

# The detection and classification of acute myeloid leukaemia blood cell images based on different YOLO approaches

Kaung Myat Naing<sup>1</sup>, Veerayuth Kittichai<sup>2</sup>, Teerawat Tongloy<sup>1</sup>, Santhad Chuwongin<sup>1</sup>,  
Siridech Boonsang<sup>3</sup>

<sup>1</sup>College of Advanced Manufacturing Innovation, King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand

<sup>2</sup>Faculty of Medicine, King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand

<sup>3</sup>Department of Electrical Engineering, School of Engineering, King Mongkut's Institute of Technology Ladkrabang, Bangkok, Thailand

## Article Info

### Article history:

Received Jan 5, 2023

Revised Aug 3, 2023

Accepted Oct 11, 2023

### Keywords:

Acute myeloid leukaemia

Image classification

Model evaluations

Object detection

YOLO

## ABSTRACT

Medical image examination with a deep learning approach is greatly beneficial in the healthcare industry for faster diagnosis and disease monitoring. One of the popular deep learning algorithms such as you only look once (YOLO) developed for object detection is a successful state-of-the-art algorithm in real-time object detection systems. Although YOLO is continuously improving in the object detection area, there are still questions about how different YOLO versions compare in terms of performance. We utilize eight YOLO versions to classify acute myeloid leukaemia (AML) blood cells in image examinations. We also acquired the publicly available AML dataset from the cancer imaging archive (TCIA) which consists of expert-labeled single cell images. Data augmentation techniques are additionally applied to enhance and balance the training images in the dataset. The overall results indicated that eight types of YOLO approaches have outstanding performances of more than 90% in precision and sensitivity. In comparison, YOLOv4-tiny has a more reliable performance than the other seven approaches. Consistently, the YOLOv4-tiny also achieved the highest AUC score. Therefore, this work can potentially provide a beneficial digital rapid tool in the screening and evaluation of numerous haematological disorders.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



## Corresponding Author:

Siridech Boonsang

Department of Electrical Engineering, School of Engineering

King Mongkut's Institute of Technology Ladkrabang (KMITL)

1 Chalongkrung Soi 1, Lat Krabang District, Bangkok 10520, Thailand

Email: siridech.bo@kmitl.ac.th

## 1. INTRODUCTION

Acute myeloid leukaemia (AML) is one subtype of the four-leukaemia blast cell that generally originate in the bone marrow and human blood [1] and it can also be classified into distinctive categories by the type of mature and immature white blood cells (blast cells). Unlike any other type of cancer cell, the stages of leukaemia are differently determined based on blood cell counts and the accumulation of leukaemia cells within the organs [2]. Since the blast cell can be propagated and differentiated at a faster pace, the early-stage detection of leukaemia can save a lot of lives. Inaccurate diagnosis can have an adverse effect directly on the patient for further prescribing the drug regimen as well as increase the cost of the treatment and also could result in a complicated treatment procedure. The identification of the abnormal white blood cells causing the disease may provide important information to the physicians in the determination of the appropriate treatments [3]. A microscopic examination method is an early method for categorizing and

justifying the cause of the disease. Examination results could vary depending on sample preparation processes and also the experience and personnel training of examiners, which lead to inter-and intra-variation of the examining results [4], [5]. Although this process is being done by the medical-trained examiner who knows the interpretations of medical data, it is potentially misdiagnosed due to its subjectivity, and the complexity of blast cell characteristics of AML. It is also challenging to assign standardized images with time constraints. Therefore, the study of blast cell detection has significant importance in early screening of medical diagnosis for further effective planning of the prescription of the drug regimen.

The rapid growth of deep learning approaches in various real-world applications has led to emerging medical research utilizing large amounts of medical data, including electronic health records, texts, videos, sensor data, and medical imaging [6]. Deep learning methods, including convolutional neural networks (CNNs), have achieved remarkable success in medical image analysis [7]. Previous research has employed various image processing algorithms [8]–[12] and CNN-based classification models [13]–[16] for the segmentation and classification of leukocyte cells. However, there are still many challenging problems due to the usage of different equipment, data quality, data availability, limited resolution in noisy images, and so on. In recent years, CNN-based object detection approaches have gained attention in the field of leukocyte cell analysis. Wang *et al.* [17] introduced CNN-based object detection approaches, including single shot multibox detector (SSD) and you only look once version3 (YOLOv3), for recognizing 11 categories of peripheral leukocyte. Their work demonstrated promising classification results in this context. Additionally, Shakarami *et al.* [18] proposed fast and efficient YOLOv3 (FED) for the detection of blood cells. Their approach aimed to achieve fast and efficient performance in blood cell detection. These studies revealed the importance of leukocyte cell stage identification in predicting patient health and guiding treatment decisions. Inspired by this, we aim to investigate a relatively large dataset with more stage categories of leukocyte cells using a deep learning approach. Matek *et al.* [19] reported the recognition of blast cells in AML using a ResNeXt CNN classification model, achieving excellent results for classes with over 400 images but facing challenges with classes having fewer than 100 images. Balancing sample sizes for each class is critical for effective CNN model training.

Although several segmentation and classification models have been used in leukocyte cell analysis, there is still a research gap that can be filled by object detection-based models. Two well-known categories in object detection approaches are the region proposal-based two-stage detector and the regression/classification based one-stage detector. The two-stage detector offers better detection accuracy and adaptability, utilizing region of interest pooling (RoIPool) and region proposal network (RPN) for object bounding box classification and regression. However, it requires longer training and detection times. On the other hand, the one-stage detector provides high inference speed by directly predicting bounding boxes without the need for an RPN step. YOLO is a popular one-stage detection model, with each new version, including YOLOv7, showing improved detection accuracy [20]–[22]. Comparing different YOLO versions in terms of performance remains an ongoing question.

In this study, we propose to compare eight different versions of YOLO for the detection and classification of 15 classes of AML cell images. Data augmentation techniques are also employed to increase and balance the training images in the dataset. We also introduce the performance evaluation technique using the receiver operating characteristic (ROC) curve. This technique can help to examine the performance of our approaches with the quantitative AUC values. Section 2 will discuss the dataset arrangement, data augmentation techniques, and provide detailed explanations of the eight YOLO approaches. Section 3 will present a comprehensive discussion of the classification results and ROC analysis. Finally, in section 4 we will conclude our study, highlighting the potential of different YOLO approaches for the detection and classification of AML blood cell images.

## 2. METHOD

### 2.1. Dataset and labeling

The dataset for this study was received from the cancer imaging archive (TCIA). The peripheral blood smears were collected from 100 patients diagnosed with AML at munich University Hospital between 2014 and 2017. The munich AML morphology dataset contains 18,365 single-cell images and a trained examiner experienced in leukocyte cell classified 15 classes for training and evaluation. Each single-cell image has a size of 400×400 pixels (corresponding to approximately 29 μm×29 μm) including background components such as erythrocytes, platelets and cell fragments. The full single-cell image dataset and corresponding annotations are publicly available at TCIA [23].

The training and testing process is the most critical factor in machine learning models and there is no useful model for how to split the training and testing dataset [24]. The researchers examined the effect of the training and testing process on performance in machine learning by using various sampling theorems. The results showed that the test rate can be selected between 10% and 20% if the dataset is low or the number of samples in the dataset is low. In general, the test rate is between 20% and 50%. So, we randomly divide the received dataset into a training dataset and testing dataset for each class where the training dataset contains

about 80% and the remaining dataset is the testing dataset. To balance the number of images in the dataset, we need to reduce some classes in the training dataset which has a large number of images. So, the three classes which have over 3000 images, were reduced; neutrophil (segmented) was reduced to 1510 images and the other two classes namely lymphocyte (typical) and Myeloblast were reduced to 1000 images per class. The training dataset is reduced from 80% to 27% in the entire dataset. We also noticed that the five classes namely lymphocyte (atypical), promyelocyte (bilobed), metamyelocyte, monoblast, and smudge cell have lower than 30 images and the testing data set has only 2 and 3 images as shown in Table 1. To evaluate these four classes, the data sets are too small and difficult to get the actual performances.

Table 1. Data utilization and class-wise performance comparison between the previous work and the average of the eight YOLO models in terms of precision and sensitivity

Class	Abb	No. of images		Testing 20%	Precision		Sensitivity	
		Training [19]	YOLOs		ResNeXt [19]	YOLOs	ResNeXt [19]	YOLOs
Neutrophil (segmented)	NGS	6788	1510	1696	0.99±0.00	0.99±0.00	0.96±0.01	0.97±0.01
Neutrophil (band)	NGB	88	88	21	0.25±0.03	0.32±0.13	0.59±0.16	0.43±0.15
Lymphocyte (typical)	LYT	3150	1000	787	0.96±0.01	0.97±0.03	0.95±0.02	0.93±0.03
Lymphocyte (atypical)	LYA	9	9	2	0.20±0.40	0.03±0.05	0.07±0.13	0.13±0.23
Monocyte	MON	1432	700	357	0.90±0.04	0.84±0.04	0.90±0.05	0.88±0.05
Eosinophil	EOS	340	340	84	0.95±0.04	0.98±0.01	0.95±0.01	0.95±0.02
Basophil	BAS	64	64	15	0.48±0.16	0.70±0.11	0.82±0.07	0.65±0.14
Myeloblast	MYO	2615	1000	653	0.94±0.01	0.94±0.02	0.94±0.02	0.89±0.09
Promyelocyte	PMO	57	57	13	0.63±0.16	0.53±0.14	0.54±0.20	0.68±0.10
Promyelocyte (bilobed)	PMB	15	15	3	0.45±0.32	0.18±0.24	0.41±0.37	0.50±0.40
Myelocyte	MYB	34	34	8	0.46±0.19	0.42±0.14	0.43±0.07	0.52±0.21
Metamyelocyte	MMZ	12	12	3	0.07±0.13	0.41±0.43	0.13±0.27	0.25±0.24
Monoblast	MOB	21	21	5	0.52±0.30	0.22±0.02	0.58±0.26	0.35±0.28
Erythroblast	EBO	64	64	14	0.75±0.20	0.65±0.20	0.87±0.09	0.94±0.06
Smudge cell	KSC	13	12	2	0.53±0.28	0.58±0.42	0.77±0.20	0.81±0.37
Total		14701	7705	3663				

Image labeling is the process of manually labeling the regions of an object in an image and creating text-based descriptions of those regions for object classification. The description of the image labeling was performed by well-trained personals. The main objective of image labeling is to allow the users to highlight or specify the pixel area of the objects in an image. The CiRA CORE platform [25] was used as a tool for image labeling before continuing with data augmentation techniques.

## 2.2. Data augmentation

Data augmentation is one of the most widely used techniques for increasing the sample size of a dataset for the model training process. Many researchers used numerous image transformation techniques, such as position augmentation and color augmentation, to obtain various types of original images. Since deep learning models were trained with both the original and various augmentation images, they have more generalization capabilities. As a survey result on image data augmentation for deep learning, various data augmentation techniques have been developed to lessen the overfitting of the learning model by providing better generalization [26]. Therefore, we used the four data augmentation techniques: i) rotation (rotating the image in 45° intervals between -180° and 180° to produce a variety of images), ii) contrast (adjusting between the darkest and brightest image portions by multiplying all pixel values with 0.4, 0.6, 0.8, and 1.0), iii) noise (image noise injection by using a gaussian noise distribution with three standard deviations ( $\sigma=0, 10, 20$ )), and iv) blur (blurring the image by using gaussian filter with a standard deviation of 9). After performing the above four image processing tasks, an individual image can be augmented up to 108 images with a size of 608x608 pixels. The training dataset with a 608x608 pixels image was further prepared for the training dataset of eight versions of the YOLO algorithm.

## 2.3. Detection of AML blood cell images based on different YOLO approaches

The YOLO object detection models, including their tiny versions, have been utilized for the detection of AML blood cell images. These models consist of YOLOv2, YOLOv2-tiny, YOLOv3, YOLOv3-tiny, YOLOv4, YOLOv4-tiny, YOLOv7, and YOLOv7-tiny. Each model has distinct hyperparameters (Table 2) and characteristics. The tiny versions have the advantage of being able to run on a CPU instead of a GPU. The original YOLO model, proposed by Redmon *et al.* [27] introduced an end-to-end network that combines feature extraction, candidate frame classification, and regression. While YOLOv1 significantly improved the detection rate compared to two-stage methods, it had reduced detection accuracy. However, subsequent modifications to the YOLO model have significantly enhanced object detection performance.

Table 2. The hyperparameters of various YOLO models in model training

Parameters	v2	v3	v4	v7	v2-tiny	v3-tiny	v4-tiny	v7-tiny
Input image size	608×608	608×608	608×608	640×640	416×416	416×416	416×416	416×416
Batch size	64	64	64	64	64	64	64	64
Subdivision	16	16	16	16	16	16	16	16
Channels	3	3	3	3	3	3	3	3
Momentum	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9
Weight decay	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005	0.0005
Base learning rate	0.001	0.001	0.001	0.00261	0.001	0.001	0.00261	0.00261
Learning rate scales	[0.1, 0.1]	[0.1, 0.1]	[0.1, 0.1]	[0.1, 0.1]	[0.1, 0.1]	[0.1, 0.1]	[0.1, 0.1]	[0.1, 0.1]
Activation function	leaky ReLU, linear	leaky ReLU, linear	mish, leaky, linear	SiLU, logistic	leaky ReLU, linear	leaky ReLU, linear	leaky ReLU, linear	leaky ReLU, logistic

YOLOv2, introduced by Redmon and Farhadi in 2017 [28], utilized the DarkNet19 feature extraction network and incorporated a batch normalization layer for faster network convergence. It also employed a k-means clustering algorithm to automatically determine prior anchor boxes, resulting in improved detection performance. YOLOv2 outperformed other detection systems at the time and required less processing power due to its 19 convolutional layers and 5 max-pooling layers. The model produced one output feature map of size 19×19 for object prediction with five anchor boxes. YOLOv2-tiny is a smaller version of YOLOv2 with 9 convolution layers and 6 max-pooling layers for feature extraction. It predicts one output feature map using the same five anchor boxes as YOLOv2. The trained model provides an output feature map of size 13×13 for object prediction.

YOLOv3, known for its speed and accuracy, introduced the Darknet-53 backbone for feature extraction [29]. It incorporated ResNet shortcut connections to address gradient disappearance and additional convolutional layers for predicting three different bounding boxes. YOLOv3 adopted the sum of squared error loss and logistic regression function for bounding box predictions. It utilized independent logistic classifiers and binary cross-entropy loss for multilabel class predictions. The model employed k-means clustering to determine bounding box priors and produced three-branch outputs with nine anchor boxes in the feature map for each cell image, with different sizes such as 19×19 image size for a large object, 38×38 image size for medium object and 76×76 image size for a small object, respectively. YOLOv3-tiny, a simplified version, offered faster processing and reduced memory requirements. It shared similarities with YOLOv2-tiny, featuring 9 convolution layers and 6 max-pooling layers for feature extraction. The target was distributed into two scales using k-means clustering. YOLOv3-tiny produced two branch outputs with six anchor boxes for prediction, corresponding to feature maps of size 13×13 and 26×26.

YOLOv4, proposed by Bochkovskiy *et al.* [30], improved processing speed and detection accuracy, but it still needs to be improved in terms of efficiency. Its structure comprised the CSP Darknet53 backbone network for feature extraction, a neck network incorporating spatial pyramid pooling (SPP) and path aggregation network (PANet) for feature fusion, and a head network for output prediction. YOLOv4 introduced the mish activation function and utilized the CSP block module for improved learning ability. It employed the complete intersection over union (CIoU) loss function and the distance IoU (DIoU) non-maximum suppression algorithm. The model's final output prediction used the same three output feature maps as YOLOv3. Because the YOLOv4-tiny is based on YOLOv4, the feature extractor is the CSPDarknet53-tiny backbone. The YOLOv4-tiny CSP block, in contrast to YOLOv4, uses the leaky rectified linear unit (ReLU) activation function rather than the Mish activation function. The YOLOv4-tiny also produces two-branch outputs using two distinct scales of the feature map for the output prediction, similar to the YOLOv3-tiny.

Despite the fact that YOLOv5 and YOLOv6 were launched in 2020 and 2022, respectively, YOLOv7 claims to be the fastest and most accurate real-time object detector to date [31]. In fact, the YOLOv7 model preprocessing approach is associated with YOLOv5, and the extended efficient layer aggregation network (E-ELAN) is proposed as the network's backbone to improve self-learning ability while retaining the original gradient path. Unlike YOLOv5, YOLOv7 integrates the head and neck networks into a single head network, but the functionalities remain the same. The head network consists of five sections including spatial pyramid pooling cross-stage partial connection (SPCSPC), a series of CBS (used for convolution, normalization, and activation function), MP (composed of MaxPool and CBS), Catconv (concatenation, convolution), and Repconv (re-parameterized convnet structure). From this structure, the model preprocesses the input image and resizes it to 640×640 sizes before feeding it into the backbone and head network. The model then outputs three layers of different size feature maps for the detection result of the image, such as 20×20 image size for a large object, 40×40 image size for a medium object, and 80×80 image size for a small object, respectively. YOLOv7-tiny, a compressed version designed for edge GPU, employed a leaky ReLU activation function, and generated three branch outputs with nine anchor boxes. The

feature map sizes were 13×13, 26×26, and 52×52, matching those of YOLOv7. Each YOLO model predicted class probabilities and bounding box coordinates using thresholding and non-maximum suppression (NMS).

In operation, the YOLO models divide images into grid cells and predict bounding boxes and class probabilities for objects within each cell. The YOLO method extracts significant characteristics from images using a CNN as its backbone network. These features are then fed into a series of convolutional and fully connected layers, which predict the bounding boxes and class labels. YOLO predicts objects of different sizes and aspect ratios using a single unifying model, making it efficient and capable of real-time object detection. YOLO also uses anchor boxes to handle object scale and location variations. The method predicts multiple bounding boxes at the same time and assigns a confidence score to each box, representing its probability of containing an object. NMS is applied to remove redundant bounding boxes and produce the final set of object detections.

#### 2.4. Statistical analysis

The four models' performance is evaluated using a testing dataset containing 15 classes of AML blood cell images. The performance of the eight models is evaluated using a testing dataset that includes images of 15 different classes of AML blood cells. When the images from each class are tested, the prediction results will be obtained with a confidence level. The number of prediction results for each class is counted and summarized in a 15×15 confusion matrix table. The four statistical parameters namely precision, sensitivity, specificity, and accuracy are considered performance metrics as (1) to (4) [32]:

$$Precision = \frac{TP}{TP+FP} \quad (1)$$

$$Sensitivity = \frac{TP}{TP+FN} \quad (2)$$

$$Specificity = 1 - FPR = \frac{TN}{TN+FP} \quad (3)$$

$$Accuracy = \frac{TP+TN}{TP+FP+TN+FN} \quad (4)$$

Besides, we can visualize the model performances and evaluate the classifiers with the well-known technique, the ROC curve [33]. ROC curve is a two-dimensional graph in which true positive rate (TPR) and false positive rate (FPR) values are plotted on the Y-axis and X-axis, respectively. TPR and FPR values can be received from the confusion matrix table to form an ROC curve. TPR value is the same as sensitivity and FPR value can be calculated as (5):

$$FPR = 1 - Specificity = \frac{FP}{FP+TN} \quad (5)$$

Since the ROC is constructed by changing the threshold level on the prediction score, each changing threshold generates one point in the ROC curve [34]. In this study, we constructed the ROC curve at the threshold level with every 5% increment because our testing dataset has over 3000 images. The above descriptions of true positive (TP), false positive (FP), true negative (TN), and false negative (FN) are simply used in the binary class decision. However, we use the multi-class classification with a confusion matrix of 15×15 for blood cell image analysis. In the multi-class classification, the four types of conditions could be computed by using a one-versus-rest approach. Then, the performance calculations can be achieved with two operations, namely micro-averaging and macro-averaging. Because of the imbalance in the class dataset, we used micro-averaging in this study [35]. To plot the ROC curve, the required TPR and FPR are computed as (6), (7):

$$TPR = \frac{TTP}{TTP+TFN} = \frac{\sum_{i=1}^n TP_i}{\sum_{i=1}^n TP_i + FN_i} \quad (6)$$

$$FPR = \frac{TFP}{TFP+TTN} = \frac{\sum_{i=1}^n FP_i}{\sum_{i=1}^n FP_i + TN_i} \quad (7)$$

where  $i$  stands for each testing class and  $n$  will be 15 classes for this study. If  $i=1$ , the 1st class is considered as positive and the rest of the 14 classes as negative class. In the formulas,  $TP_i$ ,  $FP_i$ ,  $TN_i$ , and  $FN_i$  are associated with each testing class  $i$ , and  $TTP$ ,  $TFP$ ,  $TTN$ , and  $TFN$  are the total of each condition for all classes.

In addition to evaluating the models based on the ROC curve, we calculated the area under the curve (AUC) as a measure of their usefulness. AUC represents the integration of the area under the ROC curve and provides valuable insights into the model's performance. A higher AUC value indicates a more effective model in testing. Alongside the TPR, which is equivalent to sensitivity, we also computed three other performance metrics for all classes as (8) to (10):

$$Precision = \frac{\sum_{i=1}^n TP_i}{\sum_{i=1}^n TP_i + FP_i} \quad (8)$$

$$Specificity = \frac{\sum_{i=1}^n TN_i}{\sum_{i=1}^n TN_i + FP_i} \quad (9)$$

$$Accuracy = \frac{\sum_{i=1}^n TP_i + TN_i}{\sum_{i=1}^n TP_i + FP_i + TN_i + FN_i} \quad (10)$$

### 3. RESULTS AND DISCUSSION

Since YOLO algorithms require a large amount of data, we need to use a machine that has very high computing power. All of the four algorithms were configured on the operation system of the Ubuntu 16.04 LTS (64-bit) and the hardware environment as follows: processor: Intel® Core i5-8400, CPU @ 2.8GHz\*6, memory: 31.3 GiB, and graphics: GeForce GTX 1070 Ti. In this study, each training process took a maximum of four days for computing time.

#### 3.1. Class-wise performance comparison on the eight YOLO models

This section describes the performance comparison for the eight YOLO models with the testing dataset. We evaluated the prediction output scores for each class by using the corresponding trained models. The parameter selection of threshold level: 0.5 (50%) and NMS value: 0.2 are used as default to attain the prediction scores. Since there is a 0.5 classification threshold, the models can predict the classes if the probability is greater than 0.5. Besides, we utilized the NMS technique to reduce duplication because the model can produce duplicate detections for the same output [36]. The class predictions were then achieved using the testing dataset, which contains 15 classes of blood cell images with a total of 3,663 single-cell images. The confusion matrix table was created by using the actual class label and the predicted class score. To complete the confusion matrix, we needed to count the predicted class for each image test and fill in the received counted number on the prediction class of the matrix. The eight types of confusion matrix tables for the eight YOLO algorithms are accessible at data availability.

During the testing of the trained models, we observed that YOLOv2, YOLOv3-tiny, and YOLOv7 demonstrated the ability to correctly predict more classes compared to other algorithms. Specifically, these three models can predict 14 classes accurately, while YOLOv2-tiny, YOLOv4, YOLOv4-tiny, and YOLOv7-tiny can predict 13 classes, and YOLOv3 can predict 12 classes, despite some classes having a small dataset. Figure 1 illustrates the typical results obtained from the detection and classification of each single-cell image.

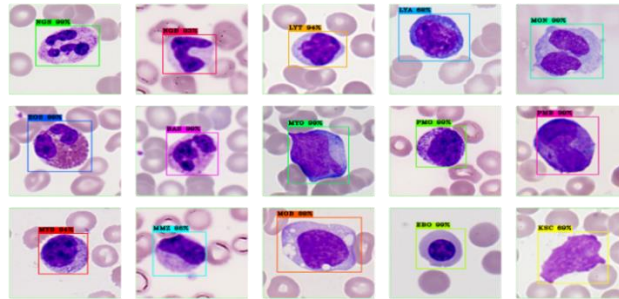


Figure 1. The classification and localization of AML blood cell images with a threshold level of 0.5 and NMS of 0.2

As mentioned in the dataset and labeling section, the eight models were difficult to correctly perform in the small testing dataset. Nevertheless, the trained models accurately predicted the class namely Smudge cell because it has a distinctive character from other classes. YOLOv4-tiny predicted fewer classes than YOLOv2, YOLOv3-tiny, and YOLOv7, but it has more prediction images than the other models. For the total number of image predictions, the eight models correctly predicted over 3663 single-cell images in descending order as follows: YOLOv4-tiny predicted 3455 images, YOLOv3 predicted 3444 images, YOLOv3-tiny predicted 3430 images, YOLOv7 predicted 3419 images, YOLOv2 predicted 3998 images, YOLOv4 predicted 3383 images, YOLOv7-tiny predicted 3382 images and YOLOv2-tiny predicted 3297 images, respectively.

Additionally, apart from the small five classes, we compare the precision and sensitivity values to evaluate the quality of class-wise prediction using the one-versus-rest approach, as shown in Figures 2 and 3. At this point, we used the results of the previous work [19] to make the comparison which used the ResNeXt CNN approach for training and 5-fold cross-validation for testing. Consequently, they performed five different times for training and testing. Although they presented their interval values in precision and sensitivity, we only used the average results for comparison.

Figure 2 presents the precision results for the previous work and eight types of YOLO with the bar graph. In general, we found that all models except YOLOv2-tiny have at least 90% precision in the three large dataset classes (neutrophil (segmented) (NGS), lymphocyte (typical) (LYT) and myeloblast (MYO)). Among the medium dataset classes, neutrophil (band) (NGB) has the lowest precision value because its biological pattern is comparable to NGS. Eosinophil (EOS) cells, on the other hand, have a high precision (at least 92%) for all models due to their unique characteristics, as illustrated in Figure 1.

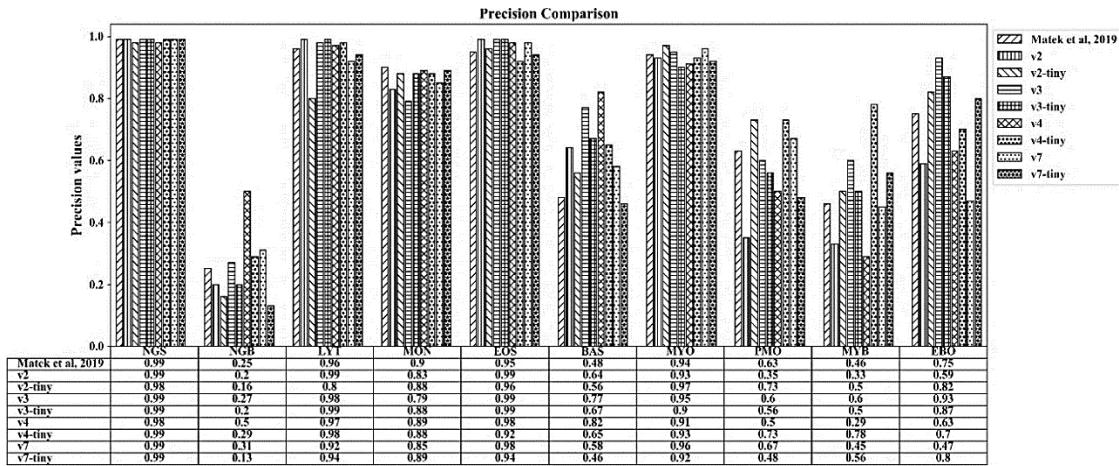


Figure 2. The comparison of precision values for the eight types of YOLO and the previous work

Figure 3 displays the sensitivity results for all of the models. The best sensitivity scores were also obtained at least 86% in the three classes as described in the precision result in all models except YOLOv2-tiny. The significantly characterized class, namely EOS, has a relatively high sensitivity score of at least 92%. To summarize the above facts, the large training datasets still provided the capability of YOLO in the classification of blood cell images. Besides, the performance of multi-classification dropped due to the similar characteristics of blood cells for a small dataset.

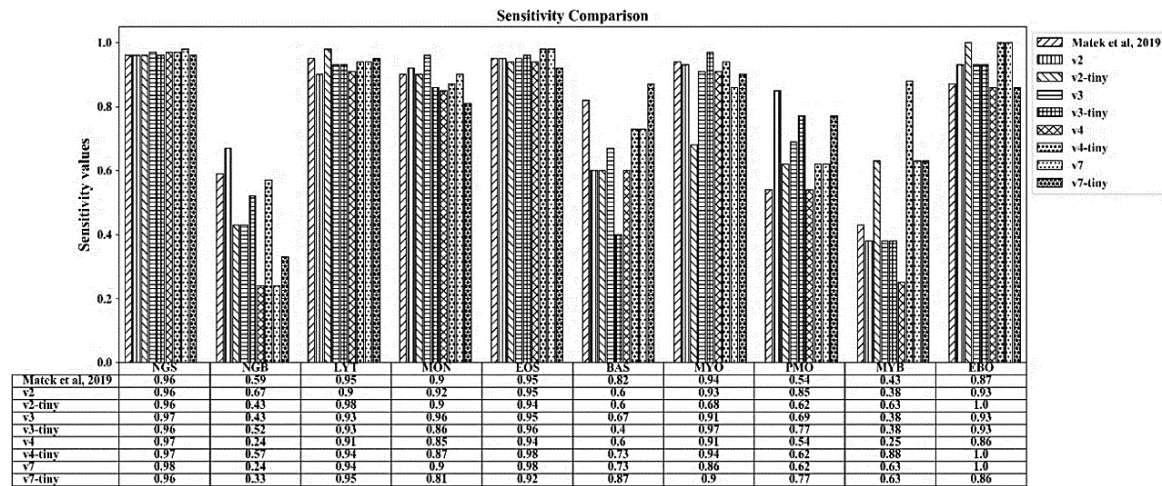


Figure 3. The comparison of sensitivity values for the eight types of YOLO and the previous work

The overall results of four-performance metrics for eight types of the YOLO model are shown in Figure 4. YOLOv4-tiny has 94% in overall precision and sensitivity and 99% in overall specificity and accuracy, which is a better performance than the other comparative YOLO models. As a result, we can conclude that the YOLOv4-tiny model is the most suitable one for detecting AML blood cells during image classification. Furthermore, YOLOv4-tiny is also compatible with edge GPU and CPU devices because it consumes less memory.



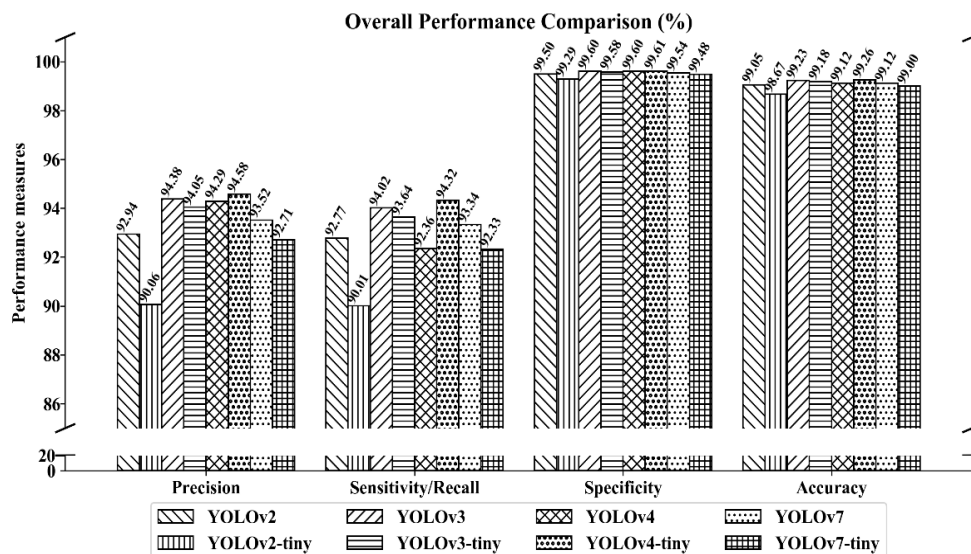


Figure 4. The overall score for the four-performance metrics namely precision, sensitivity, specificity, and accuracy with a threshold level of 0.5, and NMS of 0.2, the model performances are calculated by using a micro-averaging approach

In addition, Table 1 presents the average precision and sensitivity scores of eight YOLO models, comparing them with previous work [19]. YOLO models perform well in common cell classes (NGS, LYT, MON, EOS, MYO), but some classes remain challenging. Despite using only 50% of the training dataset, YOLO models' performance is comparable to the previous work with the same test dataset size, showing better precision in 6 classes and better sensitivity in 8 classes.

### 3.2. ROC curve and the performance analysis on the varied of threshold levels for the eight YOLO models

For performance analysis using a ROC curve, eight types of YOLO models are employed as classifiers to provide only a class decision, i.e., true class or false class on each instance. After applying these classifiers to a test dataset, it yields a confusion matrix for each threshold value that corresponds to one point in the ROC curve.

The classification results logically yield a numeric value of an instance probability with the predicted classes as shown in Figure 1. The classifier produces the predicted classes if its output is higher than the predefined threshold values. From these results, we collected the overall values of four conditions such as TTP, TFP, TTN, and TFN scores of 15 classes from a confusion matrix. After that, we calculated the two important values (TPR and FPR) to plot a single point in the ROC curve. In this way, there are many different points in the ROC curve by varying the threshold values. Conceptually, we alter the threshold values from 0 (0%) to 1 (100%) to complete the ROC curve. Next, we performed the classification by using the testing dataset with the threshold values for each 0.05 (5%) increment. Consequently, we obtained the different 21 corresponding points in the ROC curve. The summarization of the classification results for eight types of YOLO approaches to plot the ROC curve is accessible at data availability.

Considering the TPR and FPR results, we used the Jupyter Notebook open-source web application with a Python environment to plot the ROC curve as shown in Figure 5. In the ROC curve analysis, the AUC is an effective way to evaluate the performance of the trained model. The AUC value is always bounded between 0 and 1, where a perfectly inaccurate test represents a value of 0 and a perfectly accurate test represents a value of 1. In general, an AUC value can be defined as follows: under 0.5 is no realistic model, 0.5 is no discrimination model, 0.7 to 0.8 is considered an acceptable model, 0.8 to 0.9 is considered an excellent model, and more than 0.9 is considered an outstanding model [37].

Based on the findings presented in Figure 5, we can conclude that the YOLOv4-tiny model is a detection model, as it achieved the highest AUC value. The AUC values for the eight YOLO models are as follows: 0.963 for YOLOv2, 0.948 for YOLOv2-tiny, 0.969 for YOLOv3, 0.967 for YOLOv3-tiny, 0.964 for YOLOv4, 0.971 for YOLOv4-tiny, 0.966 for YOLOv7, and 0.961 for YOLOv7-tiny. Since all the AUC values are higher than 0.9, it is evident that each version of the YOLO model can be considered an outstanding model for AML cell classifications. However, the YOLOv4-tiny model stands out with the highest AUC value among them.



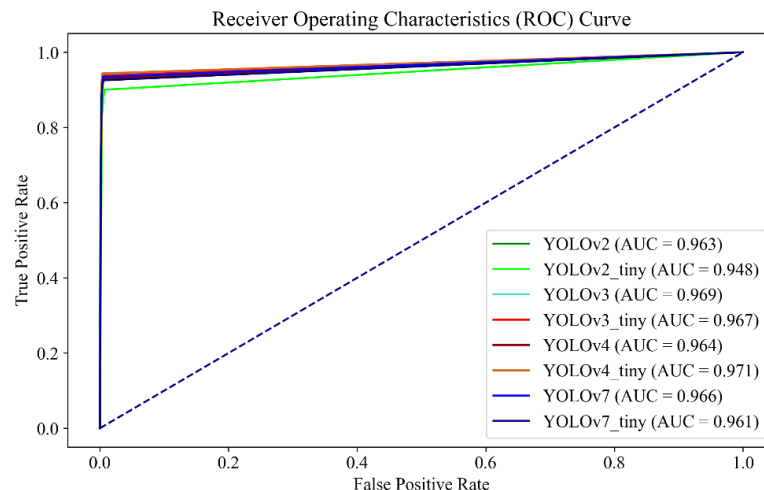


Figure 5. The ROC curve for eight types of YOLO approaches after evaluating the testing dataset

We present a summary of our findings regarding the overall performance comparison between eight types of YOLO approaches, as shown in Table 3. While we varied the threshold values for testing, we kept the NMS unchanged for all data analyses. Overall, we observed that the eight different YOLO models demonstrate high effectiveness based on their performance scores. From the analysis of Table 3, we observed variations in the four performance values as we adjusted the threshold values. The results indicate that higher threshold values led to greater precision and specificity scores in the models' predictions.

Table 3. Overall performance comparison between the eight types of YOLO approach for each 10% threshold value increment

Performances	Model	Threshold								
		10%	20%	30%	40%	50%	60%	70%	80%	90%
Precision	YOLOv2	0.9282	0.9282	0.9282	0.9285	0.9294	0.9361	0.9438	0.9590	0.9835
	YOLOv3	0.9421	0.9421	0.9426	0.9431	0.9438	0.9438	0.9462	0.9498	0.9577
	YOLOv4	0.9390	0.9394	0.9398	0.9411	0.9429	0.9445	0.9479	0.9534	0.9607
	YOLOv7	0.9347	0.9347	0.9350	0.9350	0.9352	0.9368	0.9415	0.9481	0.9597
	YOLOv2-tiny	0.9006	0.9006	0.9006	0.9006	0.9006	0.9017	0.9048	0.9143	0.9306
	YOLOv3-tiny	0.9380	0.9380	0.9387	0.9392	0.9405	0.9417	0.9437	0.9486	0.9550
	YOLOv4-tiny	0.9438	0.9440	0.9440	0.9451	0.9458	0.9470	0.9505	0.9557	0.9627
	YOLOv7-tiny	0.9257	0.9257	0.9256	0.9259	0.9271	0.9303	0.9376	0.9510	0.9634
Sensitivity	YOLOv2	0.9282	0.9282	0.9282	0.9282	0.9277	0.9195	0.9031	0.8676	0.6653
	YOLOv3	0.9413	0.9413	0.9408	0.9405	0.9402	0.9391	0.9358	0.9293	0.9203
	YOLOv4	0.9293	0.9268	0.9252	0.9244	0.9236	0.9203	0.9137	0.9042	0.8886
	YOLOv7	0.9345	0.9345	0.9342	0.9339	0.9334	0.9304	0.9233	0.9129	0.8832
	YOLOv2-tiny	0.9001	0.9001	0.9001	0.9001	0.9001	0.8990	0.8949	0.8851	0.8572
	YOLOv3-tiny	0.9375	0.9369	0.9369	0.9369	0.9364	0.9348	0.9328	0.9274	0.9165
	YOLOv4-tiny	0.9438	0.9438	0.9438	0.9438	0.9432	0.9421	0.9391	0.9312	0.9162
	YOLOv7-tiny	0.9249	0.9249	0.9238	0.9238	0.9233	0.9189	0.9113	0.8998	0.8613
Specificity	YOLOv2	0.9949	0.9949	0.9949	0.9949	0.9950	0.9955	0.9962	0.9973	0.9992
	YOLOv3	0.9959	0.9959	0.9959	0.9959	0.9960	0.9960	0.9962	0.9965	0.9971
	YOLOv4	0.9957	0.9957	0.9958	0.9959	0.9960	0.9961	0.9964	0.9968	0.9974
	YOLOv7	0.9953	0.9953	0.9954	0.9954	0.9954	0.9955	0.9959	0.9964	0.9973
	YOLOv2-tiny	0.9929	0.9929	0.9929	0.9929	0.9929	0.9930	0.9933	0.9941	0.9954
	YOLOv3-tiny	0.9956	0.9956	0.9956	0.9957	0.9958	0.9959	0.9960	0.9964	0.9969
	YOLOv4-tiny	0.9960	0.9960	0.9960	0.9961	0.9961	0.9962	0.9965	0.9969	0.9975
	YOLOv7-tiny	0.9947	0.9947	0.9947	0.9947	0.9948	0.9951	0.9957	0.9967	0.9977
Accuracy	YOLOv2	0.9904	0.9904	0.9904	0.9904	0.9905	0.9904	0.9900	0.9887	0.9769
	YOLOv3	0.9922	0.9922	0.9922	0.9922	0.9923	0.9922	0.9922	0.9920	0.9920
	YOLOv4	0.9913	0.9911	0.9911	0.9911	0.9912	0.9911	0.9909	0.9907	0.9902
	YOLOv7	0.9913	0.9913	0.9913	0.9913	0.9912	0.9912	0.9911	0.9909	0.9902
	YOLOv2-tiny	0.9867	0.9867	0.9867	0.9867	0.9867	0.9867	0.9867	0.9868	0.9862
	YOLOv3-tiny	0.9917	0.9917	0.9917	0.9918	0.9918	0.9918	0.9918	0.9918	0.9916
	YOLOv4-tiny	0.9925	0.9925	0.9925	0.9926	0.9926	0.9926	0.9927	0.9925	0.9902
	YOLOv7-tiny	0.9900	0.9900	0.9900	0.9900	0.9900	0.9900	0.9900	0.9902	0.9902

In addition, Table 4 presents a comparison of our results with other state-of-the-art approaches. Dasariraju *et al.* [10] achieved an overall accuracy of 92.99% using the random forest algorithm. Rastogi *et al.* [12] utilized LeuFeatx features with extra trees classifier, achieving an overall accuracy of 96.15%. In paper [15] and [38] used two-stage hybrid model and ghost-ResNeXt, achieving overall accuracies of 97.00% and 98.61%, respectively. In this paper, we propose to use the different YOLO models for blood cell detection and classification. The analysis of Table 4 indicates that our proposed YOLOv4-tiny model outperforms other models in terms of accuracy.

Table 4. Comparison with existing state-of-the-art models

Studies	Dataset	Technique	Overall accuracy (%)
Dasariraju <i>et al.</i> , 2020 [10]	Single cell AML dataset	Random forest algorithm	92.99
Rastogi <i>et al.</i> , 2022 [12]		LeuFeatx+ETC	96.15
Elhassan <i>et al.</i> , 2023 [15]		Two-stage hybrid model	97.00
Bairaboina and Battula, 2023 [38]		Ghost-ResNeXt	98.61
This study		YOLOv4-tiny	99.26

#### 4. CONCLUSION

We introduce the four main types of the well-known object detection strategy, the YOLO approach to detect and classify the 15-class of WBC cell images. To enhance the training dataset, we employed data augmentation techniques, including rotation, contrast adjustment, noise addition, and blur, resulting in one image being augmented into up to 108 images. The training dataset, enriched with these augmented images of 608x608 pixels, was used to assess the performance of eight YOLO models (YOLOv2, YOLOv2-tiny, YOLOv3, YOLOv3-tiny, YOLOv4, YOLOv4-tiny, YOLOv7, and YOLOv7-tiny) for the classification of AML blood cell images. Among these models, YOLOv4-tiny demonstrated superior performance, achieving over 94% in overall precision and sensitivity, and over 99% in overall specificity and accuracy. Furthermore, we proposed a performance evaluation procedure using the ROC curve, allowing for a quantitative examination of the approaches through AUC values. The AUC values for the eight YOLO models were found to be notably high, ranging from 0.948 to 0.971. With all obtained AUC scores exceeding 0.9, these eight YOLO models are deemed outstanding for AML cell classifications. Considering the remarkable performance achieved, our approach utilizing a one-stage object detection model holds great potential for enabling more reliable and faster clinical diagnoses in the future, thus contributing to advancements in the field of healthcare and medical applications.

#### DATA AVAILABILITY

All trained models with weight files, confusion matrices, summarization of the classification results for eight types of YOLO approach to plot ROC curve, sample images of four augmentation techniques, and some test images can be accessed via this link: <https://doi.org/10.6084/m9.figshare.21762620.v1>.

#### ACKNOWLEDGEMENTS

This work was financially supported by (Research grant for New Scholar, Grant No. RGNS 65 - 212) from the Office of the Permanent Secretary, Ministry of Higher Education, Science, Research, and Innovation (OPS MHESI), and Thailand Science Research and Innovation (TSRI). This work was also supported by King Mongkut's Institute of Technology Ladkrabang (KMITL), Thailand. We acknowledge with thanks The Cancer Imaging Archive (TCIA) for their publicly available dataset.




#### REFERENCES

- [1] N. Ahmed, A. Yigit, Z. Isik, and A. Alpkocak, "Identification of leukemia subtypes from microscopic images using convolutional neural network," *Diagnostics*, vol. 9, no. 3, p. 104, 2019, doi: 10.3390/diagnostics9030104.
- [2] V. L. Katz, G. Lentz, R. A. Lobo, D. Gershenson, and M. Willard, "Small Animal Clinical Diagnosis by Laboratory Methods," *Small Animal Clinical Diagnosis by Laboratory Methods*, pp. 63–91, 2012, doi: 10.1016/C2009-0-59996-5.
- [3] L. K. Riley and J. Rupert, "Evaluation of Patients with Leukocytosis," *American family physician*, vol. 92, no. 11, pp. 1004–1011, 2015.
- [4] X. Fuentes-Arderiu and D. Dot-Bach, "Measurement uncertainty in manual differential leukocyte counting," *Clinical Chemistry and Laboratory Medicine*, vol. 47, no. 1, pp. 112–115, 2009, doi: 10.1515/CCLM.2009.014.
- [5] P. Font *et al.*, "Inter-observer variance with the diagnosis of myelodysplastic syndromes (MDS) following the 2008 WHO classification," *Annals of Hematology*, vol. 92, no. 1, pp. 19–24, 2013, doi: 10.1007/s00277-012-1565-4.
- [6] M. I. Razzak, S. Naz, and A. Zaib, "Deep learning for medical image processing: Overview, challenges and the future," *Lecture Notes in Computational Vision and Biomechanics*, vol. 26, pp. 323–350, 2018, doi: 10.1007/978-3-319-65981-7\_12.




- [7] H. Hu, Y. Shao, and S. Hu, "A Review of the Application of Deep Learning in Brachytherapy," *OALib*, vol. 07, no. 07, pp. 1–9, 2020, doi: 10.4236/oalib.1106589.
- [8] Q. Wang, L. Chang, M. Zhou, Q. Li, H. Liu, and F. Guo, "A spectral and morphologic method for white blood cell classification," *Optics and Laser Technology*, vol. 84, pp. 144–148, 2016, doi: 10.1016/j.optlastec.2016.05.013.
- [9] H. Cao, H. Liu, and E. Song, "A novel algorithm for segmentation of leukocytes in peripheral blood," *Biomedical Signal Processing and Control*, vol. 45, pp. 10–21, 2018, doi: 10.1016/j.bspc.2018.05.010.
- [10] S. Dasariraju, M. Huo, and S. McCalla, "Detection and classification of immature leukocytes for diagnosis of acute myeloid leukemia using random forest algorithm," *Bioengineering*, vol. 7, no. 4, pp. 1–12, 2020, doi: 10.3390/bioengineering7040120.
- [11] W. J. Al-Kubaisy and M. Mahmood, "Classification of texture using random box counting and binarization methods," *Bulletin of Electrical Engineering and Informatics*, vol. 10, no. 1, pp. 533–540, 2021, doi: 10.11591/eei.v10i1.2480.
- [12] P. Rastogi, K. Khanna, and V. Singh, "LeuFeatx: Deep learning-based feature extractor for the diagnosis of acute leukemia from microscopic images of peripheral blood smear," *Computers in Biology and Medicine*, vol. 142, p. 105236, 2022, doi: 10.1016/j.compbiomed.2022.105236.
- [13] A. Acevedo, S. Alf  rez, A. Merino, L. Puigvi, and J. Rodellar, "Recognition of peripheral blood cell images using convolutional neural networks," *Computer Methods and Programs in Biomedicine*, vol. 180, p. 105020, 2019, doi: 10.1016/j.cmpb.2019.105020.
- [14] H. Kutlu, E. Avci, and F.   zyurt, "White blood cells detection and classification based on regional convolutional neural networks," *Medical Hypotheses*, vol. 135, p. 109472, 2020, doi: 10.1016/j.mehy.2019.109472.
- [15] T. A. Elhassan *et al.*, "Classification of Atypical White Blood Cells in Acute Myeloid Leukemia Using a Two-Stage Hybrid Model Based on Deep Convolutional Autoencoder and Deep Convolutional Neural Network," *Diagnostics*, vol. 13, no. 2, p. 196, 2023, doi: 10.3390/diagnostics13020196.
- [16] K. M. Naing, V. Kittichai, T. Tongloy, S. Chuwongin, and S. Boonsang, "Images Retrieval and Classification for Acute Myeloid Leukemia Blood Cell Using Deep Metric Learning," *Communications in Computer and Information Science*, vol. 1863 CCIS, pp. 27–39, 2023, doi: 10.1007/978-3-031-42430-4\_3.
- [17] Q. Wang, S. Bi, M. Sun, Y. Wang, D. Wang, and S. Yang, "Deep learning approach to peripheral leukocyte recognition," *PLoS ONE*, vol. 14, no. 6, p. e0218808, 2018, doi: 10.1371/journal.pone.0218808.
- [18] A. Shakarami, M. B. Menhaj, A. Mahdavi-Hormat, and H. Tarrah, "A fast and yet efficient YOLOv3 for blood cell detection," *Biomedical Signal Processing and Control*, vol. 66, p. 102495, 2021, doi: 10.1016/j.bspc.2021.102495.
- [19] C. Matek, S. Schwarz, K. Spiekermann, and C. Marr, "Human-level recognition of blast cells in acute myeloid leukaemia with convolutional neural networks," *Nature Machine Intelligence*, vol. 1, no. 11, pp. 538–544, 2019, doi: 10.1038/s42256-019-0101-9.
- [20] U. Nepal and H. Eslamiat, "Comparing YOLOv3, YOLOv4 and YOLOv5 for Autonomous Landing Spot Detection in Faulty UAVs," *Sensors*, vol. 22, no. 2, 2022, doi: 10.3390/s22020464.
- [21] K. M. Naing *et al.*, "Automatic recognition of parasitic products in stool examination using object detection approach," *PeerJ Computer Science*, vol. 8, p. e1065, 2022, doi: 10.7717/PEERJ-CS.1065.
- [22] Z. Han *et al.*, "One-stage and lightweight CNN detection approach with attention: Application to WBC detection of microscopic images," *Computers in Biology and Medicine*, vol. 154, p. 106606, 2023, doi: 10.1016/j.compbiomed.2023.106606.
- [23] C. Matek, S. Schwarz, C. Marr, and K. Spiekermann, "A Single-cell Morphological Dataset of Leukocytes from AML Patients and Non-malignant Controls [Data set]," 2019.
- [24] M. K. U  ar, M. Nour, H. Sindi, and K. Polat, "The Effect of Training and Testing Process on Machine Learning in Biomedical Datasets," *Mathematical Problems in Engineering*, vol. 2020, p. 2836236, 2020, doi: 10.1155/2020/2836236.
- [25] C. C. D. Group, "CiRA CORE," 2023, [Online]. Available: <https://git.cira-lab.com/cira>
- [26] C. Shorten and T. M. Khoshgoftaar, "A survey on Image Data Augmentation for Deep Learning," *Journal of Big Data*, vol. 6, no. 1, 2019, doi: 10.1186/s40537-019-0197-0.
- [27] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2016, pp. 779–788, doi: 10.1109/CVPR.2016.91.
- [28] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, IEEE, Jul. 2017, pp. 6517–6525, doi: 10.1109/CVPR.2017.690.
- [29] J. Redmon and A. Farhadi, "YOLOv3: An Incremental Improvement," *arXiv e-prints*, p. 1804.02767, 2018, doi: 10.48550/arXiv.1804.02767.
- [30] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "YOLOv4: Optimal Speed and Accuracy of Object Detection," *arXiv preprint*, p. 2004.10934, 2020, doi: 10.48550/arXiv.2004.10934.
- [31] C.-Y. Wang, A. Bochkovskiy, and H.-Y. M. Liao, "YOLOv7: Trainable Bag-of-Freebies Sets New State-of-the-Art for Real-Time Object Detectors," *arXiv preprint arXiv:2207.02696*, pp. 7464–7475, 2023, doi: 10.1109/cvpr52729.2023.00721.
- [32] V. Kittichai *et al.*, "Classification for avian malaria parasite Plasmodium gallinaceum blood stages by using deep convolutional neural networks," *Scientific Reports*, vol. 11, no. 1, p. 16919, 2021, doi: 10.1038/s41598-021-96475-5.
- [33] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006, doi: 10.1016/j.patrec.2005.10.010.
- [34] A. Tharwat, "Classification assessment methods," *Applied Computing and Informatics*, vol. 17, no. 1, pp. 168–192, 2018, doi: 10.1016/j.aci.2018.08.003.
- [35] V. Maslej-Kre  n  kov  , M. Sarnovsk  , P. Butka, and K. Machov  , "Comparison of deep learning models and various text pre-processing techniques for the toxic comments classification," *Applied Sciences (Switzerland)*, vol. 10, no. 23, pp. 1–26, 2020, doi: 10.3390/app10238631.
- [36] J. Hosang, R. Benenson and B. Schiele, "Learning Non-maximum Suppression," 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 2017, pp. 6469–6477, doi: 10.1109/CVPR.2017.685.
- [37] A. J. Scott, D. W. Hosmer, and S. Lemeshow, "Applied Logistic Regression," *Biometrics*, vol. 47, no. 4, p. 1632, 1991, doi: 10.2307/2532419.
- [38] S. S. R. Bairaboina and S. R. Battula, "Ghost-ResNeXt: An Effective Deep Learning Based on Mature and Immature WBC Classification," *Applied Sciences (Switzerland)*, vol. 13, no. 6, p. 4054, 2023, doi: 10.3390/app13064054.

## BIOGRAPHIES OF AUTHORS






**Kaung Myat Naing**    received the Bachelor of Engineering in Electronics from Technological University (Mawlamyine), Mon State, Myanmar, in 2011 and the Master of Science in Technical Education (Electrical Engineering) from King Mongkut's University of Technology North Bangkok (KMUTNB), Thailand, in 2018. He is currently pursuing the Ph.D. degree in advanced manufacturing system engineering at the College of Advanced Manufacturing Innovation, King Mongkut's Institute of Technology Ladkrabang (KMITL), Thailand. His research interests include deep learning applications and data evaluations. He can be contacted at email: 62609004@kmitl.ac.th.






**Veerayuth Kittichai**    received the B.Sc degree in medical technology from Chulalongkorn University (CU), Thailand, in 2007, the M.Sc degree in medical science (molecular biology and genetics) from Chulalongkorn University (CU), Thailand, in 2011, and Ph.D. degree in medical science (molecular biology and genetics) from Chulalongkorn University (CU), Thailand, in 2015. He has been a postdoctoral researcher with the Mahidol Vivax Research Unit, Faculty of Tropical Medicine, Mahidol University (MU). He is the author of 7 articles. His research interests deep learning and applications in medical field. He is also a lecturer of Faculty of Medicine, KMITL since 2017. He can be contacted at email: veerayuth.ki@kmitl.ac.th.






**Teerawat Tongloy**    received the B.E degree in Computer Engineering from Khonkean University (KKU), Thailand, in 2015 and the M.Eng. (Advanced Manufacturing System Engineering) from King Mongkut's Institute of Technology Ladkrabang (KMITL), Thailand, in 2020. He is currently an engineer at the College of Advanced Manufacturing Innovation (AMI), KMITL. His research interests include deep learning applications and robot operating systems. He can be contacted at email: teerawat.tongloy@gmail.com.



**Santhad Chuwongin**    received the B.Eng degree in Telecommunications Engineering from King Mongkut's Institute of Technology Ladkrabang (KMITL), Thailand, in 1998, the M.Sc degree in electrical engineering from University of Texas at Arlington, USA in 2008 and Ph.D degree in electrical engineering from University of Texas at Arlington, USA in 2012. He is also a lecturer at the College of Advanced Manufacturing Innovation, KMITL. He is the author of more than 10 articles, and more than 10 inventions. His research interests industrial robotic and deep learning, optoelectronic materials, and devices: LEDs, lasers, detectors, and solar cells, silicon photonics and photonic integration, flexible photonics and bio-inspired optics, renewable energy, and flexible solar cells. He can be contacted at email: santhad.ch@kmitl.ac.th.



**Siridech Boonsang**    received the B.Eng degree in Electrical Engineering from King Mongkut's Institute of Technology Ladkrabang (KMITL), Thailand, in 1998, the M.Sc degree in Electrical and Electronic Engineering from University of Manchester Institute of Science and Technology (UMIST), UK in 2001 and Ph.D degree in Instrumentation from UMIST in 2004. Since 2010, he has been an Associate Professor with the Electrical Engineering Department, KMITL. He is the author of two books, more than 10 articles, and more than 10 inventions. His research interests include instrumentation, optical metrology, deep learning and applications, robotics. He is also the Dean of Faculty of Information Technology, KMITL since 2020. He was a recipient of the Outstanding Technologist Award from Foundation for the Promotion of Science and Technology under the Patronage of H.M. the King in 2019. He can be contacted at email: siridech.bo@kmitl.ac.th.