective of task 1 is to detect the blood cell objects (WBC, RBC, and Platelets) in the blood smear images. Among all objects, the detection of WBC is more important since it will then be imported into task 2. We performed one unsupervised and two supervised object detection techniques. Figure 2 will represent all the steps that we performed in this task.

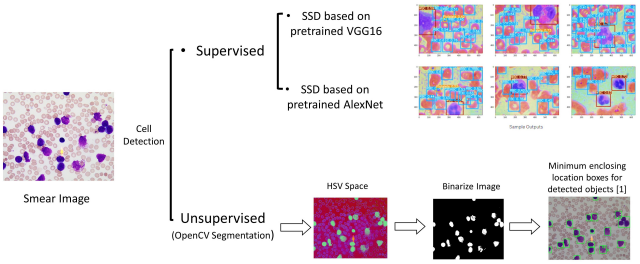

Figure 2. Graphical Representation of models utilized in Task 1

- **Unsupervised:** In the unsupervised technique, we tried to detect the WBC objects by OpenCV Segmentation. In this process, we first convert the smear images from RBG color space into the HSV color space. Then,

we binarized the image by OpenCV masking and then we tried to detect the rectangular WBC objects by minimum enclosing location boxes for detected objects [17]. After all these steps, we realized that overlapping WBCs will not be identified and the detection accuracy is very low (around 40 %). Therefore, we decided to utilize supervised techniques since the data set was annotated.

- **Supervised:** For the supervised approach, we used Single Shot Multibox Detector (SSD) method [7]. Single-shot models localize and detect objects in a single network, which make them significantly faster techniques. A SSD model uses multi-scale convolutional bounding box outputs attached to multiple feature maps at the top of the network to efficiently model the space of possible box shapes [7]. SSD network is consisted of three parts, base convolutions to extract lower level feature maps, auxilary convolutions which provide higher level feature maps and prediction convolutions to locate objects in these feature maps. As for the convolution based we used two different pretrained network architecture which already has shown good performance for image classification.
  - **SSD with VGG16 base**: a convolution network proposed by a group from university of Oxford trained on Imagenet database. This network has a depth of 16 layers the first 13 layers are convolutional layers with some followed by max-pooling and the last three layers are fully connected layers[14].
  - **SSD with AlexNet base** a convolution network architecture designed by Alex Kirzhevsky which contain eight layers; the first five are convolutional layers, some of them followed by max-pooling layers, and the last three fully connected layers krizhevsky2012ima

For both SSD with pretarined VGG16 base and AlexNet base, as implemented in original SSD paper, we removed the last two fully connected layers of these base convolutions and introduced two convolution layers to fit in to our object detection model. The object detection loss function for the training is a weighted sum of the localization loss (loc) and the confidence loss (conf)[7]:

$$L(x, c, l, g) = \frac{1}{N}(L_{conf}(x, c) + \alpha L_{loc}(x, l, g)) \qquad (1)$$

where N is the number of matched default boxes, and the localization loss is the Smooth L1 loss [6] between the predicted box (l) and the ground truth box (g) parameters

We trained the models using an Adam optimizer with initial learning rate of 10^{-3} , 0.9 momentum, 0.0005 weight decay, batch size 8, and 150 number of epochs. Results of both supervised models will be elaborated in the evaluation section. The full training and testing code is available at https://github.com/..... .

Task 2: The main objective of this task is to classify the cropped smear images of detected WBC into four sub-types (Neutrophil, Monocyte, Lymphocyte, Eosinophil). To do so, we utilize eight CNN models: (1) customized architecture 1 (Figure 3); (2) customized architecture 2 (Figure 4); (3) pre-trained ResNet50 [5]; (4) pre-trained DenseNet121 [6]; (5) pre-trained VGG16 [15]; (6) pre-trained MobileNet-v2 [13];(7) pre-trained Xception [3]; and (8) pre-trained Inception [16]. In this study, we performed customized architectures for 20 epochs and pre-trained architectures for 10 epochs with learning rate 0.0001; Adam optimizer; and mini-batch size= 128 in the training set. The classification performance of all these eight networks are discussed in the evaluation section.

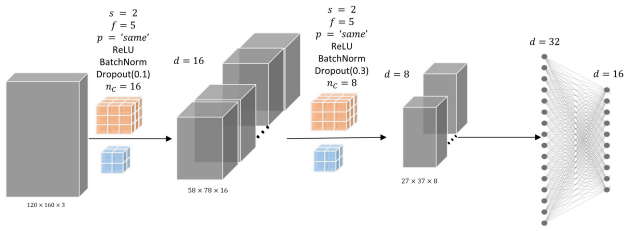

Figure 3. Graphical Representation of customized architecture 1

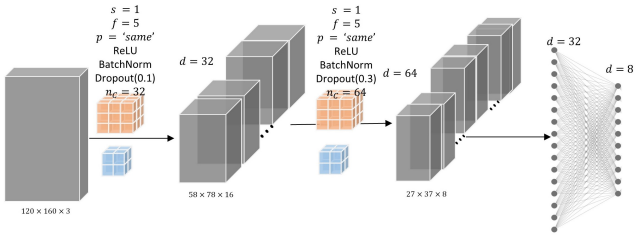

Figure 4. Graphical Representation of customized architecture 2

Task 3: The main objective of this task is to classify COVID-19 diagnosis based on the collected features collected from task 1 and task 2. The training of this task is based on the second data set in which the sample size is too small (201 samples). Therefore, we decided to perform this classification task by the classical machine learning classification models instead of deep learning models. To do so, we are using nine classifiers as follows: (1) Logistic Regression [12]; (2) K-Nearest Neighbor [9]; (3) Linear SVM [4]; (4) Non-Linear SVM [4]; (5) Gaussian Process [11]; (6) Decision Tree[8]; (7) Random Forest [2]; (8) Multi-Layer Perceptron (MLP) [10] with three layers containing (20,40,20) neurons, respectively; (9) AdaBoost [18]. The performance of all nine models are evaluated with details in the evaluation section.

5 Model Evaluation

As I explained before, in this project, we will perform three main tasks. I will explain about the computational experiments and results that we gained through each task in the following paragraphs.

Task 1: The dataset for this task includes images with three different types of objects (WBCs, RBCs, and platelets), and we split it into training and testing sets with ratio of 75%/25%. The performance of detection is evaluated using three measures: Precision, Recall and mean average precision(mAP). Table 2 shows the comparison between VGG16-based and Alexnet-based SSD. VGG-based SSD outperformed th Alexnet version for all three objects by 48% in mAP. Both models showed a relatively higher accuracy for detection of white blood cells while the precision of detection of platelets were as low as 15% and 74% for Alexnet-based and VGG16-based, respectively. To further evaluate the models performance, we use four image samples to detect the cells. As can be seen in Figure 5, Alexnet based model has a high false negative rate while the detection performance of VGG-based model was perfectly distinguish different type of cells.

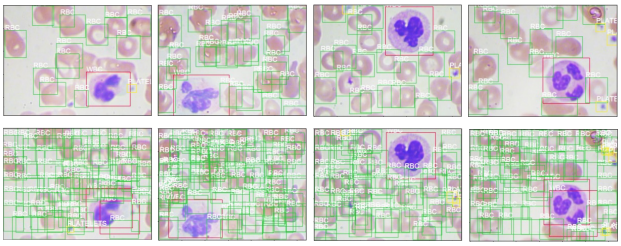

Figure 5. Detection examples on BCCD test dataset with VGG16(top) and Alexnet(bottom) based SSD models.

Table 2. Tasks1 Model Evaluation Results

		Alexnet-based SSD300		VGG16-based SSD300	
		Precision	Recall	Precision	Recall
Average Precision	WBC	0.89	0.84	0.99	1.0
	RBC	0.69	0.88	0.79	0.89
	Platelet	0.15	0.30	0.74	0.83
Mean Average Precision (mAP)		0.36		0.84	

Task 2: For this task, we split the balanced labeled data set to train/test with ratio 80%/20%. The size of images have been reduced from (540,960) to (120,160) to improve the training speed. Figure 6 represents the learning procedure of six pretrained models over epochs on the train and test set. Based on this figure, we can see that ResNet50 has the lowest performance in both training and validation over epochs (I know it's strange). On the other hand, it seems that DenseNet121 has the best performance (highest validation accuracy in the last epoch). The second and third best pretrained models seem to be MobileNet-v2 and VGG16.

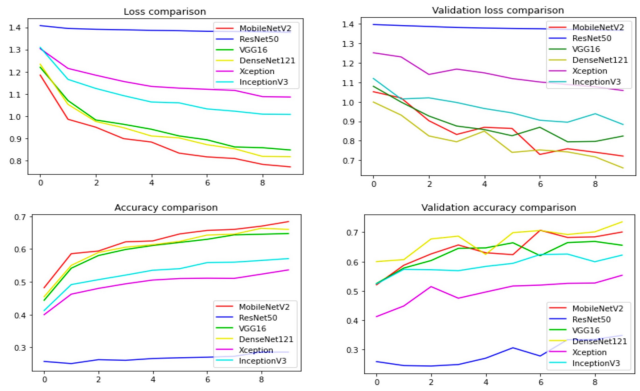

Figure 6. Pretrained models' improvement of loss and accuracy over epochs for train and test set

Figure 7 illustrates the learning procedure of customized architecture 1 and 2 over epochs. Based on this figure, we can claim that architecture 2 is performing way better compared to architecture 1 in the validation and training accuracy over epochs. It seems architecture 2 still has over-fitting. I tried different dropout and weight-decay techniques but couldn't improve it more than this.

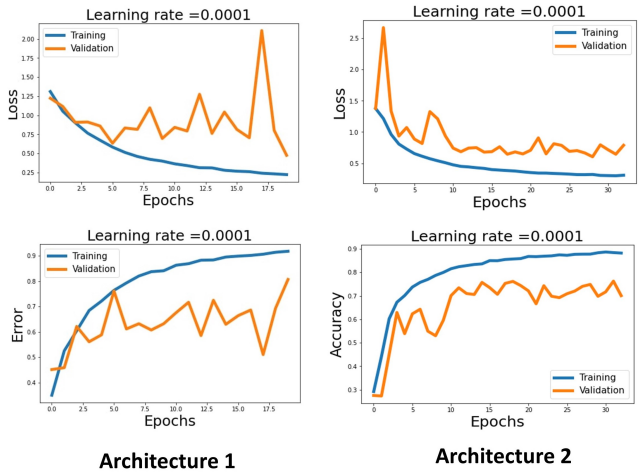

Figure 7. Customized models' improvement of loss and accuracy over epochs for train and test set

The classification performance of all these eight models are reported in Table 3. In this study, we are using accuracy score, precision, recall, and F-1 score as evaluation metrics. Based on Tables 3, we can see that pre-trained DenseNet121 and customized architecture 2 are performing better compared to others in terms all evaluation metrics. Among these top-2 models, DenseNet121 outperforms in terms of accuracy score, and recall, while architecture 2 outperforms in terms of precision. Moreover, architecture 2 has F-1 score 0.739 and DenseNet121 has F-1 score 0.744. So, it seems they have almost similar performance in this classification task.

It's very interesting since running time of architecture 2 is much faster compared to DenseNet121 and it can still reach to a performance as good as DenseNet121.

Table 3. Task2 Model Evaluation Results

	Accuracy Score	Precision	Recall	F-1 Score
Customized Arch1	0.58	0.52	0.58	0.55
Customized Arch2	0.76	0.72	0.76	0.739
Pretrained ResNet50	0.35	0.26	0.34	0.3
Pretrained DenseNet121	0.74	0.76	0.73	0.744
Pretrained VGG16	0.66	0.69	0.65	0.67
Pretrained MobileNet-v2	0.7	0.7	0.7	0.7
Pretrained Xception	0.55	0.56	0.55	0.55
Pretrained Inception	0.64	0.62	0.62	0.62

After finding the top-2 models DenseNet121 and Architecture 2, we decided to visualize the layer-1 feature maps that will be generated based on these top-2 models. Figure 8 and Figure 9 illustrate the layer-1 feature maps generated from DenseNet121 and architecture 2,respectively, for three randomly selected samples. From these figures, it can be observed that DenseNet121 is identifying the WBCs as important features, however, it also considers the black regions created around pictures after augmentation as important features in classification which is not true. In contrast, architecture 2 only consider the details of WBC as important features in layer-1. Therefore, it seems the architecture 2 is identifying important features of classification in layer 1 better than DenseNet121. So, our claim is true and this architecture seems to be more efficient.

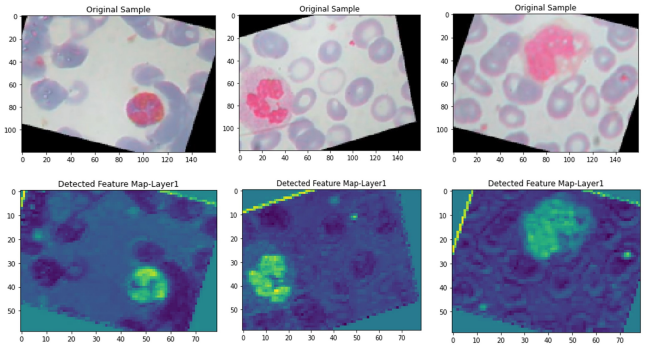

Figure 8. Layer-1 feature maps generated by pretrained DenseNet121

Figure 10 shows the confusion matrix of architecture 2 model on the test data set. based on the confusion matrix we can see that although the architecture 2 shows better performance in terms of efficiency and effectiveness compared to other models, it still gets confused in between two scenarios (*i*) Eosinophil and Neutrophil; and (*ii*) Monocyte and Neutrophil.

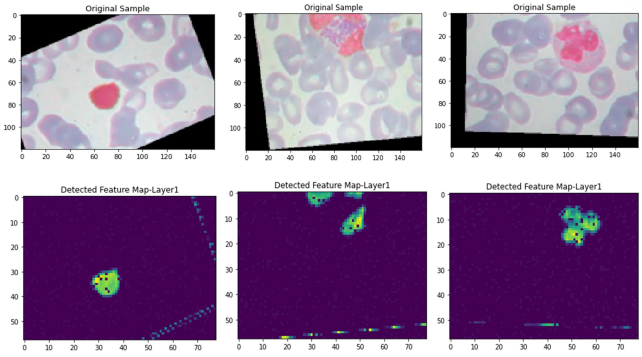

Figure 9. Layer-1 feature maps generated by customized architecture 2

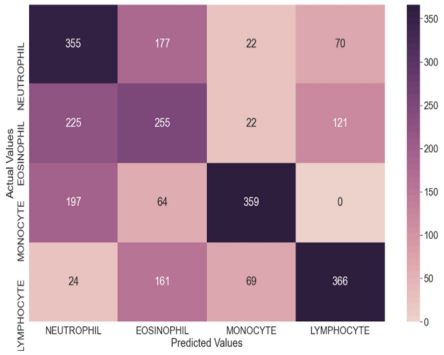

Figure 10. Confusion Matrix of Task 2 for Architecture 2

Task 3: For this task, we split the balanced labeled data set to train/test with ratio 75%/25%. wE will evaluate the classification performance of nine models mentioned in the model building section. The grid search cross validation is conducted on the test set to tune the hyperparameters of each mentioned classification method. Table 4 represents the testing classification performance of nine classical classification models on the COVID-19 diagnosis in task 3.

Table 4. Task3 Model Evaluation Results

	COVID-19	Accuracy score	Precision	Recall	F1-score	False negative (%)
Logistic Regression	negative	0.87	0.84	0.89	0.86	7.89
	positive		0.89	0.85	0.87	
10-Nearest Neighbors	negative	0.89	0.89	0.89	0.89	5.26
	positive		0.9	0.9	0.9	
Linear SVM	negative	0.87	0.84	0.89	0.86	7.89
	positive		0.89	0.86	0.87	
NonLinear SVM	negative	0.84	0.83	0.83	0.83	7.89
	positive		0.85	0.85	0.85	
Gaussian Process Classifier	negative	0.89	0.89	0.89	0.89	5.26
	positive		0.9	0.9	0.9	
Decision Tree	negative	0.82	0.79	0.83	0.81	10.53
	positive		0.84	0.82	0.8	
Random Forest	negative	0.76	0.74	0.768	0.76	13.16
	positive		0.79	0.75	0.77	
MLP	negative	0.87	0.84	0.89	0.86	7.89
	positive		0.89	0.85	0.87	
AdaBoost	negative	0.84	0.89	0.89	0.84	10.53
	positive		0.89	0.8	0.84	

Since in the COVID-19 classification, the false-negative has a high risk, therefore, a model that can create low false-negative rate or high recall will have a better performance.

Based on Tale 4, we can see that Gaussian Process and 10-Nearest Neighbors classifiers create lower false-negative rates. Figure 11 shows the confusion matrix of this binary classification for the Gaussian Process (GP) classifier on the test set. Based on this figure, we can also see the low false-negative rate of GP.

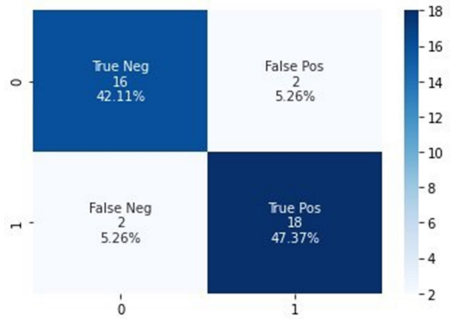

Figure 11. Confusion Matrix of Task 3 for Gaussian Process Classifier

6 Google Drive Link

Since the volume of the source code and data set was more than 5MB, we added them into a google drive folder. You can check the following google drive link. https://drive.google.com/drive/folders/1w8HQ8xBfPb-_8kfDLnLsMu7SpaqUTlm3?usp=sharing

References

[1] [n.d.]. Routine blood analysis greatly reduces the false-negative rate of RT-PCR testing for Covid-19. 91 ([n. d.]). https://doi.org/10.23750/abm.v91i3.9843

[2] Leo Breiman. 2001. Random Forests. *Machine Learning* 45, 1 (2001), 5–32. https://doi.org/10.1023/A:1010933404324

[3] Francois Chollet. 2017. Xception: Deep Learning With Depthwise Separable Convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[4] Corinna Cortes and Vladimir Vapnik. 1995. Support-vector networks. *Machine learning* 20, 3 (1995), 273–297.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep Residual Learning for Image Recognition. arXiv:1512.03385 [cs.CV]

[6] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2018. Densely Connected Convolutional Networks. arXiv:1608.06993 [cs.CV]

[7] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C Berg. 2016. Ssd: Single shot multibox detector. In *European conference on computer vision*. Springer, 21–37.

[8] Wei-Yin Loh. 2011. Classification and regression trees. *WIREs Data Mining and Knowledge Discovery* 1, 1 (2011), 14–23. https://doi.org/10.1002/widm.8 arXiv:https://onlinelibrary.wiley.com/doi/pdf/10.1002/widm.8

[9] Antonio Mucherino, Petraq J. Papajorgji, and Panos M. Pardalos. 2009. *k-Nearest Neighbor Classification*. Springer New York, New York, NY, 83–106. https://doi.org/10.1007/978-0-387-88615-2_4

[10] Allan Pinkus. 1999. Approximation theory of the MLP model in neural networks. *ACTA NUMERICA* 8 (1999), 143–195.

[11] Carl Edward Rasmussen. 2004. *Gaussian Processes in Machine Learning*. Springer Berlin Heidelberg, Berlin, Heidelberg, 63–71. https://doi.org/10.1007/978-3-540-28650-9_4

[12] Claude Sammut and Geoffrey I. Webb (Eds.). 2010. *Logistic Regression*. Springer US, Boston, MA, 631–631. https://doi.org/10.1007/978-0-387-30164-8_493

[13] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. MobileNetV2: Inverted Residuals and Linear Bottlenecks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

[14] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556* (2014).

[15] Karen Simonyan and Andrew Zisserman. 2015. Very Deep Convolutional Networks for Large-Scale Image Recognition. arXiv:1409.1556 [cs.CV]

[16] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alexander Alemi. 2017. Inception-v4, Inception-ResNet and the Impact of Residual Connections on Learning. 31 (Feb. 2017). https://ojs.aaai.org/index.php/AAAI/article/view/11231

[17] Jianwei Zhao, Minshu Zhang, Zhenghua Zhou, Jianjun Chu, and Feilong Cao. 2017. Automatic detection and classification of leukocytes using convolutional neural networks. *Medical & Biological Engineering & Computing* 55, 8 (01 Aug 2017), 1287–1301. https://doi.org/10.1007/s11517-016-1590-x

[18] Ji Zhu, Hui Zou, Saharon Rosset, and Trevor Hastie. 2009. Multi-class AdaBoost.