Received: June 6, 2023. Revised: June 29, 2023.

International Journal of
Intelligent Engineering & Systems

http://www.inass.org/

A New Model: Commutative Hypercomplex – Convolutional Neural Network to Classify Acute Lymphoblastic Leukemia Images

Arif Muntasa[1]* **Rima Tri Wahyuningrum**[1] **Durotun Nafisah**[1]

[1]*Universitas Trunojoyo Madura, East Java, Indonesia*
*Corresponding Author's Email: arifmuntasa@trunojoyo.ac.id

Abstract: The purpose of the paper is to classify the Leukemia images. In this paper, we proposed a commutative model of a convolutional neural network for Leukemia image classification. We employ commutative hypercomplex modeling A[+1, -1] and A[-1, +1] to build the new model. We hire an augmentation model to enrich the image data sets for the training sets through rotation, zooming, and flipping. We evaluated our proposed method using acute lymphoblastic leukemia image database type 2 (ALL-IDB2). The results show that our proposed method has delivered the best average accuracy at 96.43% for A[+1,-1] and 97.05 for A[-1,+1]. We have measured and found maximum accuracy at 100% for A[+1,-1] and A[-1,+1]. Comparison results show that our proposed method outperformed K-nearest neighbor, support vector machine-radial basis function, support vector machine-linear, support vector machine-polynomial, naïve bayes gaussian, naïve bayes complement, decision tree, and colour hybrid modelling.

Keywords: Commutative, Hypercomplex, Leukemia, Convolutional neural networks, Image classification.

1. Introduction

Acute lymphoblastic leukemia (ALL), a disease primarily affecting children, has become one of the top causes of cancer-related deaths. An expert doctor must assess whether someone has Leukemia. The diagnosis of ALL can be complex and time-consuming, requiring thorough examination of bone marrow and blood samples with laboratory techniques such as flow cytometry, immunohistochemistry, and cytogenetics. This method also carries the risk of not being able to detect minimal residual disease (MRD). Furthermore, it is highly costly and requires substantial time investment; many people cannot bear the cost or fear that the long wait for a diagnosis may negatively impact their health.

Researchers have conducted and produced numerous results using the gray level co-occurrence matrix (GLCM) regarding acute lymphoblastic Leukemia [1–3]. Theoretically, GLCM features can be utilized for many computer vision tasks, for instance, image classification, detection, recognition, restoration, and segmentation. The GLCM forms a matrix that measures the probability of a pixel's center concerning its neighbor based on distance and angle. GLCM can be processed to deliver the main object features. The size of the GLCM does not rely on the size of the image but on its utmost gray level. Therefore, various image sizes with the same maximum gray level value will produce the same GLCM size, but GLCM formation will take different amounts of time. One of them is second-order statistic-based features such as contrast, homogeneity, energy, and correlation, which contribute to reducing the characteristic dimensions of an object while concurrently reducing computation time. The critical processes required for object classification were pre-processing, building GLCM, feature extraction, and classification. They have produced GLCM and extracted the eleven shape feature vectors employed in the study. This method was used to capture the spatial association between pixels in an image and essential details related to its texture [4]. Nonetheless, it also has several drawbacks; it is vulnerable to image noise, orientation, and angle, while its accuracy depends on differences in distance and angle used. In addition, complex textures cannot be captured using the GLCM, thus requiring more advanced analysis to produce appropriate features.

A color-based segmentation approach has been suggested for classifying acute lymphoblastic Leukemia based on stages, pre-processing, segmentation, feature extraction, and classification. A 95.38% precision has been attained in representing objects via shape and texture characteristics [5]. Flaws in image segmentation may result in incorrect categorization, necessitating enhancement. Texture characteristics, including homogeneity, entropy, energy, and contrast, have been produced to overcome the limitation of GLCM. The accuracy rate has been boosted to 96.67% due to adding these four characteristics. Even though image segmentation enhancement produced a mere 1.29% increase in accuracy, research suggests that refining segmentation alone may not substantially improve [3].

Muntasa and Yusuf [3] have surpassed the limitations of the GLCM by using multiple channels, distances, and orientations. They utilized three channels: red, green, and blue, to capture all object characteristics in the image. Then, they built a GLCM for each channel, resulting in sixteen GLCMs with sixty-four features each. Multiplying sixteen GLCMs by four resulted in one hundred and ninety-two features used to classify the images. The experimental results revealed that this approach produced better accuracy than previous research [1]. An accuracy of 96.97% was obtained with the Chebyshev method, and 96.16% was obtained with the Canberra method. It suggests that utilizing multiple perspectives at various angles and orientations can increase accuracy for all classifications. A limitation of this research is that segmentation errors may cause classification errors; however, even correct segmentation results do not always guarantee accurate classifications [6].

Classification based on the CYMK model has been proposed by Abdeldaim et al. [7]. They offered the CYMK model from the RGB model. They improved image sample sets using Histogram Equalization, then determined the thresholding value to remove the image background. The researchers have employed several feature models from different perspectives to extract the image, i.e., color, texture, and shape. They use several similarity measurement methods, such as K-NN, support vector machine (radial basis function (SVM-RBF), linear (SVM-L), and polynomial (SVM-P)), naïve bayes gaussian, naïve bayes complement, and decision tree to classify the image. They employed ALL-IDB2 to evaluate their proposed method. The results show that 96.01%, 92.8%, 93.43%, 93.89%, 89.97%, 86.02%, and 86.81% accuracies for KNN, SVM-RBF, SVM-L, SVM-P, NB-G, NB-C, and Decision Tree respectively.

In Laosai and Chamnongthai [8, 9] improved ALL images' quality by employing contrast enhancement and morphological color segmentation techniques. Moreover, they classified features using fuzzy C-mean, resulting in an accuracy rate of 98% on the ALL-IDB1 dataset. Nevertheless, additional tests with various indices are necessary to determine if their method can be relied upon. Additionally, they developed a new approach for classifying Leukemia images by combining knowledge-based morphology with cluster of differentiation (CD) markers. This methodology was also used to classify images by isolating white blood cells based on four characteristics: nucleoli, color, texture, and shape. Consequently, with the help of CD markers, the proposed algorithm achieved an accuracy rate of 99.67%; however, no information was provided regarding the computation time for image classification [10].

In recent years, various attempts have been made to investigate ALL image databases using machine learning (ML) techniques to facilitate the detection, classification, and diagnosis of ALL. Medical imaging has revealed significant prospects for ML algorithms, as they can reveal patterns and structures that may go unnoticed. Convolutional neural networks (CNNs) are a general approach that is highly effective for image recognition applications such as medical image processing. CNNs eliminate the need for pre-processing, feature extraction, and classification operations by learning directly from the image datasets. It reduces computation time while increasing accuracy in object recognition and segmentation applications. CNNs have several advantages, like dealing with high-dimensional data, unsupervised feature extraction, and brief generalization to novel data space; nonetheless, it has some pitfalls, such as overfitting, lack of interpretability, and computational complexity. Even though these issues exist, CNNs are still a favored and efficient tool for many computer vision tasks, such as diagnosing and classifying medical images, including ALL classification.

To improve the classification of Leukemia, researchers have devised a deep learning framework and implemented data augmentation techniques to bolster the variability present in the training dataset. To mitigate the issue of over-fitting, the researchers employed the utilization of transfer learning alongside a dense convolutional neural network (DCNN) architecture. Additionally, an ensembled technique was applied for the extraction of features.

Received: June 6, 2023. Revised: June 29, 2023.

Numerous deep convolutional neural network (DCNN) models, namely, AlexNet, VGGNet-16, VGGNet-19, MobileNet, ShuffleNet, and two NASNet variants, InceptionV3 and Xception, along with DenseNet20 and MobileNet have been deployed in the context of classifying Leukemia. Collectively, these models have yielded an essential accuracy of 96.58%. Nonetheless, it must be noted that there exist limitations inherent in this methodology, such as the prolonged duration required for model construction and the substantiated expense involved in achieving accurate image classification [11].

Researchers have adopted deep learning architectures and applied data augmentation to improve the diversity of the training set for image classification. Transfer learning with DCNN architectures and an ensemble technique are utilized to achieve similar feature extraction outcomes to avoid overfitting. Various DCNNs such as LeNet, AlexNet, VGGNet, Efficientbet Bo till B7, ResNet50, InceptionResNetV2, Googlenet, and Inception have been employed to facilitate Leukemia classification. Unfortunately, this method is time-consuming and costly in constructing the model for image categorization [12, 13].

The visual geometry group (VGG) suggested employing VGG16 and VGG19 architectures for image classification and the xception CNN for data extraction and labeling. Research results revealed that VGG16 had a 92.48% accuracy, VGG19 achieved 91.59%, and xception CNN furnished 90.41% accuracy after four hundred passes with a 0.00001 learning rate. The proposed approach has been evaluated using C-NMC 2019 dataset, and their experiments have produced F1-score of 92.6%, 91.7%, and 90.7% for VGG16, VGG19, and xception CNN, respectively [14]. Nevertheless, these advanced architectures possess numerous trainable parameters, necessitating powerful computing hardware to quickly execute the image extraction and classification processes.

A new technique has been devised to escalate the performance of VGG16 and VGG19 by transferring knowledge and utilizing VGG-f architecture for feature extraction. Subsequently, the features assign labels to images using a support vector machine. VGG-f is modeled after AlexNet, yet it employs kernels of smaller size in the first, third, and fourth stages of convolution [15–17]. Furthermore, the number of fully connected neurons in VGG-f is retained as 4096, similar to that of AlexNet. The researchers have also augmented image quality by transforming color space from the original to CIELAB color space.

A modern approach of exchange learning utilizing VGGf to extricate the highlights is proposed to move forward the VGG16 and VGG19. They employed a back vector machine to classify the highlight extraction comes about. VGG-f could be a convolutional neural organize design that's built based on AlexNet. The VGG-f utilized littler parts for the convolution preparation within, to begin with, the third and fourth [17]. Be that as it may, they still have 4096 neurons for fully-connected layers like the AlexNet design. They made strides in the image by changing it into a CIELAB color space. Image division is conducted to partition the most question and foundations, and the comes about were utilized to calculate the shape highlights. At long last, the bolster vector machine is used to classify the image. They claimed the proposed strategy had been assessed utilizing massive databases, i.e., ALL-IDB1. ALL-IDB2, Leukocytes, and CellaVision datasets. The comes about appears that the VGG-f has delivered 99% exactness.

Additionally, they have conducted image segmentation to eliminate the image background to extract the main object accurately. In the concluding step, they used a support vector machine to classify the ALL image data testing. The authors claim this technique has been thoroughly tested with numerous databases, such as ALL-IDB1, ALL-IDB2, Leukocytes, and CellaVision datasets. It has achieved a success rate of 99% when VGG-f is used [17].

Traditional CNNs have a problem with complex or multidimensional data because they can only overcome simple numbers. It can cause mistakes in the information they deliver as the features. We proposed complex numbers to overcome incorrect details. Our strong point is that we can extract more object features because our proposed method can store information in two directions instead of just one. We employ Hypercomplex-valued CNNs (HC-CNNs) to solve this problem. Our proposed architecture utilizes numbers like quaternions, octonions, and Clifford algebras. These numbers can overcome more complex data than regular numbers. We created a new model to process images called CH-CNNs to fix when mistakes happen during their processing. A CHCNN is a computer system that can work with detailed information. It is like math with more imaginary numbers. It is helpful because it allows for even more ways to show things. CHCNNs are a type of technology that uses special math to find essential parts of the information you give it. It is different than regular math and can discover more critical components. CHCNNs can work with data that does not follow a specific order, and the outcome changes based on the order. Regular CNNs are made for data that is the same regardless of the order in which it is carried out. In simple terms, CHCNNs can understand how different features in the input are related to each other because hypercomplex numbers have both size and direction.[18].

In addition, this paper is divided into four sections: introduction, material and method, experimental and discussions, and conclusions. The "Introduction part" discusses acute lymphoblastic Leukemia (ALL), its impact on cancer-related deaths, and the challenges in diagnosing the disease. Furthermore, it also discusses prior studies that explored various approaches to classify acute lymphoblastic Leukemia. These methods encompass the texture model, second-order statistic, color-based, probabilistic-based, and convolutional neural network models. Research gaps, brief purposes, innovation, and strong points are also mentioned in this part. We explained our proposed method in the "Material and Method." We also demonstrate in detail the results of "The Experiment and Discussion" as the outcomes of the new model and a comparison with existing methods. Finally, the general conclusion and future research are presented as well.

2. Material and method

In this study, we presented a novel classification technique for Leukemia images utilizing commutative hypercomplex-based convolutional neural networks (CH-CNNs). Before training the model, we divide our dataset into training and validation sets. We apply image augmentation to create multiple new images to increase the size of the training dataset. This process involves transforming the original image in various ways, such as rotation, zooming, and flipping. It can ensure the model accurately recognizes an object's different sizes, shapes, and orientations. Additionally, augmenting the data can reduce overfitting by introducing randomness and diversity into the training data - thus preventing it from simply memorizing rather than understanding the fundamental patterns of an image.

The convolution process is a mathematical operation that combines the input image with a set of learnable filters to generate a set of feature maps in a convolutional neural network (CNN) employing commutative hypercomplex values. The convolution filters are represented as commutative hypercomplex values, an extension of complex numbers with additional components beyond the real and imaginary portions of the original complex numbers. In contrast to conventional real-valued filters, CNNs use commutative hypercomplex values to capture more abstract and complex input image features. Representing the filters as commutative hypercomplex values, including commutative ones, enables the neural network to generate a more complex feature map and abstract characteristics of the input image. In addition, the convolution process can capture spatially invariant features and reduce the required parameters to achieve greater precision and stability. Here is our proposed method, as shown in Fig. 1.

2.1 ALL-IDB2 as experimental datasets

We use the dataset of acute lymphoblastic Leukemia image database type 2 (ALL-IDB2) to evaluate our proposed method. It consists of 260 images, of which 130 images are categorized as healthy images, and the remaining are Leukemia images [19]. We obtain the ALL-IDB2 datasets from the department of computer science - università degli studi di Milano (in acronym "UniMi"). Expert oncologists have provided the classification and location of ALL lymphoblasts for every image in the dataset, as shown in the image samples in Fig. 2. The ALL-IDB2 image datasets are crop results from the ALL-IDB1 datasets with a 2592 x 1944 resolution size of 24-bit color depth.

We employ many variables to represent the model and proposed method, as shown in Table 1.

2.2 Image input

Fig. 1 depicts the first layer of a CH-CNN, also known as the input layer, responsible for receiving the image datasets supplied into the network. It comprises numerous neurons or nodes representing a pixel or other image component. Our CH-CNNs input layer consists of three dimensions, which we refer to as height, breadth, and depth. The depth dimension reflects the number of red, green, and blue color channels or characteristics extracted from the image.

In this study, we utilized 130 images for the healthy population and the remaining images for the leukemia population. In addition, we altered the pixel values of each channel in the input image. There are 255 possible values for each D symbol, representing the red, green, and blue channels. In addition, we divided each pixel by 28-1; 8 illustrates image data as the following equation.

$$\mathcal{F}_{i,j,D} = \frac{f_{i,j,D}}{(2^8-1)} \tag{1}$$

Furthermore, we replace the real model space with hypercomplex space by adding zero matrices on

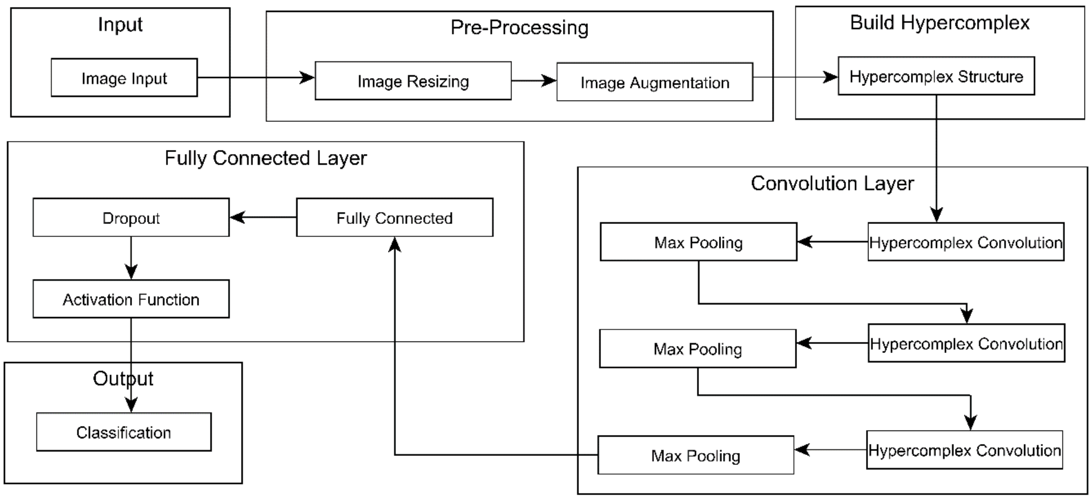

Figure. 1 Our proposed method

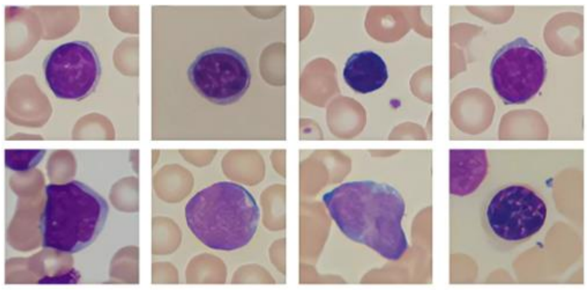

Figure. 2 Image sample datasets: First row indicates healthy cells, Second row is lymphoblasts [19]

Table 1. List of variables used

No	Variables	Explanation
1	F and $\mathcal{F}$	Image matrix after normalization
2	i and j	Row and column indexes for the image
3	i' and j'	Row and column indexes for the image process result
4	D	Channel
5	D'	Number of kernels
6	(x, y)	An index of pixel position
7	(x', y')	The new index after rotation
8	θ	Angle
9	m and n	Number of rows and columns
10	μ and v	Index for columns for both
11	s_x and s_y	Zooming scale for a row and column
12	z'_r and z'_c	Row and column of the zooming image
13	(x'_g, y'_g)	New position after flipping
14	α	Constant value
15	$\mathbb{H}$	Hypercomplex domain
16	p and q	The real and imaginary values
17	$\mathbb{J}^{(h)}(p,k)$	Hypercomplex output
18	σ_A	Activation function
19	$\left(b^{(h)}(k)\right)$	Hypercomplex bias
20	$\left(\mathbb{I}^{(h)}\right)$	Hypercomplex value image
21	$\left(\mathbb{F}^{(h)}\right)$.	Hypercomplex value filter
22	$\mathbb{J}^{(h)}(p,k)$	Hypercomplex output
23	$\mathcal{K}$	Image kernel
24	$\mathbb{R}^{H'\times W'\times D\times D'}$	The domain of a kernel
25	H', W', D', and D	Kernel height, Kernel width, number of kernels, and number of channels
26	W'	Kernel width
27	$\mathcal{Y}$	Image convolution result
28	b	Bias
29	P and S	Padding and stride
30	P_h^-, P_h^+, P_w^-, and P_w^+	Top image border, bottom image border, left image border, and the right image border.
31	S_h and S_w	Vertical and Horizontal stride
32	H'' and W''	Height and width of the image convolution results
33	W''' and H'''	Maximum values of row and column indexes for the image process result
34	$\mathbb{P}_w$ and $\mathbb{P}_h$	Width and height pooling dimension result
35	$\mathcal{w}_{ij}$	Random weight
36	$\mathcal{B}_j$	Bias initials
37	$p(x)$ and $q(x)$	Actual and prediction values of the model
38	TP, TN, FP, and FN	True Positive, true negative, false positive, and false negative

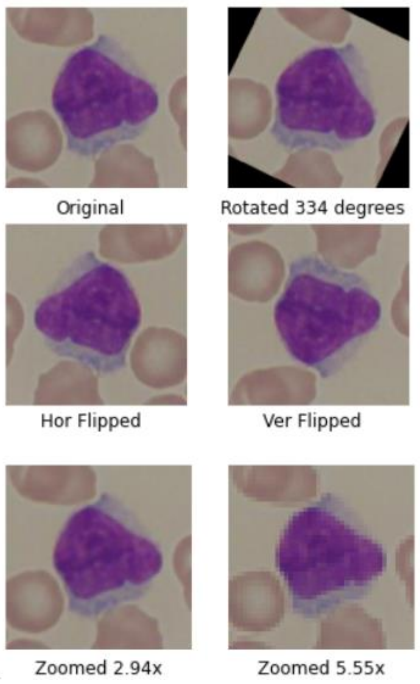

Figure. 3 Sample of image augmentation

the fourth dimension. The next step is to separate the training and validation data, followed by an augmentation process to reproduce the training data. In this case, we hired image rotation, zooming, and flipping to rebuild the training set.

2.3 Image rotation

Suppose we have an image with m rows and c columns. We can conduct rotate an image using the following equation:

$$x' = x\cos\theta - y\sin\theta \tag{2}$$

$$y' = x\sin\theta + y\cos\theta \tag{3}$$

In this case, (x, y) depicts an index of pixel position, where m indicates row and n presents the column of the image, (x', y') express new position after rotation, and θ interprets rotation angle.

2.4 Image zooming

Zooming in on an image, or scaling it up or down, is a standard operation performed on digital images. This method sees extensive use in computer vision and digital image processing settings. In most cases, this is accomplished by interpolating the values of the original image's pixels to generate new higher-resolution pixels. Conversely, when you zoom out, the image gets smaller, giving the impression that it is farther away and has less information. Subsampling or decimation is commonly used to achieve this effect, in which some of the original image's pixel values are discarded to create a smaller, lower-resolution image, as written in the following equation.

$$z'_r = s_x . m \tag{4}$$

$$z'_c = s_y . n \tag{5}$$

Symbol of m and n indicates the row and column sizes from the original image, while z'_r and z'_c Indicate the row and column of the zooming image. We utilized s_x and s_y to depict the zooming scale for a row and column.

2.5 Image flipping

In addition to rotation and zooming, we flip the image to obtain more sample variants of the image datasets as the following equation.

$$(x'_g, y) \to (x, n - y) \tag{6}$$

$$(y'_g, y) \to (m - x, y) \tag{7}$$

Fig. 3 displays the various flipping models that use the input images by rotation, zooming, vertical flipping, and horizontal flipping. Image augmentation aims to enrich the training set with new information to assist the network in learning more generalizable and stable patterns.

Fig. 3 displays the various flipping models that use the input images by rotation, zooming, vertical flipping, and horizontal flipping. Image augmentation aims to enrich the training set with new information to assist the network in learning more generalizable and stable patterns

2.6 Hypercomplex building layer

The technique of producing hypercomplex images entails the generation and alteration of images by employing hypercomplex numbers as the primary tool. The hypercomplex model, which also contains quaternion and octonion numbers, is a generalization made from complex numbers. Complex numbers are the building blocks of hypercomplex numbers, which can be considered an extension of complex numbers. When making hypercomplex images, each pixel in an image is represented by a number that is itself hypercomplex. It is performed as part of the process of creating hypercomplex images. The significance of the hypercomplex number determines not only the color but also the position of each pixel in the image. There are several ways in which the construction of hypercomplex images and mathematical physics, particularly string theory and quantum mechanics, can be compared. With the assistance of hypercomplex numbers, scientists have been able to model the behavior of subatomic particles and space-time geometry.

A method that can describe color information in a more expressive and adaptable manner has been developed using hypercomplex four-dimensional integers. The actual part of the number corresponds to the pixel's intensity, while the three fake components correspond to the color information of the image pixels. According to this method, each pixel is represented as a hypercomplex 4-D number. It enables the communication of more nuanced color relationships and a more comprehensive range of tones than was previously feasible.

One use of hypercomplex 4-D numbers in image processing is color image segmentation. Dividing an image into portions reflecting a different entity or set of characteristics is called segmentation. It is possible to improve the quality of image segmentation algorithms by defining color characteristics with hypercomplex four-dimensional integers. These color characteristics are resistant to changes in lighting and shading.

In general, using hypercomplex 4-dimensional integers in image processing applications can significantly facilitate the encoding of color information. The value of this assistance could be derived from various sources. It is feasible to bypass some of the shortcomings of typical RGB color models and design image processing algorithms that are more powerful and versatile if hypercomplex numbers are used to describe color data. It is one of the advantages of utilizing hypercomplex numbers.

It is called a commutative hypercomplex convolution neural network (CH-CNN), a subclass of convolutional neural networks. CH-CNNs are CNNs that can process commutative hypercomplex-valued input. In CH-CNN, the input data and the network weights are represented as hypercomplex numbers, allowing the information's amplitude and phase to be accurately represented. In certain circumstances, the values of R can stand in for the real number. On top of this, the hypercomplex can be described in the following way:

$$\alpha + 0i_1 + 0i_1 + 0i_1 + \cdots + 0i_n \in \mathbb{H} \tag{8}$$

The term "hypercomplex" refers to a collection of hypercomplex numbers that can be employed using either the dot product or multiplication and addition as the fundamental algebraic operations. Imagine that we have two highly complex numbers, as shown below:

$$p = p_0 + p_1 i_1 + p_2 i_2 + p_3 i_3 + \cdots + p_n i_n \tag{9}$$

$$q = q_0 + q_1 i_1 + q_2 i_2 + q_3 i_3 + \cdots + q_n i_n \tag{10}$$

Furthermore, the multiplication between the values of p and q can be expressed using the following equation.

$$p \times q = \left(p_0 q_0 + \textstyle\sum_{\mu,v=1}^{n} p_\mu q_v a_{\mu v,0}\right) + \left(p_0 q_1 + p_1 q_0 + \textstyle\sum_{\mu,v=1}^{n} p_\mu q_v a_{\mu v,1}\right) i_1 + \cdots + \left(p_0 q_n + p_n q_0 + \textstyle\sum_{\mu,v=1}^{n} p_\mu q_v a_{\mu v,n}\right) i_n \tag{11}$$

In most cases, the real number and convolution layers are utilized in the convolution process. In addition to this, the convolution layer is changed, which results in the real number becoming hypercomplex, as seen in the following equation:

$$\mathbb{J}^{(h)}(p,k) = \sigma_A\left(b^{(h)}(k) + \left(\mathbb{I}^{(h)} * \mathbb{F}^{(h)}\right)(p,k)\right) \tag{12}$$

In this case, $\mathbb{J}^{(h)}(p,k)$ depicts the hypercomplex output as the convolution results using an activation function σ_A . In the activation function, the hypercomplex bias $\left(b^{(h)}(k)\right)$ has been added to the convolution results between a hypercomplex value image $\left(\mathbb{I}^{(h)}\right)$ and filter $\left(\mathbb{F}^{(h)}\right)$. First, describing an image $\mathbb{I}^{(h)}$ and $\mathbb{F}^{(h)}$ filter using C channels specified on a four-dimensional hypercomplex algebra is possible as the following equation

$$\mathbb{I}^{(h)} = \mathbb{I}_0 + \mathbb{I}_1 i + \mathbb{I}_2 j + \mathbb{I}_3 k \tag{13}$$

$$\mathbb{F}^{(h)} = \mathbb{F}_0 + \mathbb{F}_1 i + \mathbb{F}_2 j + \mathbb{F}_3 k \tag{14}$$

In Eq. (13), the symbols of $\mathbb{I}_0, \mathbb{I}_1 i, \mathbb{I}_2 j,$ and $\mathbb{I}_3 k$ points out the real image using C channels, while the values of $\mathbb{F}_0, \mathbb{F}_1 i, \mathbb{F}_2 j, \mathbb{F}_3 k$ represents a hypercomplex image filter matrix value.

2.7 Convolution layer

If the image kernel is indicated by using $\mathcal{K} \in \mathbb{R}^{H' \times W' \times D \times D'}$, then H' points out a kernel height, W' performs kernel width, and D' represents the number of kernels. Moreover, we could write an image convolutional equation as follows

$$\mathcal{Y} = \mathcal{F} \otimes \mathcal{K} \tag{15}$$

$$\mathcal{Y}_{i',j',D'} = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{d=1}^{D}\mathcal{K}_{i,j,d} \times \mathcal{F}_{i'+i-1,j'+j-1,d,D'} \quad (16)$$

$\mathcal{Y}_{i',j',D'}$ implies the image convolution result. The indexes of $\mathcal{Y}$ are i specify new height, j indicates an image width, and D' represents the number of kernels. We could add bias values on the convolutional neural network results as follows.

$$\mathcal{Y}_{i',j',D'} = b^{D'} + \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{d=1}^{D}\mathcal{K}_{i,j,d} \times \mathcal{F}_{i'+i-1,j'+j-1,d,D'} \quad (17)$$

Padding (P) and stride (S) The two primary parameters are required for the convolution process to occur successfully. Adding values equal to zero on an image border is known as "zero padding", and it appears rather frequently. The padding of zero pixels on the top image border (P_h^-), bottom image border (P_h^+), left image border (P_w^-), and the right image border (P_w^+) as follows

$$\mathcal{Y}_{i',j',D'} = \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{d=1}^{D}\mathcal{K}_{i,j,d} \times \mathcal{F}_{S_h(i'-1)+i-P_h^-,S_w(j'-1)+j-P_w^-,d,D'} \quad (18)$$

Moreover, we also add bias values when the convolution process is carried out as the following equation.

$$\mathcal{Y}_{i',j',D'} = b^{D'} + \sum_{i=1}^{m}\sum_{j=1}^{n}\sum_{d=1}^{D}\mathcal{K}_{i,j,d} \times \mathcal{F}_{S_h(i'-1)+i-P_h^-,S_w(j'-1)+j-P_w^-,d,D'} \quad (19)$$

We could calculate the image convolution size based on Eqs. (16), (17), (18), and (19), as shown in Eqs. (20) and (21). In this case, the variable of H'' and W'' indicate the height and width of the image convolution results.

$$H'' = \left\lfloor \frac{H-H'+P_h^-+P_h^+}{S} \right\rfloor + 1 \quad (20)$$

$$W'' = \left\lfloor \frac{W-H'+P_h^-+P_h^+}{S} \right\rfloor + 1 \quad (21)$$

The following process calculates the activation function using a rectified linear unit (ReLU). In this function, all of the element values $\mathcal{Y}_{i',j',D'}$ will be replaced with 0 when the value is less than 0. Otherwise, the value does not change, as described in the following equation

$$f(\mathcal{Y}_{i',j',D'}) = \begin{cases} 0, & \mathcal{Y}_{i',j',D'} < 0 \\ \mathcal{Y}_{i',j',D'}, & \mathcal{Y}_{i',j',D'} \geq 0 \end{cases} \quad (22)$$

Through pooling, we used the ReLU function's result to reduce the dimension of the convolution result. In this case, we also apply two constraints to conduct the pooling operation: size and stride. The pooling result is well-known as a feature map. Using two regulations will determine how big the size of the results of the feature map. We replace the variable of $\mathcal{Y}_{i',j',D'}$ using $\mathbb{x}$ for ease of writing and understanding. Therefore, we could write a pooling mathematical in Eqs. (23) and (24)

$$\mathcal{Y}_{i''j''D} = \max_{1\leq i'\leq H''',1\leq j'\leq W'''} \mathbb{x}_{i''+i'-1}, j''+j'-1, D \quad (23)$$

$$\mathcal{Y}_{i''j''D} = \frac{1}{W'''\times H'''} \times \sum_{1\leq i'\leq H''',1\leq j'\leq W'''} \mathbb{x}_{i''+i'-1}, j''+j'-1, D \quad (24)$$

Based on Eqs. (23) and (24), we could calculate the size of the maximum pooling result as follows.

$$\mathbb{P}_w = \left(\frac{(W''-W''')}{S}\right) + 1 \quad (25)$$

$$\mathbb{P}_h = \left(\frac{(H''-H''')}{S}\right) + 1 \quad (26)$$

The size of the pooling dimension result is $(\mathbb{P}_w, \mathbb{P}_h, D)$ as shown, the value of $\mathbb{P}_w$ and $\mathbb{P}_h$ in Eqs. (25) and (26). Eqs. (23) and (24) show that the feature map result will be replaced the one-dimensional, flattening model. Furthermore, the flattening results are supposed as the input layer. The input layer will be multiplied by the random weight $(\mathcal{w}_{ij})$ of the hidden layer and added by bias initials $(\mathcal{B}_j)$ of the hidden layer. The results will be utilized as input on the activation function layer.

$$y_j = \Sigma_{i=1}^{n}(\mathcal{Y}_{i''j''D})_i \times \mathcal{w}_{ij} + \mathcal{B}_j \quad (27)$$

$$\mathscr{y}_j = \sigma_{yj}(y_j) \quad (28)$$

As seen in Eq. (27), the hidden layer results will be input in the activation function as in Eq. (28) and Fig. 4. Moreover, the equation result Eq. (27) is used to input in the output layer as follows:

Received: June 6, 2023. Revised: June 29, 2023.

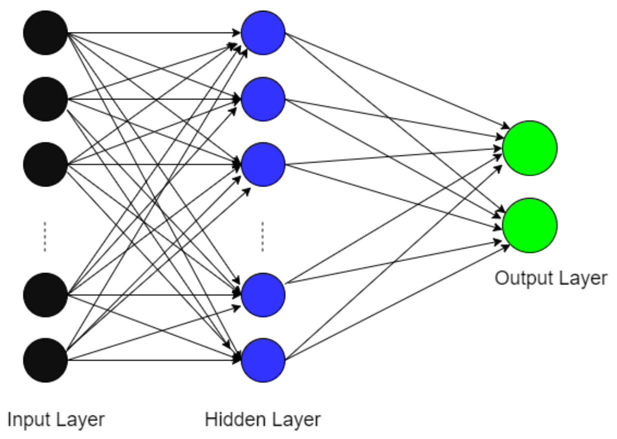

Figure. 4 Representation of Eqs. (27), (28), and (29).

$$net_k = \Sigma_{j=1}^{m}\left(y_j \times w_{jk} + \mathcal{B}_k\right) \quad (29)$$

We can compute the error of the training and the testing. Log loss error can be calculated by using the following equation.

$$\boldsymbol{Loss = -p(x)\, In\, q(x) - (1-p(x)) In (1-q(x))} \quad (30)$$

$\boldsymbol{p(x)}$ and $\boldsymbol{q(x)}$ represent the actual and prediction values of the model.

2.8 Accuracy calculation

When compared to the overall number of forecasts, the accuracy of a prediction may be understood by looking at the ratio of the number of correct predictions to the total number of predictions. The classification results that correspond to the accurate prediction are those in which the predicted values are identical to those that occurred, as shown in the following equation.

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} \quad (31)$$

True positive (TP) describes actual and prediction conditions that are positive. In contrast, the actual and prediction have negative values, well-known as true negative (TP). If the actual value is negative and it is predicted as a positive value, it is called a false positive (FP). On the contrary, if the prediction result is negative and the actual value is positive, then it is categorized as a false negative (FN).

In addition, we compute both precision and recall to evaluate our inquiry's correctness and comprehensiveness. When positive predictions can be made, precision refers to the amount of those

Table 2. Confusion matrix

Actual	Prediction	
	Positive	Negative
Positive	True Positive (*TP*)	False Negative (*FN*)
Negative	False Positive (*FP*)	True Negative (*TN*)

Table 3. Our experimental scenarios

	K	Commutative Hypercomplex Values					
		A[+1,-1]			A[-1,+1]		
K-Fold Cross Validation	1	10^{-5}	10^{-6}	10^{-7}	10^{-5}	10^{-6}	10^{-7}
	2	10^{-5}	10^{-6}	10^{-7}	10^{-5}	10^{-6}	10^{-7}
	3	10^{-5}	10^{-6}	10^{-7}	10^{-5}	10^{-6}	10^{-7}
	4	10^{-5}	10^{-6}	10^{-7}	10^{-5}	10^{-6}	10^{-7}
	5	10^{-5}	10^{-6}	10^{-7}	10^{-5}	10^{-6}	10^{-7}

positive predictions that can be made. It is a crucial statistic utilized to investigate the classification results, precisely the rate of false positives. As the following equation demonstrates, the proposed method's precision improves whenever fewer false positives are found.

$$Precision = \frac{TP}{TP+FP} \quad (32)$$

$$Recall = \frac{TP}{TP+FN} \quad (33)$$

Similarly, the last measurement is a well-known recall, and it can be defined as a comparison between the true positive and the summation of actual's positives as Eq. (33). We can easy representation Eqs. (31), (32), and (33) as demonstrated in the Confusion matrix model Table 2.

3. Experimental and discussion

We used 260 ALL images, of which 130 were considered normal, and the remaining images were supposed to be Leukemia. We split the data sets into eighty and twenty percent for the training and testing. Moreover, to integrate training datasets, we enrich training datasets using the augmentation process as Eqs. (1) until (7). We have selected the learning rate values for the commutative hypercomplex, which are 0.0001=10^{-5}, 0.00001=10^{-6}, and 0.000001=10^{-7} with 100 epochs.

3.1 Experimental results for A[+1, -1] commutative of hypercomplex value

We have completed the experiment to classify ALL image datasets for training and validation. We employ K-Fold cross validation to obtain the best

Received: June 6, 2023. Revised: June 29, 2023.

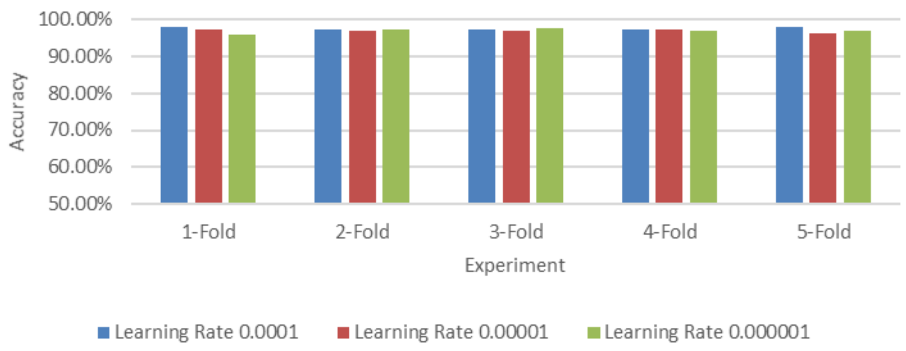

Figure. 5 Summary of average accuracy using A[+1, -1] commutative hypercomplex for the training

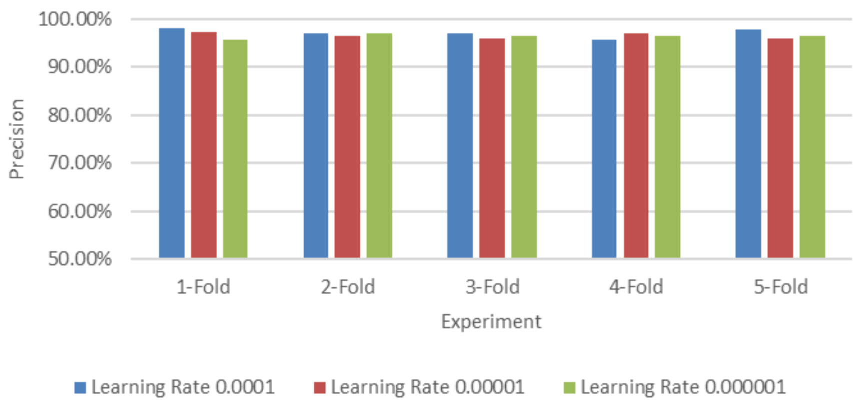

Figure. 6 Summary of average precision using A[+1, -1] commutative hypercomplex for the training

model and validate our image testing set. We have used 0.0001, 0.00001, and 0.000001 as the learning rate and one hundred epochs. The results show that we have obtained the best average model for the A[+1, -1] commutative of hypercomplex value is produced by the first fold using 0.0001 learning rate, which is 98.01%, 98.02%, and 98.35% for the accuracy in Fig. 5, precision in Fig. 6, and recall in Fig. 7, respectively. We have found 38 times our training was delivered 100% accurately. In addition, we have also evaluated precision and recall. We have obtained 44 and 43 times for precision and recall 100% correctly, as demonstrated in Fig. 8. Figs. 5, 6, and 7 display the resume results for the first until the last fold. We can investigate similar results, meaning our proposed method has delivered stable results for accuracy, precision, and recall.

Here, we demonstrated the accuracy, precision, and recall, followed by error for the first fold and 0.0001 learning rate, as shown in Fig. 8. We can investigate whether the accuracy, precision, and recall overlap, and it shows that the training accuracy, precision, and recall are stable.

In addition, we further investigate related to the error result of the training for the first fold from the first to the one-hundredth expressing that the error results are convergence, which results in continuously moving toward the zero value, as demonstrated in Fig. 9.

We have evaluated the best training model produced by the first fold and 0.0001 learning rates. The results show that our model delivered the average accuracy, precision, and recall for the testing validation are 95.38%, 95.50%, and 94.27%, respectively. While the maximum result for accuracy is 98.68%, precision is 100%, and recall is 100%, as demonstrated in Fig. 10. However, we have registered the average standard deviation for the training and testing validation are 0.08 and 0.06. It proved that our proposed method could produce stable accuracy, even though the validation error obtained fluctuates in the 1st to 26th and 69th to 85th epochs, as shown in Fig. 11. All models have tried to improve the intelligence to gain the minimum error.

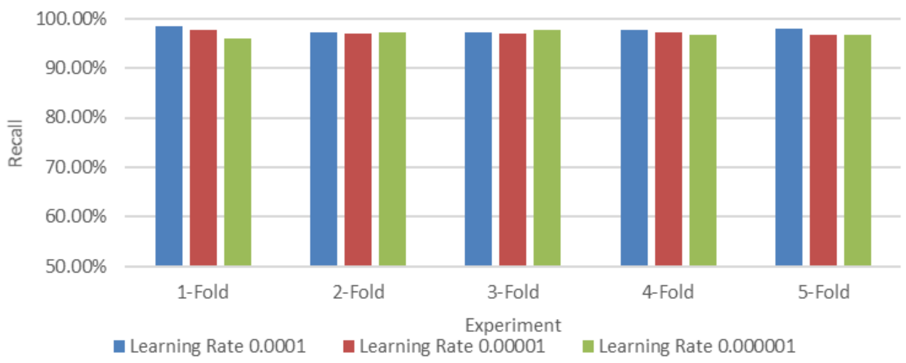

Figure 7. Summary of average recall using A[+1, -1] commutative hypercomplex for the training

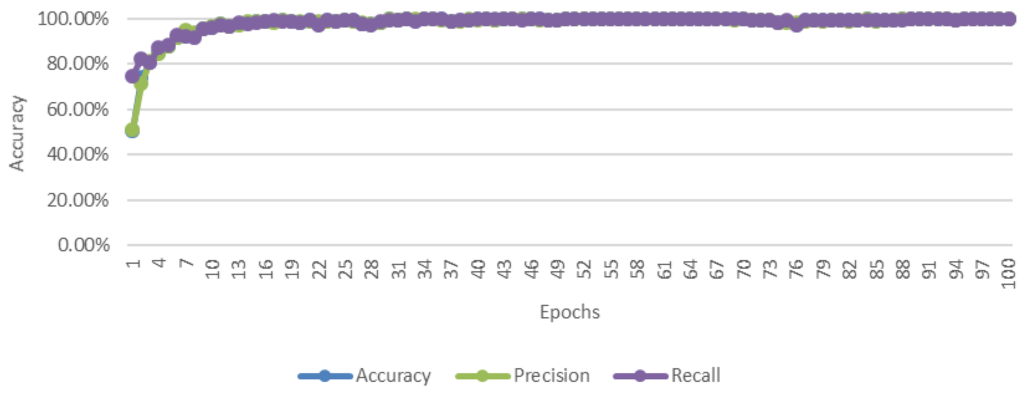

Figure. 8 Accuracy, precision, and recall using A[+1, -1] commutative hypercomplex for the training using 0.0001 learning rate and 100 epochs

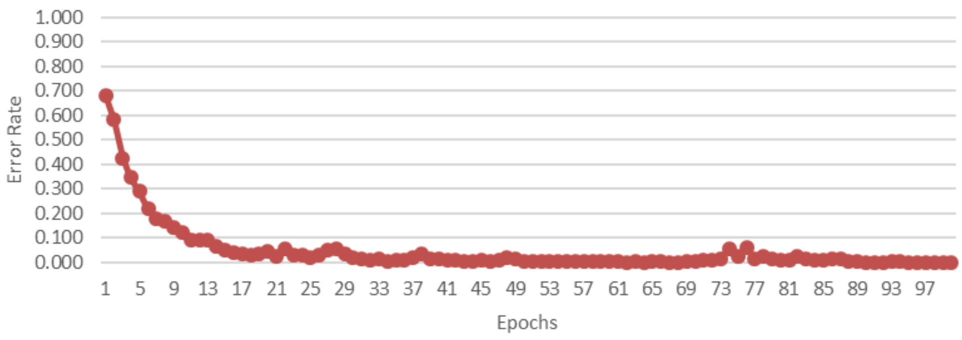

Figure. 9 Error results using A[+1, -1] commutative hypercomplex for the training using 0.0001 learning rate and 100 epochs

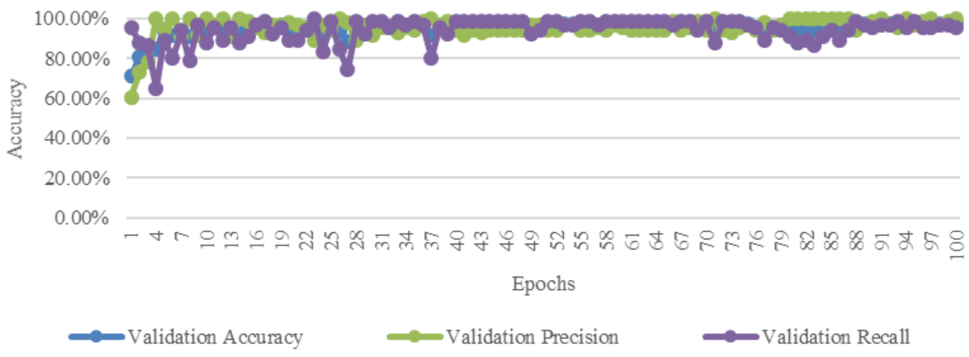

Figure 10. Accuracy, precision, and recall using A[+1, -1] commutative hypercomplex for the testing validation using 0.0001 learning rate and 100 epochs.

Received: June 6, 2023. Revised: June 29, 2023.

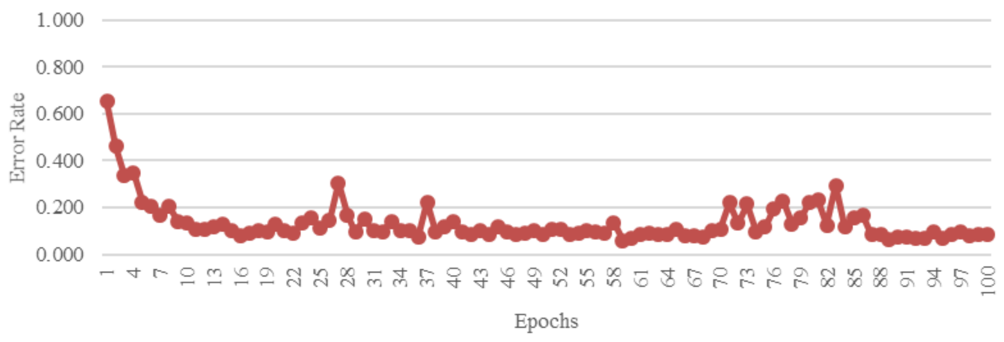

Figure. 11 Error results using A[+1, -1] commutative hypercomplex for the testing validation using 0.0001 learning rate and 100 epochs

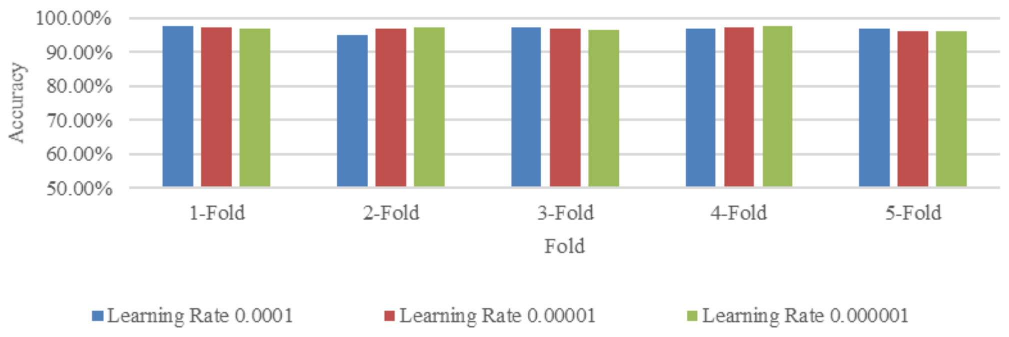

Figure. 12 Summary of average accuracy using A[-1, +1] commutative hypercomplex for the training

3.2 Experimental results for A[-1, +1] commutative of hypercomplex value

In addition, we have also conducted the training and the testing validation using the A[-1, +1] commutative hypercomplex value. We employed the 0.0001, 0.00001, and 0.000001 learning rates. We hired 100 epochs to learn and search for the best model for each cross-validation. Based on the experiment results, the first fold cross-validation has delivered the best model for the A[-1, +1] commutative of hypercomplex value. Furthermore, the accuracy, precision, and recall are practically similar, with no significant difference, as seen in Figs. 12, 13, and 14 for accuracy, precision, and recall. However, it is no significant difference among k-fold cross-validation from the 1st to the 5th. It shows that the architecture model is stable.

Fig. 12 clearly shows that the accuracy tends to weaken as the learning rate decreases (from 0.0001 to 0.000001). It could be because the model converges more slowly or becomes trapped in local optima when the learning rate is low. There is a difference in the percentage of correct folds. For instance, the model achieves its best accuracy across all three learning rates in the first fold. Furthermore, it is essential to remember that the accuracy variations between the various learning rates and folds are less than 2%. It indicates that the learning rate may not significantly affect the performance model in this setting.

Fig. 13 demonstrates that the highest precision percentages are typically accomplished with a learning rate of 0.0001, while the lowest precision percentages are typically performed with a learning rate of 0.000001. It can be seen by comparing the two, and it hints that a more sedate learning rate of 0.00001 or 0.0001 would work better for this specific model. There is some difference in the precision percentages across the different folds of the paper. For instance, the model obtains the maximum precision in the 1st and 4th folds with a learning rate of 0.0001, but in the second fold and third folds, the model achieves the highest precision with a learning rate of 0.0001. However, if we investigate more comprehensively, the results for each experiment are not almost different.

Fig. 14 displays that the maximum recall percentages are achieved with a learning rate of 0.00001 over all folds. The current learning rate may be ideal for this model and data set. There is some discrepancy in the accuracy percentages between the various folds, though it is insignificant. With a learning rate 0.0001, the 1st fold achieves the highest recall, whereas the fifth fold obtains the highest recall percentage with a 0.000001 learning rate. It indicates that the result depends on the fold utilized. However,

Received: June 6, 2023. Revised: June 29, 2023.

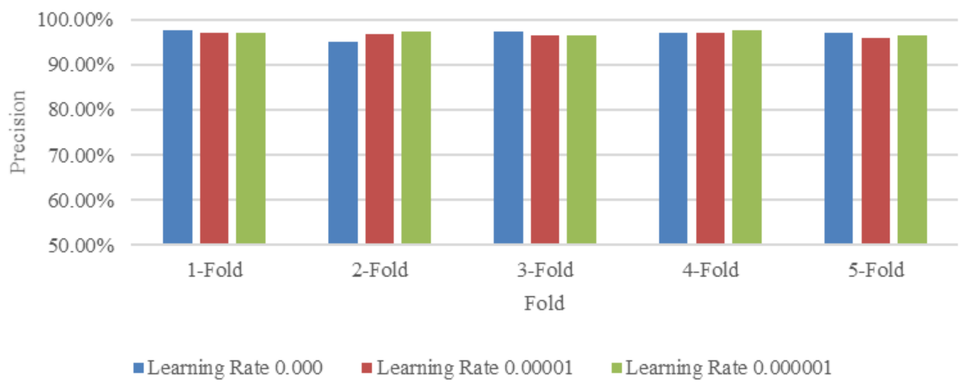

Figure. 13 Summary of average precision using A[-1, +1] commutative hypercomplex for the training

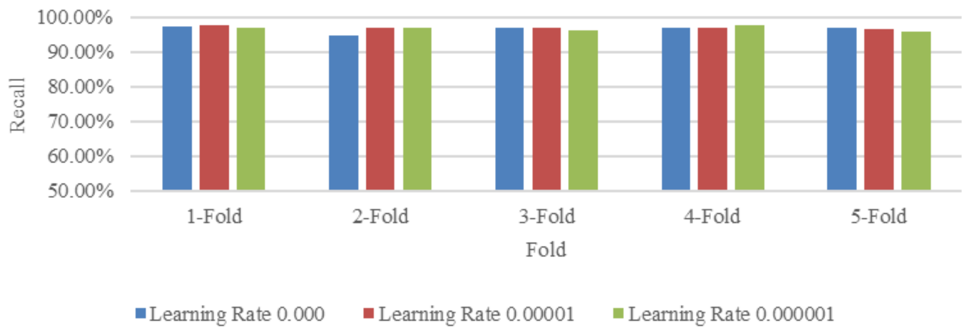

Figure 14. Summary of average recall result using A[-1, +1] commutative hypercomplex for the training

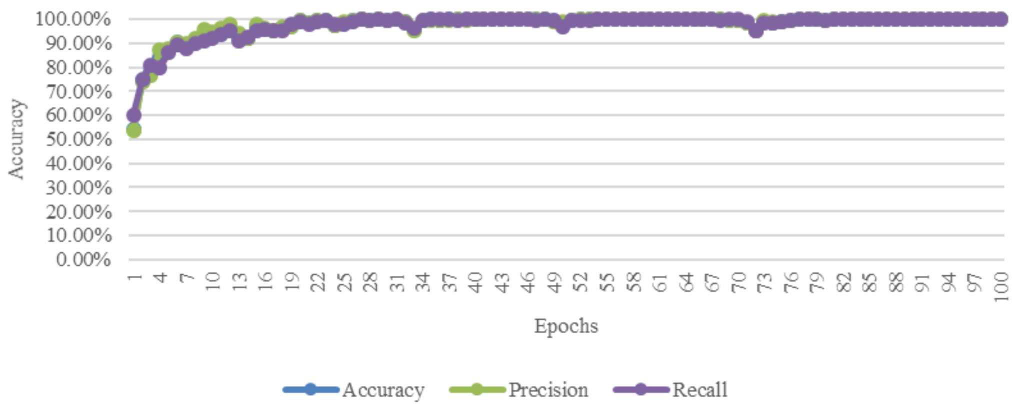

Figure. 15 Accuracy, precision, and recall using A[-1, +1] commutative hypercomplex using 0.0001 learning rate and 100 epochs

the performance models are also affected by the learning rate used. We can check some consistency between the learning rates and folds in detail, even though the recall percentage changes are always tiny.

In addition, we spilled in detail the accuracy, precision, and recall using a 0.0001 learning rate and all-fold cross-validation. Fig. 15 displays the experimental accuracy, precision, and recall results. The results show that the accuracy, precision, and recall increase according to epoch rise, and the results have significantly grown before the 26th epoch. However, the performance of our proposed method is stable after the 26th epoch. If we have observed in detail, there are three decreasing points on the 32, 52, and 72 epochs for accuracy, precision, and recall, even though it is insignificant and stable after passing this point. Therefore, the error of the training delivered raised on the same location, as displayed in Fig. 16.

The fact that the error values have decreased implies that the model is becoming better and more accurate as more epochs pass. The presented data indicate that the model is improving and optimizing over time, as displayed in Fig. 16. However, it is essential to remember that the pace at which error values decrease appears to change during the training

Received: June 6, 2023. Revised: June 29, 2023.

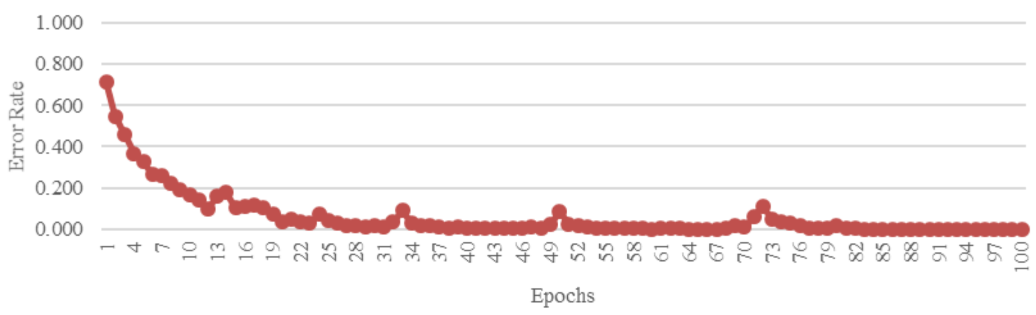

Figure. 16 Error results using A[-1, +1] commutative hypercomplex using 0.0001 learning rate and 100 epochs

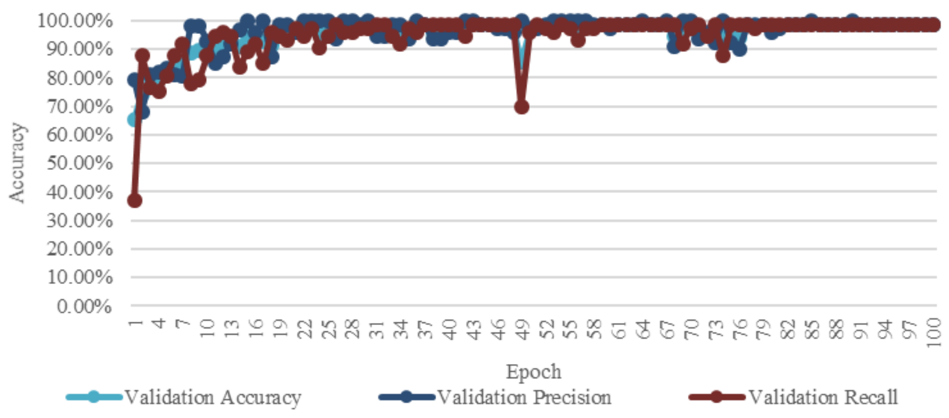

Figure. 17 Accuracy, precision, and recall using A[-1, +1] commutative hypercomplex for the testing validation using 0.0001 learning rate and 100 epochs

process, with some epochs displaying higher gains than others. It is an essential point to keep in mind. This information may help determine which epochs should be the primary attention point when analyzing the model performance to optimize the training process.

We have completed an experiment to evaluate the image validation testing using the best model. We employed a 0.0001 learning rate and 100 epochs. The results show that from the beginning to the 9th epoch displayed that the model has learned fastly, and it can be demonstrated that the validation accuracy, precision, and recall increase significantly. Furthermore, the model moves consistently until the last epoch, even though some points have occurred in the validation recall. It indicated that an overfitting condition appears for the validation recall on the 49th epoch, while accuracy and precision have consistently delivered results for the validation. However, the metrics exhibit certain inconsistencies across different epochs, a phenomenon ascribed to various factors, including the stochastic initialization of the model's parameters, the choice of hyperparameters, and the stochasticity of the training data. From the results, we have inferred that the maximum results for the validation accuracy, precision, and recall are 99.34%, 100%, and 98.63%, respectively, while the average validation accuracy, precision, and recall delivered 95.62%, 96.22%, and 94.74%. Overall, the current analysis of the metrics indicates that the model exhibits a satisfactory performance on the validation set. Moreover, as the training proceeds, the model appears to acquire knowledge from the dataset. It is imperative to continually monitor the metrics to ascertain that the model is not exhibiting tendencies of overfitting or underfitting concerning the underlying data.

Fig. 18 demonstrates the error results of the testing validation using A[-1, +1] commutative hypercomplex with a 0.0001 learning rate and 100 epochs. The results show that the error goes down from the beginning at the onset and concludes at the 48th epoch. The model has adapted to the training results to evaluate the validation testing set. However, overfitting occurred on the 49th epoch, where the error rate increased significantly. Fortunately, this condition can be improved until the end of an iteration. The error also can be decreased considerably on the 50th epoch, though the loss still increases on the certainty points, but finally, the error decrease until near zero. If we deeply observed Fig. 17, we obtained information that considerably decreases the recall on the same epoch. Overfitting may manifest in a machine learning model that is

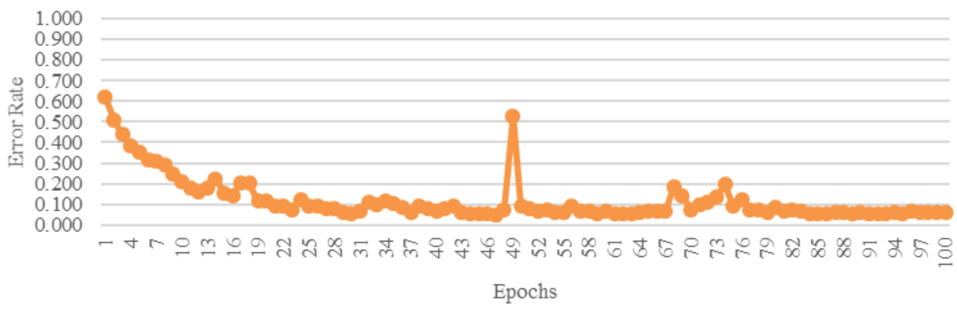

Figure. 18 Error results using A[-1, +1] commutative hypercomplex for the validation testing using 0.0001 learning rate and 100 epochs

excessively intricate and consequently encapsulates the excessive noise, or random changes, embedded within the training dataset instead of discerning the fundamental pattern that applies to novel data. This phenomenon may lead to a developed model that exhibits substantial aptitude when evaluated on the training set yet a poor performance outcome when exposed to new, unseen data. It is attributable to the model having retained the training data explicitly rather than acquiring a more broadly applicable pattern, reflecting an inadequate degree of generalization.

3.3. Comparison results and discussion

The training model's building is always the random number for the first epoch. If the initial random numbers used are suitable for building the model, the model will deliver good results for accuracy, precision, and recall. Otherwise, the model will send bad results. We have conducted several experiments with different scenarios divided into two parts. Firstly, we build the model based on the training sets. In addition, the best model is used to evaluate the validation set of the samples. Secondly, we select the best model to evaluate the validation sets. Our proposed method's best training model selection is based on the average accuracy of each fold for all epochs.

Moreover, we applied the highest average accuracy to evaluate the image validation image sets. We guarantee that the validation image sets have zero intersection with the training image sets. It means that we do not employ the image training sets for the validation image sets.

We can see that the model for the training sets has delivered stable results after the tenth epoch. If we ignore the first till the ninth epochs, our proposed method produces over 99% average accuracy for all scenarios. The lowest accuracy for training and validation processes usually appears before the tenth epoch. We have noticed the experimental results that our proposed method has delivered 100% accurate results for each fold cross-validation. It shows that our proposed method can be used as a reference model to develop a Leukemia detection system.

Our proposed method's best training model selection is based on the average accuracy of each fold for all epochs. Moreover, we applied the highest average accuracy to evaluate the image validation image sets. We guarantee that the image validation sets have zero intersection with the training image sets. It means that we do not employ the image training sets for the validation image sets.

Furthermore, we compare our proposed method results to the others, such as K-nearest neighbour (KNN) [7], support vector machine-radial basis function (SVM-RBF) [7], support vector machine-linear (SVM-L) [7], support vector machine-polynomial (SVM-P) [7], Naïve Bayes Gaussian [7], Naïve Bayes complement [7], decision tree [7], Muti distance model [3], colour hybrid modelling pyramid model [20], and hypercomplex model [18] as shown in Fig. 19. We compare the best training model's average accuracy to the others. In addition, we have implemented the best model of our proposed method to evaluate the validation image sets. We calculate the average accuracy of the best model, as shown in Fig. 19.

Fig. 19 shows that our proposed method A[+1, -1] Commutative Hypercomplex model outperformed KNN [7], SVM-RBF [7], SVM-L [7], SVM-P [7], Naïve Bayes Gaussian [7], Naïve Bayes complement [7], decision tree [7], colour hybrid modelling [20]. Moreover, the Pyramid model [20] and hypercomplex model [18] have produced better accuracy than our proposed method. However, It is not better than the multi distance model [3], Pyramid model [20], and hypercomplex model [18]. In addition, our proposed method A[-1, +1] commutative hypercomplex model has exceeded the accuracy of A[+1, -1] commutative hypercomplex model, the multi distance model [3], and the

Received: June 6, 2023. Revised: June 29, 2023.

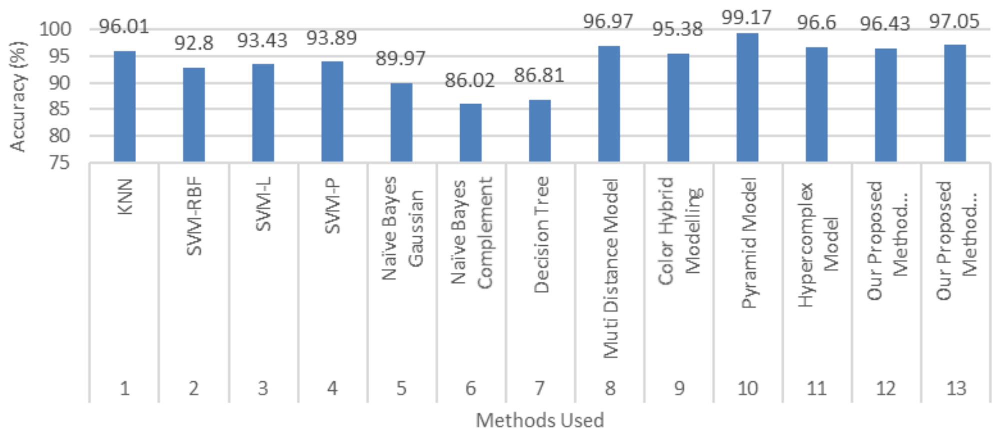

Figure. 19 The comparison results of the average accuracy of our proposed method to the others

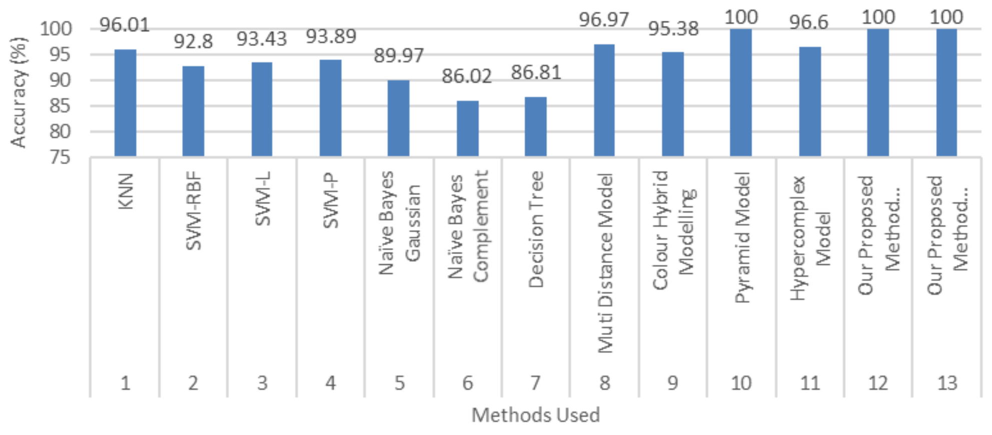

Figure. 20 The comparison results of maximum accuracy of our proposed method to the others

Hypercomplex model [18]. However, it is not better than Pyramid model [20].

We also compare the maximum accuracy of our proposed method to the others. Our proposed method is better than the others, except the Pyramid model [20]. The pyramid model has generated the same accuracy as our proposed method for A[+1, -1] and A[-1,+1] commutative hypercomplex models. It is accuracy at 100%, as demonstrated in Fig. 20.

4. Conclusion and future research

Our experimental results have delivered the best model for the training using A[+1, -1] and A[-1, +1] commutative of hypercomplex. Our proposed method has yielded the best average accuracy at 96.43% for A[+1,-1] and 97.05 for A[-1,+1] commutative hypercomplex. In addition, our proposed method also produced maximum accuracy at 100%. Maximum accuracy has occurred several times during the experiment for training and validation processes. Theoretically, using a learning rate has affected the accuracy and error rates, and a lower learning rate has delivered better accuracy than a higher one. Our proposed method also produced maximum precision and recall for all models, 100% for A[+1, -1] and A[-1,+1]. In other words, our proposed method can be used as a reference to implement in the real world, especially in the medical field.

However, our proposed method shows that the use of different learning rates slightly differs in accuracy for training and validation processes. According to comparison results, our approach has defeated K-Nearest Neighbor, Support Vector Machine-Radial Basis Function, Support Vector Machine-Linear, Support Vector Machine-Polynomial, Nave Bayes Gaussian, Nave Bayes Complement, Decision Tree, and Color Hybrid Modeling.

We will improve our convolutional neural network architecture for further research to obtain better results. In addition, we also employ other image datasets with a more considerable number of images to reduce computation time during the training process due to the utilized massive image datasets.

Conflicts of interest

We would like to declare no conflicts of interest.

Author contributions

We have completed our research. The Directorate of Higher Education - Ministry of Research, technology, and higher education fully fund our research for supporting our research under the fundamental research funding scheme 2023.

Finally, our research has been finished by our team. Each person has completed their work as follows: Conceptualization: Arif Muntasa and Rima Tri Wahyuningrum; Methodology: Arif Muntasa and Rima Tri Wahyuningrum; Literature review: Arif Muntasa; Image Data preparation: Durotun Nafisah; Architecture model: Arif Muntasa, Rima Tri Wahyuningrum, and Durotun Nafisah; Algorithm Designer: Arif Muntasa, Rima Tri Wahyuningrum, and Durotun Nafisah; Software Implementation: Durotun Nafisah; Experimental designer: Arif Muntasa; Result analysis: Rima Tri Wahyuningrum, and Arif Muntasa; Investigation: Arif Muntasa and Rima Tri Wahyuningrum; Writing original draft preparation: Arif Muntasa; Writing review and editing: Rima Tri Wahyuningrum; Supervision: Arif Muntasa; Project administration: Rima Tri Wahyuningrum.

Acknowledgment

We are truly honored and thankful to be a beneficiary of the full funding for research in 2023. We express our deepest gratitude to the Ministry of Education, Culture, Research, and Technology for its extraordinary support and commitment to providing full funding for research projects and valuable contribution to promoting fundamental research through full funding support.

References

[1] J. Rawat, A. Singh, H. S. Bhadauria, and J. Virmani, “Computer Aided Diagnostic System for Detection of Leukemia Using Microscopic Images”, in *Procedia Computer Science*, Vol. 70, pp. 748–756, 2015, doi: 10.1016/j.procs.2015.10.113.

[2] S. Mishra, B. Majhi, P. K. Sa, and L. Sharma, “Biomedical Signal Processing and Control Gray level co-occurrence matrix and random forest based acute lymphoblastic leukemia detection”, *Biomed. Signal Process. Control*, Vol. 33, pp. 272–280, 2017, doi: 10.1016/j.bspc.2016.11.021.

[3] A. Muntasa and M. Yusuf, “Multi Distance And Angle Models Of The Gray Level Co-Occurrence Matrix(Glcm) To Extract The Acute Lymphoblastic Leukemia (All) Images”, *Int. J. Intell. Eng. Syst.*, Vol. 14, No. 6, pp. 357–368, 2021, doi: 10.22266/ijies2021.1231.32.

[4] S. Mishra, B. Majhi, and P. K. Sa, “Biomedical Signal Processing and Control Texture feature based classification on microscopic blood smear for acute lymphoblastic leukemia detection”, *Biomed. Signal Process. Control*, Vol. 47, pp. 303–311, 2019, doi: 10.1016/j.bspc.2018.08.012.

[5] A. Muntasa and M. Yusuf, “Color-based hybrid modeling to classify the acute lymphoblastic leukemia”, *Int. J. Intell. Eng. Syst.*, Vol. 13, No. 4, pp. 408–422, 2020, doi: 10.22266/ijies2020.0831.36.

[6] J. Rawat, A. Singh, H. S. Bhadauria, and J. Virmani, “Computer Aided Diagnostic System for Detection of Leukemia”, *Procedia - Procedia Comput. Sci.*, Vol. 70, pp. 748–756, 2015, doi: 10.1016/j.procs.2015.10.113.

[7] A. M. Abdeldaim, A. T. Sahlol, M. Elhoseny, and A. E. Hassanien, “Computer-Aided Acute Lymphoblastic Leukemia Diagnosis System Based on Image Analysis”, *Advances in Soft Computing and Machine Learning in Image Processing*, A. E. Hassanien and D. A. Oliva, Eds. Cham: Springer International Publishing, pp. 131–147, 2018.

[8] J. Laosai and K. Chamnongthai, “Biomedical Signal Processing and Control Classification of acute leukemia using medical-knowledge-based morphology and CD marker”, *Biomed. Signal Process. Control*, Vol. 44, pp. 127–137, 2018, doi: 10.1016/j.bspc.2018.01.020.

[9] J. Laosai and K. Chamnongthai, “Classification of acute leukemia using medical-knowledge-based morphology and CD marker”, *Biomed. Signal Process. Control*, Vol. 44, pp. 127–137, Jul. 2018, doi: 10.1016/j.bspc.2018.01.020.

[10] J. Laosai and K. Chamnongthai, “Classification of acute leukemia using medical-knowledge-based morphology and CD marker”, *Biomed. Signal Process. Control*, Vol. 44, pp. 127–137, Jul. 2018.

[11] L. Putzu and C. D. Ruberto, “White Blood Cells Identification and Classification from Leukemic Blood Image”, In: *Proc. of International Work-Conference on Bioinformatics and Biomedical Engineering*, pp. 18–20. , 2013

[12] L. Putzu and C. D. Ruberto, “White Blood Cells Identification and Counting from Microscopic Blood Image”, *Int. J. Medical, Heal. Biomed. Bioeng. Pharm. Eng.*, Vol. 7, No. 1, pp. 15–22, 2013.

[13] L. Putzu, G. Caocci, and C. D. Ruberto, "Leucocyte classification for leukaemia detection using image processing techniques", *Artif. Intell. Med.*, Vol. 62, No. 3, pp. 179–191, Nov. 2014, doi: 10.1016/j.artmed.2014.09.002.

[14] J. E. M. D. Oliveira and D. O. Dantas, "Classification of normal versus leukemic cells with data augmentation and convolutional neural networks", In: *VISIGRAPP 2021 – Proc. of the 16th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, pp. 685–692, 2021.

[15] L. H. S. Vogado, R. M. S. Veras, F. H. D. Araujo, R. R. V. Silva, and R. T. Aires, "Engineering Applications of Artificial Intelligence Leukemia diagnosis in blood slides using transfer learning in CNNs and SVM for classification", *Eng. Appl. Artif. Intell.*, Vol. 72, No. October 2017, pp. 415–422, 2018, doi: 10.1016/j.engappai.2018.04.024.

[16] L. H. S. Vogado, R. D. M. S. Veras, A. R. Andrade, F. H. D. D. Araujo, R. R. V. Silva, and K. R. T. Aires, "Diagnosing Leukemia in Blood Smear Images Using an Ensemble of Classifiers and Pre-Trained Convolutional Neural Networks", In: *Proc. of 2017 30th SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI)*, pp. 367–373, 2017, doi: 10.1109/SIBGRAPI.2017.55.

[17] L. H. S. Vogado, R. M. S. Veras, F. H. D. Araujo, R. R. V. Silva, and K. R. T. Aires, "Leukemia diagnosis in blood slides using transfer learning in CNNs and SVM for classification", *Eng. Appl. Artif. Intell.*, Vol. 72, pp. 415–422, Jun. 2018.

[18] V. Guilherme, M. E. Valle, "Acute Lymphoblastic Leukemia Detection Using Hypercomplex-Valued Convolutional Neural Networks", In: *Proc. of International Joint Conference on Neural Networks (IJCNN 2022)*, pp. 1–8, 2022.

[19] R. D. Labati, V. Piuri and F. Scotti, "All-IDB: The acute lymphoblastic leukemia image database for image processing", In: *Proc. of 2011 18th IEEE International Conference on Image Processing*, Brussels, Belgium, pp. 2045-2048, 2011, doi: 10.1109/ICIP.2011.6115881.

[20] A. Muntasa, A. Motwakel, R. T. Wahyuningrum, Z. Tuzzahra, M. Yusuf, and W. F. Mahmudi, "A Pyramid Model of Convolutional Neural Network to Classify Acute Lymphoblastic Leukemia Images", *Int. J. Intell. Eng. Syst.*, Vol. 15, No. 6, pp. 576–588, 2022, doi: 10.22266/ijies2022.1231.51.