Journal of Digital Imaging (2022) 35:623–637
https://doi.org/10.1007/s10278-022-00600-3

ORIGINAL PAPER

Optimizing a Deep Residual Neural Network with Genetic Algorithm for Acute Lymphoblastic Leukemia Classification

Larissa Ferreira Rodrigues[1] · André Ricardo Backes[1] · Bruno Augusto Nassif Travençolo[1] · Gina Maira Barbosa de Oliveira[1]

Received: 22 March 2021 / Revised: 21 January 2022 / Accepted: 28 January 2022 / Published online: 23 February 2022

Abstract
Acute lymphoblastic leukemia (ALL) is the most common childhood cancer worldwide, and it is characterized by the production of immature malignant cells in the bone marrow. Computer vision techniques provide automated analysis that can help specialists diagnose this disease. Microscopy image analysis is the most economical method for the initial screening of patients with ALL, but this task is subjective and time-consuming. In this study, we propose a hybrid model using a genetic algorithm (GA) and a residual convolutional neural network (CNN), ResNet-50V2, to predict ALL using microscopy images available in ALL-IDB dataset. However, accurate prediction requires suitable hyperparameters setup, and tuning these values manually still poses challenges. Hence, this paper uses GA to find the best hyperparameters that lead to the highest accuracy rate in the models. Also, we compare the performance of GA hyperparameter optimization with Random Search and Bayesian optimization methods. The results show that GA optimization improves the accuracy of the classifier, obtaining 98.46% in terms of accuracy. Additionally, our approach sheds new perspectives on identifying leukemia based on computer vision strategies, which could be an alternative for applications in a real-world scenario.

Keywords Leukemia classification · Convolutional neural networks · Genetic algorithm · Hyperparameter optimization · Fine-tuning

Introduction

Acute lymphoblastic leukemia (ALL) is a type of cancer caused by immature lymphocytes in bone marrow [1]. It is the most common childhood cancer worldwide and accounts for 80% of all childhood leukemia [2–4]. ALL can be diagnosed through a variety of tests, such as physical examinations, blood tests, blood counts, myelogram, lumbar punctures, and bone marrow biopsies [5].

Larissa Ferreira Rodrigues
larissarodrigues@ufu.br

André Ricardo Backes
arbackes@yahoo.com.br

Bruno Augusto Nassif Travençolo
travencolo@ufu.br

Gina Maira Barbosa de Oliveira
gina@ufu.br

1 Faculty of Computing (FACOM), Federal University of Uberlândia (UFU), Uberlândia, MG, Brazil

Visual analysis in microscopy images is the most economical method for the initial screening of patients with ALL. However, it is a very subjective and time-consuming task [1]. Leukocytes are detected by their dark purple-like appearance, but the analysis and further processing become very complicated due to their variability in shape and texture. Also, significant changes in the morphology of the cells are found in severe cases of the disease.

To overcome these limitations, computer-aided diagnosis systems based on image processing and machine learning techniques are essential and widely applied in several fields of medicine [6–10]. Moreover, these techniques are financially attractive, especially for developing countries [11].

Following the advances of computational resources, deep convolutional neural networks (CNNs) have been significantly outperforming approaches based on handcrafted features. This strategy provides an automatic feature extraction from input images and demonstrates effective results in visual recognition tasks in many areas [12, 13].

In this study, we explore the classification of ALL using microscopy images and deep learning. Our main goal is to improve the identification rate among immature lymphocytes and healthy lymphocytes, thus helping in the diagnosis of ALL in the early stages. Our deep learning approach used a ResNet-50 V2 [14] CNN architecture, and we evaluated its performance using a *k*-fold cross-validation procedure over the training set and validated the overall results in using the and test sets.

Finding the optimal hyperparameters to train a deep CNN is crucial because there is no optimum method for selecting hyperparameters. In machine learning, for example, grid search (equivalent to brute force), random search, Bayesian optimization, and GA are common approaches used for hyperparameter optimization [15–18]. However, the computational cost of grid search is too high, while random and Bayesian approaches are limited to the search space distribution [19].

To address this challenge, we innovate with a method to find a suitable setup for hyperparameter optimization using genetic algorithm (GA). Hyperparameters are a set of variables to be tuned before applying the learning algorithm to the dataset. These hyperparameters directly affect the learning speed, the convergence of the cost function, and the classification performance [20].

Our results suggest that hyperparameter optimization combined with fine-tuning training tends to be the best performing strategy. Our results demonstrate the suitability and performance score of hyperparameter optimization based on GA, and the best result achieved an accuracy of 98.46% in ALL classification.

Moreover, as far as we know, our result is the best obtained for ALL classification in microscopy images using the dataset evaluated in this study. We believe that our proposed method can contribute to future research intended to help healthcare workers to identify leukemia and manage patients' conditions.

The remaining of this paper is organized as follows: "Related Work" surveys related work. "Materials and Methods" describes the material and methods. "Experimental Protocol" presents the experimental procedure. "Results and Discussion" presents and discusses the results. Finally, conclusion and future work are presented in "Conclusions".

Related Work

Leukemia identification using microscopy images is a field of intense research, with many approaches being developed over the years. Piuri and Scotti [21] were among the first to propose an automatic system for classifying leukocytes from other blood components such as red blood cells, platelets, and plasma in microscopy images. After, Scotti [22] proposed another system that uses shape features to identify ALL from microscopic images. According to their experiments, morphological features allowed the white blood characterization.

Several studies have been proposed for classifying ALL in microscopy images with hand-crafted feature extraction (i.e., using a non-automated user-based process). Mohapatra et al. [23] proposed leukocytes classification by using a support vector machine (SVM) with shape, texture, and color features. After, [24] proposed an approach based on shape and color features and an ensemble of classifiers.

Khashman et al. [25] applied Otsu's threshold method, median filtering, Canny edge detection, and pattern averaging kernel to process 80 images from ALL-IDB2 dataset. They conducted experiments with different ratios of training and test sets and used a multi-layer perceptron for classification.

Putzu et al. [26] proposed an approach based on shape, color, and texture analysis for ALL identification in 245 sub-images obtained from ALL-IDB dataset. Bhattacharjee and Saini [27] used morphological operations, SVM, K-NN, and K-means clustering. Singhal and Singh [28] classified ALL using a SVM classifier with local binary patterns (LBP) and gray-level co-occurrence matrix (GLCM).

Rodrigues et al. [29] converted RGB images to HSV color space and extracted and converted V channel into binary. They analyzed morphological features and performed the classification with four different classifiers. Sus and Oliveira [30] extracted B and S channels, obtained from RGB and HSV color models, respectively. Segmentation methods, morphological analysis, and several classifiers were used to classify leukocytes in microscopy images. The approaches seen in [29] and [30] consider all 260 images from ALL-IDB2 dataset, and these authors reported difficulties in the segmentation stage due to problems intensity variations in the images.

Faria et al. [31] presented a simple and efficient combination of SIFT and SURF descriptors, stacking the descriptors of key points into a single matrix and evaluated two classifiers. Sahlol et al. [32] segmented each cell with Zack's threshold method and optimized a feed-forward neural network with elephant herd optimization (EHO) algorithm, which updates the weights and the biases of the network to classify ALL. In another study, Sahlol et al. [33] applied a bio-inspired meta-heuristic method called social spider optimization algorithm (SSOA) and tested several types of classifiers such as K-NN and SVM.

Although most of the previous works achieve an accuracy rate above 90%, they depend on the use of a proper segmentation process and handcrafted feature extraction. To overcome this limitation, strategies based on deep learning, specifically convolutional neural networks (CNNs), have been proposed to classify ALL in microscopy images and produce better results than all classical techniques [34–37].

Vogado et al. [38] proposed a method to identify leukemia using a hybrid-Leukocyte database and transfer learning strategy. Three CNNs architectures were used for feature extraction, and the gain ratio method was applied for feature selection. Further, the features selected were used as input to the SVM classifier.

Sipes and Li [39] developed a simple sequential CNN and compared it with conventional classifiers to classify the ALL in microscopy images. They evaluated the classifiers with training from scratch and tuned some hyperparameters empirically. However, to deal with small datasets, training based on transfer learning is most suitable [40].

Claro et al. [41] proposed a method for the automatic classification of two leukemia types and healthy cells in microscopy images using a CNN called AlertNet that shares the basic architecture of AlexNet [42] and uses a residual structure similar to ResNet [43]. In addition, they overcome overfitting using different data augmentation strategies and transfer learning.

Sahlol et al. [44] extracted features from lymphocytes images using VGG architecture and reducing the features extracted using a statistically enhanced salp swarm algorithm (SESSA). The training of VGG considered transfer learning, and six conventional classifiers received as input the features extracted by VGG architecture and reduced by SESSA algorithm.

Most recently, Das and Meher [45] proposed a hybrid transfer learning approach that ensembles MobilenetV2 and ResNet18 architectures. They considered the training based on transfer learning but did not exploit data augmentation techniques to deal with a small number of training images.

Automatic feature extraction is the main strength of previous studies based on deep CNNs. However, these studies do not investigate the appropriate choice of training-relevant hyperparameters, which can also significantly affect classification performance. Thus, to overcome this problem, we propose an approach based on GA to identify the optimal hyperparameters in a broad search space that considers a uniform distribution, data augmentation strategy, and training based on fine-tuning. Also, our method does not need the segmentation process, it is more robust to the intensity variations in the images, and it is also suitable to deal with the lack of training data for approaches based on deep learning.

Material and Methods

To fill the state-of-the-art gap concerning the best hyperparameters setup [46], we developed a novel approach based on GA. The main goal of this paper is to evaluate the performance of a deep residual CNN architecture to classify ALL in microscopy images. More precisely, we find the best hyperparameter combination and improve classification performance. Figure 1 illustrates the steps of the methodology adopted here.

Image dataset

The images used in this work were obtained from ALL-IDB2 dataset[1] provided by Department of Information Technology - Universitá degli Studi di Milano [47]. It contains 260 images, each with a single, centered cell in evidence, categorized into two classes: healthy (130 images) and immature (130 images). All images are in JPG format with 24 bit color depth and 2592×1944 pixels size. Figure 2 shows some images from the dataset for both classes, healthy and immature.

Deep Residual Network

Convolutional neural networks (CNNs) are the state-of-the-art in image classification tasks, designed to extract visual patterns from input images directly, without requiring handcrafted feature extraction [12]. In CNNs, the classification performance increases as the number of deep layers increases. However, as the number of layers increases, the accuracy tends to saturate and eventually degrade [48]. To overcome this problem, a residual network (ResNet) [43] was proposed. It uses residual blocks to address the gradient degradation in the training step and each residual block is composed of several stacked convolutional layers. Thus, the residual block adds a shortcut connection summing the input feature map (x) and the output of convolutional blocks ($f(x)$), as shown in Fig. 3(a).

In this study, we evaluated a deep residual network named ResNet-50 V2 [14]. ResNet-50 V2 is the enhanced version of ResNet-50 [43] that won the ILSVRC 2015 challenge [49]. The authors of [14] refined the residual block using a pre-activation variant of residual block, in which the gradients can flow through the shortcut connections to any other earlier layer unimpeded. The convolutional block is illustrated in Fig. 3(b), and it is composed of following layers: batch normalization, rectified linear unit (ReLU), and convolutional with a kernel of size k.

Hyperparameter Optimization

Hyperparameters define the optimization algorithms used, details of training, architecture, and topology of the CNN [20]. The choice of their values is essential to the performance of the CNN, although there is no optimum method for their selection. This choice may be performed empirically by evaluating different values until the algorithm provides satisfactory

[1] Available in: https://homes.di.unimi.it/scotti/all/

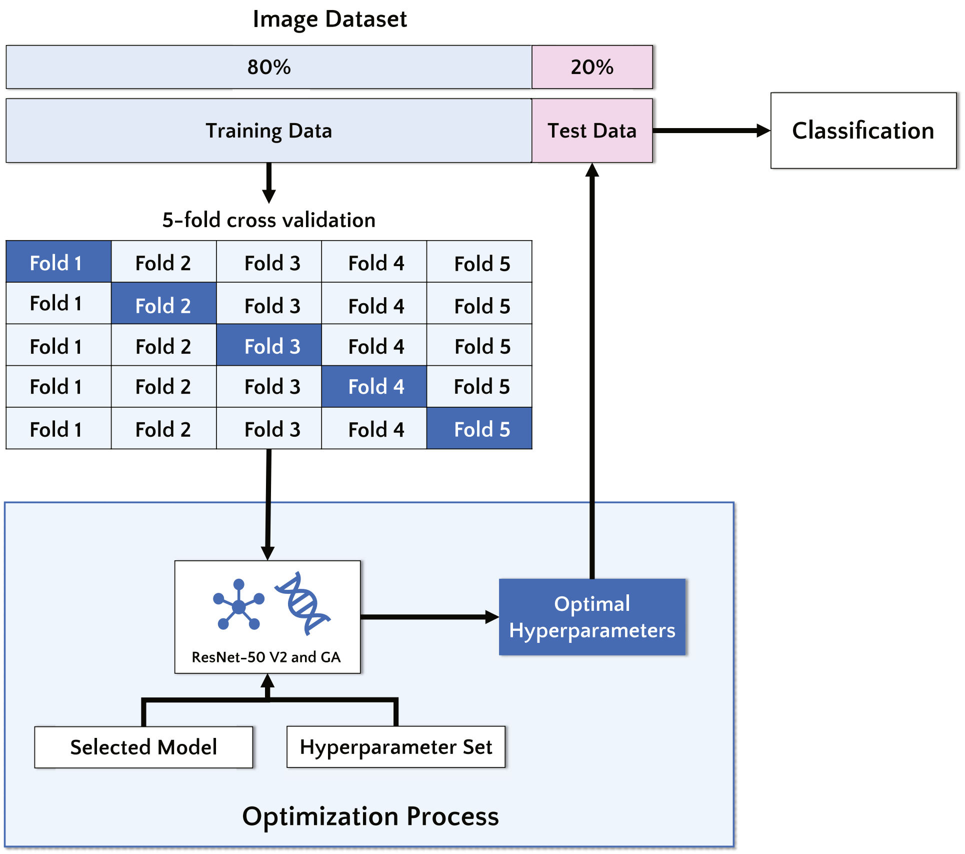


Fig. 1 Flowchart of the proposed approach

performance. However, the optimal values are unknown, and any definition of satisfaction degree is subjective.

Alternatively, the choice of hyperparameter values can be modeled as an optimization problem, where the hyperparameters are defined as decision variables, and the objective function minimizes the loss function. The fine-tuned hyperparameters in this study are as follows:

- **Dropout** (d)**:** dropout is a technique to deal with overfitting. It is based on the random dropping of neurons during the training process, i.e., a unit out is temporarily removed from the network, along with all its incoming and outgoing connections [50].
- **Learning rate** (l)**:** the learning rate is the main tuning parameter, being responsible for improving the stochas-

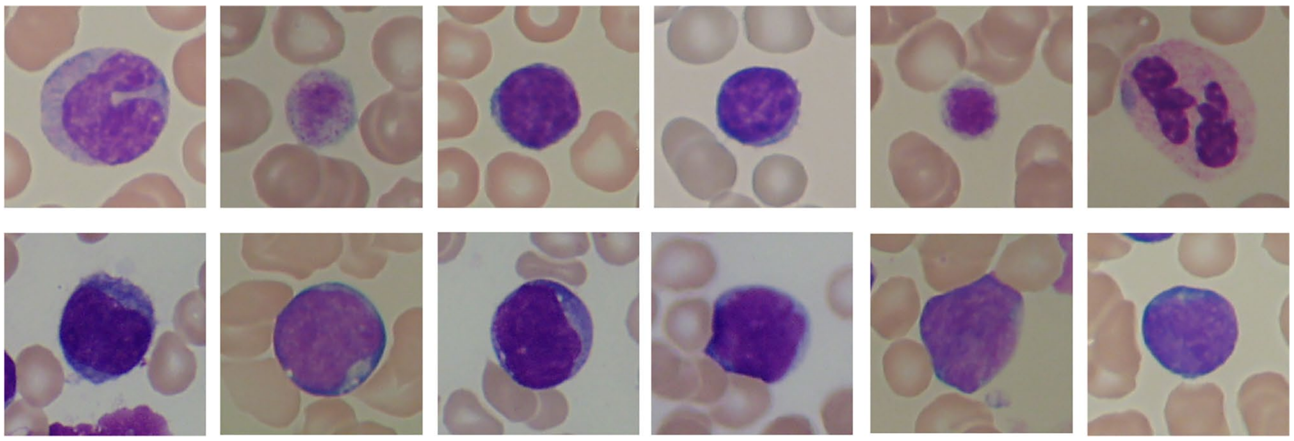


Fig. 2 Image instances from the ALL-IDB2 dataset showing healthy (top) and immature (bottom) lymphocytes

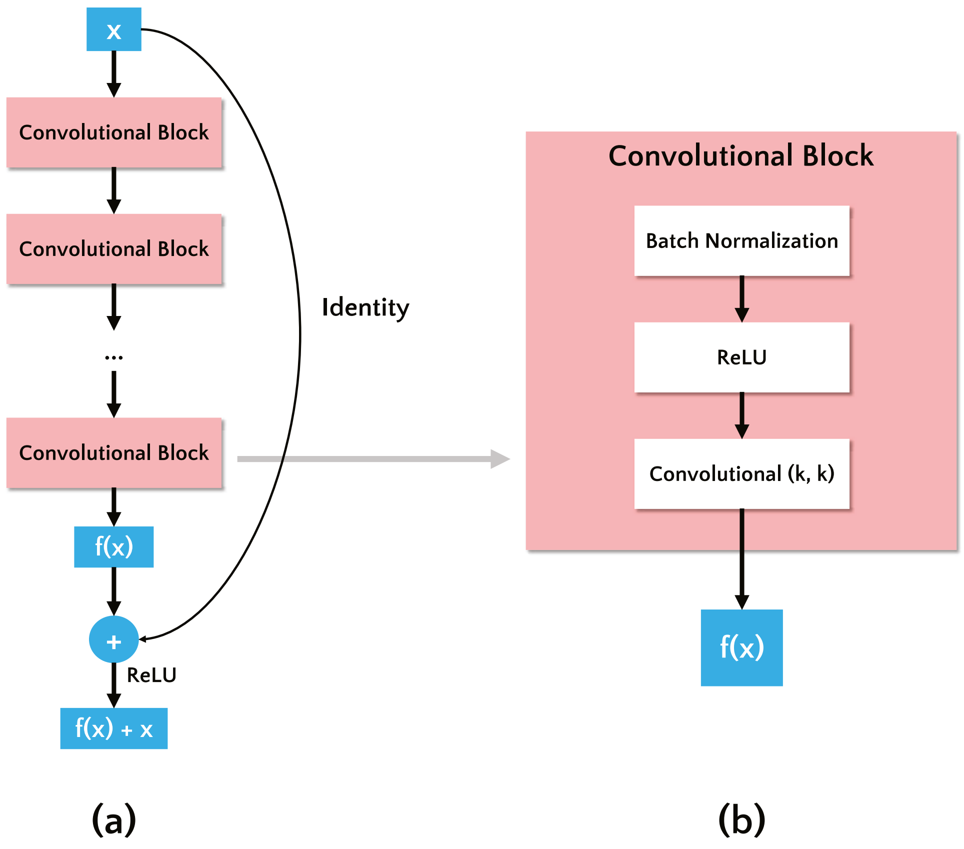


Fig. 3 (**a**) Residual block. (**b**) Detail of a convolutional block inside the Residual Block

tic gradient descent (SGD) [51] optimizer runtime. It defines the level of adjustment of weight connections and network topology applied at each training epoch. A high learning rate may sacrifice accuracy due to a lack of precision in the adjustments. On the other hand, a low learning rate requires more training epochs and longer processing time [20].

- **Momentum** (μ): the momentum coefficient [52] is responsible for reducing oscillations in the high-curvature regions of the loss function generated by the SGD. By default, its value is set to 1, but fine-tuning this hyperparameter may lead to improved results [13].

Proposed Genetic Algorithm

Genetic algorithm (GA) [53] is inspired by the natural biological evolution, and it is a type of evolutionary algorithm. Usually, a GA is composed of a population with n individuals and a series of bio-inspired operations, such as selection, crossover, and mutation.

GA is suitable for combinatorial optimization problems, and the quality of the solution obtained by GA suggests that it is better than simulated annealing or tabu search algorithms [54]. We choose GA over other heuristic algorithms for hyperparameter CNN evolution because the GA can incorporate domain-specific knowledge in all optimization phases [54], which is essential for hyperparameter combination in deep learning.

The conventional GA includes three main steps, as shown in Algorithm 1 and explained in the next subsections. Initially, n chromosomes are randomly generated according to the specific encoding method. Then, new offspring are continuously generated from the existing population and combined with the older generation to form a new generation. Finally, there is a population evolution process in which all individuals enter an iterative competition generating a new population composed of the survivors generated from the crossover of survivors and mutation operation.

Algorithm 1 GA Framework for Hyperparameter Optimization

```
1: Input: the image dataset $\mathcal{D}$, size of population $p$, the number of survival individuals per evolution generation $k$, the number of evolutionary generations $G$, and the feasible hyperparameter space $\mathcal{H}$.
2: Encoding: the gene includes three elements: dropout rate $(d)$, learning rate $(l)$, and momentum coefficient $(\mu)$.
3: Population initialization: $n$ chromosomes are randomly selected from feasible solution space and the fitness $f_i$ and evolutionary probability $p_i$ values $(i = 1, \cdots, n)$ of individuals are evaluated.
4: for $g = 1, \ldots, G$ do
5:     Selection: linear ranking and tournament $(t = 2)$.
6:     Crossover: two individuals are selected according to fitness value.
7:     Mutation: applying mutation operator.
8: end for
9: Output: In $G^{th}$ generation, the individual with the largest fitness is the optimal solution in the hyperparameter space $\mathcal{H}$.
```

Population Initialization

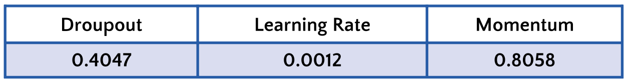
Initialization is the process of randomly selecting candidate solutions in the search space. In this study, we defined the search space size as an input from the hyperparameter set. Also, we use a uniform distribution to ensure the random distribution of candidates in the search space. There is a particular interval for all hyperparameters in which the gene can assume values within the defined range, limiting the search space of the GA.

During population initialization, hyperparameters are defined, and a random value is chosen within the predefined interval to encode the chromosome. We must include the information of dropout, learning rate, and momentum coefficient for the chromosome encoding in an array. Each chromosome is composed of three genes, and each gene encodes a real value for its respective hyperparameter. The structure of the chromosome is shown in Fig. 4.

Selection

The selection is based on fitness, i.e., the fittest individual is selected to participate in the reproduction process. In this operator, the current generation members with the highest fitness values are the most likely to generate the next population. In this study, we applied linear ranking with tournament selection.

- Linear Ranking: It is based on the classification of individuals according to fitness, being the probability of selecting an individual depends solely on fitness. For this, the worst individual gets a rating of 1; the second-



Droupout	Learning Rate	Momentum
0.4047	0.0012	0.8058

Fig. 4 Chromosome representation for hyperparameter optimization

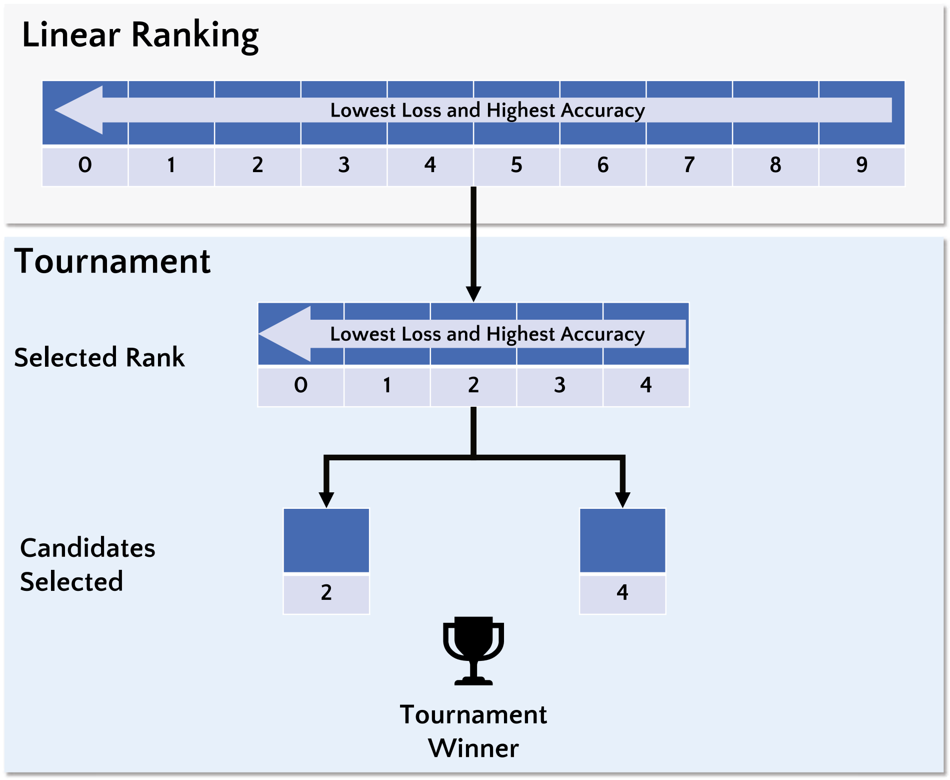


Fig. 5 Selection based on linear ranking and tournament. The linear ranking is applied to select half of the best chromosomes, sorting the highest accuracy and lowest loss at the top of the population. Then, the selection with a tournament ($t = 2$) is performed considering the best-ranked chromosomes

worst individual gets a rating of 2. This idea repeats successively until the best individual receives a rating of n (corresponding to the number of chromosomes in the population). Thus, based on their rating ($rank_i$), each individual i has the probability P_i to be selected from a population of n individuals, as defined in Eq. 1 [55].

$$P_i = \frac{rank_i}{n \times (n-1)} \tag{1}$$

- Tournament: The tournament selection [56] aims to select a set of k individuals randomly, which will be sorted according to their relative fitness. Afterward, the fittest individual is chosen for reproduction. This process is repeated several times for the entire population, and the probability P_i of each individual being selected is expressed in Eq. 2.

$$P_i = \begin{cases} C_{n-1}^{k-1} & , if\ i \in [1, n-k-1] \\ 0 & , if\ i \in [n-k, 1] \end{cases} \tag{2}$$

Figure 5 shows the selection method based on linear ranking and tournament applied in this study. All individuals are ranked, and half of the best chromosomes will be selected for the tournament. After, a tournament ($t = 2$) is performed considering the best-ranked chromosomes. Afterward, the winner of the tournament will be selected for the reproduction process.

Crossover and Mutation

The crossover operator is used to generate new individuals by recombining the genes of parent chromosomes. In this study, we used a single-point crossover. For each iteration in the GA process, one point was randomly selected. For example, as illustrated in Fig. 6, to generate a “child 1”, the random crossover point is index 1. Thus, index 0 and 1 from “parent 1” will be selected as head, and index 2 from “parent 2” will be chosen as tail. After, a similar process occurs to generate a “child 2”.

Also, we applied the mutation operator, which is initiated after the crossover process by randomly modifying one bit of an individual’s chromosome to generate a child.

Fitness Evaluation

In this paper, we considered the objective function as the loss-function defined by $\mathcal{L}(W)$. Eq. 3 show that $\mathcal{L}(W)$ is computed over a set of training samples X_j considering the tuned weights W, parameters $f(x_j)$, and the known classes y_j, where j represents the classes lymphocytes immature and healthy.

$$\mathcal{L}(W) = \frac{1}{n} \sum_{j=1}^{N} \ell(y_j, f(x_j; W)) \tag{3}$$

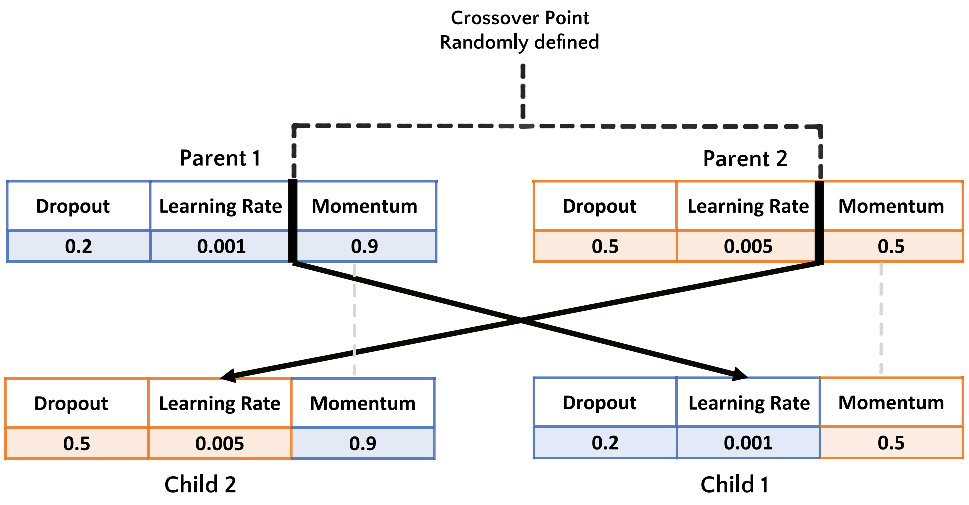


Fig. 6 Example of single-point crossover

In this way, to minimize $\mathcal{L}(W)$, we applied the stochastic gradient descent (SGD) [51] optimization algorithm with hyperparameters (dropout, learning rate, and momentum) optimized through GA. Consequently, when we minimize the loss-function, the accuracy (Eq. 4) is maximized.

Experimental Protocol

Firstly, we have to adapt the image size in the dataset for the input of the ResNet-50 V2. All images were resized to 224 × 224 pixels size using bilinear interpolation.

We arranged the dataset into training and test sets: 80% for training (further split into 80% for training and 20% for internal validation using stratified fivefold cross-validation method [57]) and 20% for testing (external validation). We applied fivefold internal cross-validation to all experiments, reporting the results on the external test set.

Neural networks like ResNet generally require a large amount of data during the training in order to avoid overfitting. Given the reduced number of images in the dataset, we used data augmentation and transfer learning techniques to improve the training of the networks [40, 41]. Data augmentation enables to increase the training set artificially by generating new samples of a given image under different variations, without introducing labeling costs [42]. In our experiments, we performed the data augmentation using only vertical and horizontal flips.

Additionally, we applied transfer learning, i.e., we used the pre-trained 2012 ImageNet weights to initialize the network [49]. This strategy enables to use low-level features learned in larger datasets, which are better in comparison to the ones learned using a network trained from scratch in a smaller dataset. For comparison, ImageNet data set contains approximately 1.2 million images divided into 1000 classes. In the sequence, we used our small dataset to fine-tune the network weights to our problem, i.e., we initialized all convolutional layers with weights from the pre-trained model, and the fine-tuning was performed only in the deeper layers [13].

We report our results in terms of average accuracy, precision, recall, and F1-score [58], averaged over the fivefold:

- Accuracy: is the ratio between the correct classifications and total samples (Eq. 4).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{4}$$

- Precision: is the ratio between TP and the total of positives classification (Eq. 5).

$$Precision = \frac{TP}{TP + FP} \tag{5}$$

- Recall: is the proportion of TP correctly classified (Eq. 6).

$$Recall = \frac{TP}{TP + FN} \tag{6}$$

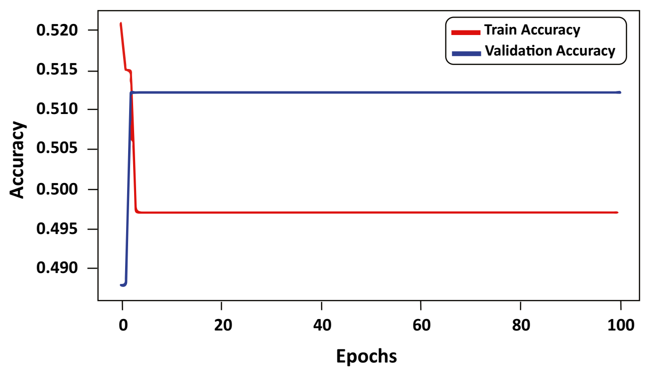
- F1-Score: is the harmonic average of the precision and recall (Eq. 7).

$$F1{-}Score = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{7}$$

where *TP* is true positive, *TN* is true negative, *FP* is false positive and *FN* is false negative. For a given fold, we obtain these

Table 1 Classification results considering training without GA using the test set

Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
50.00	25.00	50.00	33.00



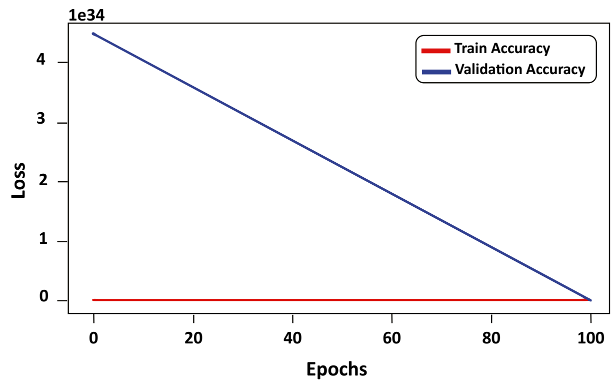


Fig. 7 Evolution of accuracy and loss values considering the training and validation set

metrics and then we compute the average metrics among the folders.

All experiments were conducted using the Google Colaboratory cloud service with a machine Intel(R) Xeon(R) 2.20GHz processor, 12 GB RAM, and a GPU NVIDIA Tesla T4. The experiments were programmed using Python (version 3.6) and Keras 2.0 [59] deep learning framework. The hyperparameter optimization algorithms random search and Bayesian optimization were drawn from the KerasTuner library [60], version 1.1.0.

Results and Discussion

We propose experiments aiming to answer the following questions:

- What are the best values of hyperparameters (dropout, learning rate, and momentum coefficient), which bring the highest classification performance?
- How much does the hyperparameter optimization increase the performance of ResNet-50 V2, considering fine-tuning approach?
- Considering metrics derived from confusion matrix, what is the most suitable approach for leukocytes image classification: training without GA or with GA?
- In terms of consumed time for training and the classification performance, considering GA, random search, and Bayesian optimal parameter finding methods, which one is the best approach?

Table 2 Hyperparameter search space used for optimization

Hyperparameter	Value
Dropout	x ∈ [0.0, 0.5]
Learning Rate	x ∈ [0.005, 0.01]
Momentum	x ∈ [0.0, 1.0]

Classification Without Hyperparameter Optimization

Aiming to assess the impact of the training without hyperparameter optimization, we analyze the classification performance according to metrics of accuracy, precision, recall and F1-score. We trained the CNN SGD optimizer with a learning rate of 0.01, momentum of 1.0, batch size of eight, and 100 epochs. The values of learning rate and momentum were defined according to the literature [13].

Regarding the classification performance, Table 1 present the result obtained for the test set. Results show that using the default values generates poor results, indicating that these hyperparameters need to be well adjusted in order to the CNN to achieve a good performance.

In order to assess the values of accuracy and loss during the training phase, the evolution of these values is presented in Fig. 7, considering the fifth iteration of the *k*-fold. Throughout the training, the behavior of the accuracy and loss function generated noise values resulting in underfitting.

Evaluating the Impact of GA Optimization

This experiment aimed to demonstrate that proper hyperparameter tuning improves the performance of the ResNet-50

Table 3 Hyperparameter optimized with GA

		Hyperparameter	
Fold	Dropout	Learning Rate	Momentum
1	0.01326	0.00179	0.15547
2	0.01489	0.00187	0.15547
3	0.01326	0.00187	0.15547
4	0.01489	0.00187	0.09274
5	0.01326	0.00179	0.15547

Table 4 Classification results considering training with five different hyperparameters setting from Table 3 applied on the test set

HS	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)
1	100.00	100.00	100.00	100.00
2	98.08	98.15	98.08	98.08
3	96.15	96.43	96.15	96.15
4	100.00	100.00	100.00	100.00
5	98.08	98.15	98.08	98.08
Average	**98.46 ± 0.01**	**98.55 ± 0.01**	**98.46 ± 0.01**	**98.46 ± 0.01**

V2 deep CNN. The selection of hyperparameter values was carried out as an optimization problem, as described in "Hyperparameter Optimization".

We applied the GA to optimize the fine-tune values, namely the dropout, learning rate, and momentum coefficient in pre-trained ResNet-50 V2. The hyperparameter optimization process for each fold took about 78 minutes to complete. The GA setting considered a mutation rate of 30%, population size of ten, and five generations. For each generation, the CNN was trained for 20 epochs. Usually, a larger population size and larger maximal generation number results in better performance, but this is time-consuming and requires more computational resources. To overcome this limitation, we applied the same setting adopted by [61] to optimize a deep CNN with GA.

Table 2 presents the hyperparameter space search evaluated in this study. All hyperparameters are searched considering a uniform distribution. Besides, Table 3 presents the best values for each hyperparameter returned by the GA.

For all analyses, we consider "HS" as the abbreviation of the hyperparameter setting, which is obtained from five-fold internal cross-validation. Given the results obtained by GA, we trained the ResNet-50 V2 using the optimized hyperparameters with 100 epochs.

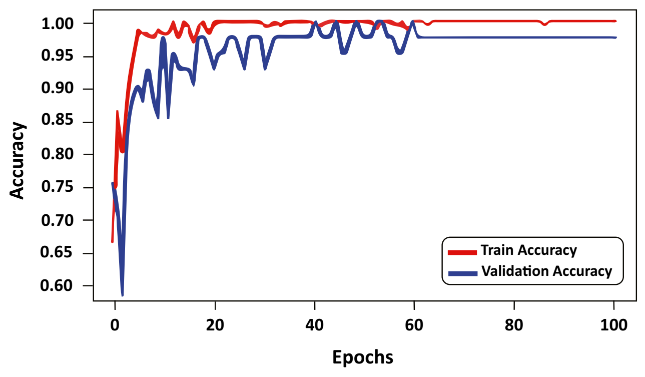
As shown in Table 4, the GA hyperparameter optimization improved the classification performance significantly. When we compare the results obtained with and without GA, it is possible to observe that optimization generated an accuracy enhancement of 48.46 percentage points (accuracy increased from 50.00% to 98.46%).

The results suggest that our hyperparameter optimization based on GA is superior in all evaluated metrics over the strategy without automatic optimization. This implies that defining optimal hyperparameters requires lots of domain expertise, and manually pre-configurations limit the feasible solution space to miss out on the better set of hyperparameters. It is a relief that GA can automatically search for the optimal solution in a large space without a manual setting. Thus, the automatic selection by GA reduces the domain expertise requirements in deep learning for researchers and helps some non-expert researchers to define the best hyperparameters obtaining high-performance classifiers.

As a check on the learning behavior of the training step, Fig. 8 shows the loss and accuracy values considering the second iteration of the *k*-fold. It is important to note the low values of the loss function. This behavior suggests that the training did not overfit the data, thus retaining the generalization property of the deep residual CNN.

The proposed model optimized by GA has further exploited the potential of ResNet-50 V2 and improved the acute lymphoblastic leukemia detection performance. In summary, the proposed model shows better and more balanced performance compared with conventional ResNet-50 V2, without GA optimization.

Figure 9 shows the feature map acquired in selected convolutional layers of the ResNet-50 V2 optimized by GA. The filters embedded in the convolutional layers shows that most of the features behaved as texture extractors and edge detectors, preserving the leukocyte cell spatial information. The results demonstrate that our approach does not need the segmentation process and it is more robust to deal with the different leukocyte features.



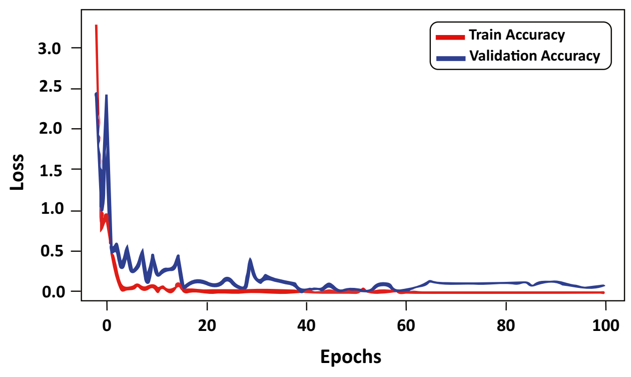


Fig. 8 Evolution of accuracy and loss values considering the training and validation set and GA hyperparameter optimization

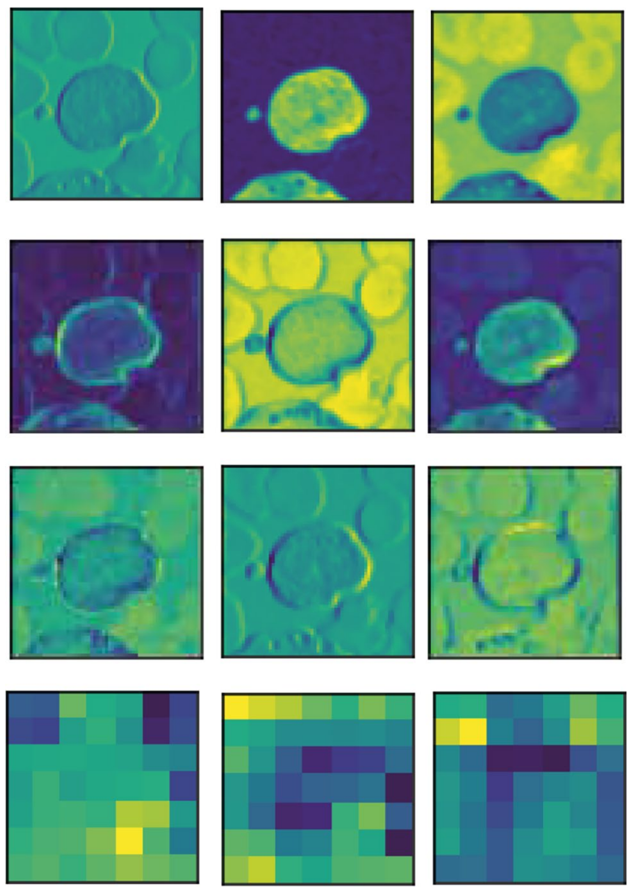


Fig. 9 Feature maps obtained for some of the convolutional layers of the ResNet-50 V2 optimized by GA

Table 5 presents the confusion matrix for ResNet-50 V2 optimized by GA, and it shows the aspects of the classification problem investigated in this study. It can be seen that our approach classifies 97.69% of leukemia images correctly and does not require preprocessing or segmentation process (commonly used in state-of-the-art techniques).

Comparison of Different Hyperparameter Optimization Approaches

In addition to optimizing with GA, we also evaluated the impact of optimization using random search and Bayesian optimization approaches to compare their performance. We choose these approaches because they are widely used in CNN hyperparameter optimization [17, 62–67].

Random search [68] is a method that selects a suitable set of optimal hyperparameter configurations by testing random combinations of hyperparameters. However, the method is entirely random and uses no intelligence in selecting the

Table 5 Confusion matrix for ResNet-50 V2 optimized with GA averaged over the five hyperparameters

	Healthy	**Leukemia**
Healthy	99.23%	0.77%
Leukemia	2.31%	97.69%

Table 6 Hyperparameter optimized with random search

	Hyperparameter		
Fold	**Dropout**	**Learning Rate**	**Momentum**
1	0.31971	0.00519	0.40853
2	0.31971	0.00519	0.40853
3	0.31971	0.00519	0.40853
4	0.27407	0.00533	0.49753
5	0.31971	0.00519	0.40853

Table 7 Hyperparameter optimized with Bayesian optimization

	Hyperparameter		
Fold	**Dropout**	**Learning Rate**	**Momentum**
1	0.0	0.005	0.0
2	0.00624	0.005	0.0
3	0.00610	0.005	0.0
4	0.86178	0.00944	0.35184
5	0.5	0.005	0.0

Table 8 Classification results on the test set considering training with five different hyperparameters setting obtained from random search

HS	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
1	71.15	71.44	71.15	71.06
2	51.92	75.49	51.92	37.47
3	50.00	25.00	50.00	33.00
4	71.15	71.97	71.15	70.88
5	67.31	80.23	67.31	63.40
Average	**62.31 ± 0.10**	**64.83 ± 0.22**	**62.31 ± 0.10**	**55.16 ± 0.18**

Table 9 Classification results on the test set considering training with five different hyperparameters setting obtained from Bayesian optimization

HS	**Accuracy (%)**	**Precision (%)**	**Recall (%)**	**F1-Score (%)**
1	92.31	92.56	92.31	92.30
2	98.08	98.15	98.08	98.08
3	98.08	98.15	98.08	98.08
4	57.69	68.84	57.69	50.35
5	55.77	76.53	55.77	45.01
Average	**80.39 ± 0.21**	**86.85 ± 0.13**	**80.39 ± 0.21**	**76.76 ± 0.26**

Table 10 Average classification and optimization time for each optimization method

Method	Accuracy (%)	Precision (%)	Recall (%)	F1-Score (%)	Optimization Time
Random Search	62.31	64.83	62.31	55.16	22min 10s
Bayesian Optimization	80.39	86.85	80.39	76.76	22min 36s
Genetic Algorithm	**98.46**	**98.55**	**98.46**	**98.46**	78min 45s

trial points resulting in a non-adaptive approach. On the other hand, Bayesian optimization [69] finds the value that minimizes the objective function by building a probability model (using the Gaussian process) based on the previous evaluation results of the target.

For a fair comparison, we considered the same search space evaluated in GA hyperparameter optimization (see Table 2). The relevant hyperparameters obtained for each fold with random search and Bayesian optimization are presented in Tables 6 and 7, respectively.

By comparing the values returned by the two optimizers, we observed that the Bayesian optimization better explored the search space by finding different possible optimal solutions.

Given the optimized hyperparameters obtained by random search and Bayesian optimization, we trained the ResNet-50 V2 with 100 epochs. As shown in Table 8, the random search optimization allowed to achieve an accuracy of 62.31%. However, this result is only better than classification without hyperparameter optimization (as presented in "Classification Without Hyperparameter Optimization"). In contrast, Bayesian optimization performed better than random search obtained an accuracy of 80.39% as shown in Table 9. Furthermore, when considering the results obtained in hyperparameter setting 2 and 3, we observed that the accuracy is very close to the results obtained with the GA.

In order to compare the hyperparameter optimization methods, Table 10 shows the classification performance metrics and optimization time. It is important to note that the GA improved leukemia image classification significantly, although it requires more time to perform the optimization. This result demonstrates that random search and Bayesian optimization approaches don't explore the search space distribution as well as the GA. At the same time, GA can incorporate domain-specific knowledge in all optimization phases, guaranteeing gains in all classification performance metrics. Also, once GA obtains the best hyperparameters, the training time becomes similar to training with manual configuration, taking around 10 minutes.

Table 11 Highest accuracy of other classification methods using the ALL-IDB2 dataset

Method	Accuracy (%)
Rodrigues et al. [29]	85.00
Singhal and Singh [28]	93.84
Sahlol et al. [32]	91.80
Sus and Oliveira [30]	94.60
Faria et al. [31]	97.22
Sahlol et al. [33]	95.23
Sahlol et al. [44]	96.11
Das and Meher [45]	97.18
Our proposal (2022)	**98.46**

Comparison With Literature

We also compared the best result achieved in this study in terms of accuracy with other state-of-art work in the literature. The best result in our work was obtained with GA hyperparameter optimization, which scored 98.46% of accuracy (as shown in Table 4). The best results reported in the literature are presented in Table 11 for the same ALL-IDB2 dataset. It can be seen that our best score is upper to the best state-of-the-art technique reported in the literature.

Conclusion

The results presented in this paper point to a promising use of deep CNN with hyperparameter optimization to classify ALL cases based on microscopy images. We compared the performance of ResNet-50 V2 optimized with GA, trained with transfer learning and data augmentation approaches. Our best result of 98.46% was upper to the highest accuracy score presented in the literature.

Additionally, we demonstrated that hyperparameter tuning with GA leads CNN to achieve higher performance rates against approaches without optimization and optimization using random search and Bayesian algorithms, answering the raised questions regarding classification performance. Besides, we found, quantified, and presented the hyperparameters and measured the improvement they promoted in the performance of ResNet-50 V2 over the dataset evaluated.

Moreover, the transfer learning strategy helped to reduce the training time, enabling us to evaluate the hyperparameter optimization and the use of k-fold cross-validation. Our dataset partitioning complies with the state of the art once our experimental protocol avoids biased model and abnormal CNN behaviors.

The presented results open new opportunities towards better machine learning based on deep CNN for automated detection of leukemia and the development of new computer-aided diagnosis applications. We believe that hyperparameter optimization using GA and other evolutionary approaches is an alternative to overcome challenges in deep CNNs.

As future work, we intend to evaluate other CNN architectures, different data augmentation strategies, and different settings for GA with a more extensive set of hyperparameters. Also, we plan to test the proposed method with other datasets of images containing lymphocytes and other types of cells to verify that our findings hold for similar datasets.

Acknowledgements André R. Backes and Gina M. B. de Oliveira gratefully acknowledge the financial support of CNPq (Grant #307100/2021-9 and #311022/2018-9). Also, this study was financed in part by the Coordenação de Aperfeiçoamento de Pessoal de Nível Superior – Brasil (CAPES) – Finance Code 001.

References

1. M. Onciu. Acute lymphoblastic leukemia. Hematology/Oncology Clinics of North America, 23(4):655 – 674, 2009. Neoplastic Hematopathology. https://doi.org/0.1016/j.hoc.2009.04.009
2. L. M. Force and et al. The global burden of childhood and adolescent cancer in 2017: an analysis of the global burden of disease study 2017. The Lancet Oncology, 20(9):1211–1225, 2019. https://doi.org/10.1016/S1470-2045(19)30339-0
3. R. L. Siegel, K. D. Miller, and A. Jemal. Cancer statistics, 2016. CA: A Cancer Journal for Clinicians, 66(1):7–30, 2016. https://doi.org/10.3322/caac.21332
4. E. Ward, C. DeSantis, A. Robbins, B. Kohler, and A. Jemal. Childhood and adolescent cancer statistics, 2014. CA: A Cancer Journal for Clinicians, 64(2):83–103, 2014.
5. M. Hallek, B. D. Cheson, D. Catovsky, F. Caligaris-Cappio, G. Dighiero, H. Dhner, P. Hillmen, M. J. Keating, E. Montserrat, K. R. Rai, and T. J. Kipps. Guidelines for the diagnosis and treatment of chronic lymphocytic leukemia: a report from the International Workshop on Chronic Lymphocytic Leukemia updating the National Cancer Institute-Working Group 1996 guidelines. Blood, 111(12):5446–5456, 06 2008. https://doi.org/10.1182/blood-2007-06-093906
6. A. R. Backes and J. J. de Mesquita S Junior. Virus classification by using a fusion of texture analysis methods. In 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), pages 290–295, 2020. https://doi.org/10.1109/IWSSIP48289.2020.9145325
7. D. F. dos Santos, P. R. de Faria, B. A. N. Travenolo, and M. Z. do Nascimento. Automated detection of tumor regions from oral histological whole slide images using fully convolutional neural networks. Biomedical Signal Processing and Control, 69:102921, 2021. https://doi.org/10.1016/j.bspc.2021.102921
8. D. S. Kermany and et al. Identifying medical diagnoses and treatable diseases by image-based deep learning. Cell, 172(5):1122–1131.e9, 2018. https://doi.org/10.1016/j.cell.2018.02.010
9. L. Nanni, S. Ghidoni, and S. Brahnam. Ensemble of convolutional neural networks for bioimage classification. Applied Computing and Informatics, 2018. https://doi.org/10.1016/j.aci.2018.06.002
10. L. F. Rodrigues, M. C. Naldi, and J. F. Mari. Comparing convolutional neural networks and preprocessing techniques for HEp-2 cell classification in immunofluorescence images. Computers in Biology and Medicine, 116:103542, 2020. https://doi.org/10.1016/j.compbiomed.2019.103542
11. N. Schwalbe and B. Wahl. Artificial intelligence and the future of global health. The Lancet, 395(10236):1579–1586, 2020. https://doi.org/10.1016/S0140-6736(20)30226-9
12. I. Goodfellow, Y. Bengio, and A. Courville. Deep Learning. MIT Press, 2016. http://www.deeplearningbook.org
13. M. A. Ponti, L. S. F. Ribeiro, T. S. Nazare, T. Bui, and J. Collomosse. Everything you wanted to know about deep learning for computer vision but were afraid to ask. In 2017 30th SIBGRAPI Conference on Graphics, Patterns and Images Tutorials (SIBGRAPI-T), pages 17–41, Oct 2017. https://doi.org/10.1109/SIBGRAPI-T.2017.12
14. K. He, X. Zhang, S. Ren, and J. Sun. Identity mappings in deep residual networks, 2016. arXiv:1603.05027
15. J. S. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl. Algorithms for hyper-parameter optimization. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, Advances in Neural Information Processing Systems 24, pages 2546–2554. Curran Associates, Inc., 2011. http://papers.nips.cc/paper/4443-algorithms-for-hyper-parameter-optimization.pdf
16. A. P. Marcos, N. L. Silva Rodovalho, and A. R. Backes. Coffee leaf rust detection using genetic algorithm. In 2019 XV Workshop de Viso Computacional (WVC), pages 16–20, 2019. https://doi.org/10.1109/WVC.2019.8876934
17. R. Moreira, L. F. Rodrigues, P. F. Rosa, R. L. Aguiar, and F. d. O. Silva. Packet vision: a convolutional neural network approach for network traffic classification. In 2020 33rd SIBGRAPI Conference on Graphics, Patterns and Images (SIBGRAPI), pages 256–263, 2020. https://doi.org/10.1109/SIBGRAPI51738.2020.00042
18. R. R. Silva, M. C. Escarpinati, and A. R. Backes. Sugarcane crop line detection from UAV images using genetic algorithm and Radon transform. Signal, Image and Video Processing, 15(8):1723–1730, 2021. https://doi.org/10.1007/s11760-021-01908-3
19. P. Liashchynskyi and P. Liashchynskyi. Grid search, random search, genetic algorithm: A big comparison for NAS. CoRR, abs/1912.06059, 2019.
20. Y. Bengio. Practical recommendations for gradient-based training of deep architectures. In Neural Networks: Tricks of the Trade: Second Edition, pages 437–478, Berlin, Heidelberg, 2012. Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-35289-8_26
21. V. Piuri and F. Scotti. Morphological classification of blood leucocytes by microscope images. In 2004 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications, 2004. CIMSA., pages 103–108, 2004. https://doi.org/10.1109/CIMSA.2004.1397242
22. F. Scotti. Automatic morphological analysis for acute leukemia identification in peripheral blood microscope images. In CIMSA. 2005 IEEE International Conference on Computational Intelligence for Measurement Systems and Applications, 2005., pages 96–101, 2005. https://doi.org/10.1109/CIMSA.2005.1522835
23. S. Mohapatra, S. S. Samanta, D. Patra, and S. Satpathi. Fuzzy based blood image segmentation for automated leukemia detection. In 2011 International Conference on Devices and Communications (ICDeCom), pages 1–5, 2011. https://doi.org/10.1109/ICDECOM.2011.5738491
24. S. Mohapatra, D. Patra, and S. Satpathy. An ensemble classifier system for early diagnosis of acute lymphoblastic leukemia in blood microscopic images. Neural Computing and Applications, 24(7):1887–1904, 2014. https://doi.org/10.1007/s00521-013-1438-3
25. A. Khashman and H. H. Abbas. Acute lymphoblastic leukemia identification using blood smear images and a neural classifier. In I. Rojas, G. Joya, and J. Cabestany, editors, Advances in Computational Intelligence, pages 80–87, Berlin, Heidelberg, 2013. Springer Berlin Heidelberg.
26. L. Putzu, G. Caocci, and C. Di Ruberto. Leucocyte classification for leukaemia detection using image processing techniques.

Artificial Intelligence in Medicine, 62(3):179 – 191, 2014. https://doi.org/10.1016/j.artmed.2014.09.002
27. R. Bhattacharjee and L. M. Saini. Robust technique for the detection of acute lymphoblastic leukemia. In 2015 IEEE Power, Communication and Information Technology Conference (PCITC), pages 657–662, 2015. https://doi.org/10.1109/PCITC.2015.7438079
28. V. Singhal and P. Singh. Texture features for the detection of acute lymphoblastic leukemia. In S. C. Satapathy, A. Joshi, N. Modi, and N. Pathak, editors, *Proceedings of International Conference on ICT for Sustainable Development*, pages 535–543, Singapore, 2016. Springer Singapore.
29. L. F. Rodrigues, J. H. Silva, P. H. C. C. Gondim, and J. F. Mari. Leukocytes classification in microscopy images for acute lymphoblastic leukemia identification. In XII Workshop de Visão Computacional, pages 68–73, Campo Grande, MS, Brazil, 2016. WVC.
30. J. F. L. Sus and L. F. Oliveira. Leukocyte segmentation and classification using computational vision. In XIII Workshop de Visão Computacional, Natal, RN, Brazil. WVC., pages 153–157, 2017.
31. L. C. de Faria, L. F. Rodrigues, and J. F. Mari. Cell classification using handcrafted features and bag of visual words. In 2018 Workshop de Visão Computacional (WVC), pages 68–73, Nov 2018.
32. A. T. Sahlol, F. H. Ismail, A. Abdeldaim, and A. E. Hassanien. Elephant herd optimization with neural networks: A case study on acute lymphoblastic leukemia diagnosis. In 2017 12th International Conference on Computer Engineering and Systems (ICCES), pages 657–662, 2017. https://doi.org/10.1109/ICCES.2017.8275387
33. A. T. Sahlol, A. M. Abdeldaim, and A. E. Hassanien. Automatic acute lymphoblastic leukemia classification model using social spider optimization algorithm. Soft Computing, 23(15):6345–6360, 2019. https://doi.org/10.1007/s00500-018-3288-5
34. S. Anwar and A. Alam. A convolutional neural network–based learning approach to acute lymphoblastic leukaemia detection with automated feature extraction. Medical & Biological Engineering & Computing, 58(12):3113–3121, 2020. https://doi.org/10.1007/s11517-020-02282-x
35. R. B. Hegde, K. Prasad, H. Hebbar, and B. M. K. Singh. Comparison of traditional image processing and deep learning approaches for classification of white blood cells in peripheral blood smear images. Biocybernetics and Biomedical Engineering, 39(2):382 – 392, 2019. https://doi.org/10.1016/j.bbe.2019.01.005
36. T. Pansombut, S. Wikaisuksakul, K. Khongkraphan, and A. Phon-on. Convolutional neural networks for recognition of lymphoblast cell images. Computational Intelligence and Neuroscience, 2019:7519603, 2019. https://doi.org/10.1155/2019/7519603
37. S. Shafique and S. Tehsin. Acute lymphoblastic leukemia detection and classification of its subtypes using pretrained deep convolutional neural networks. Technology in Cancer Research & Treatment, 17:1533033818802789, 2018. PMID: 30261827. https://doi.org/10.1177/1533033818802789
38. L. H. Vogado, R. M. Veras, F. H. Araujo, R. R. Silva, and K. R. Aires. Leukemia diagnosis in blood slides using transfer learning in cnns and svm for classification. Engineering Applications of Artificial Intelligence, 72:415 – 422, 2018. https://doi.org/10.1016/j.engappai.2018.04.024
39. R. Sipes and D. Li. Using convolutional neural networks for automated fine grained image classification of acute lymphoblastic leukemia. In 2018 3rd International Conference on Computational Intelligence and Applications (ICCIA), pages 157–161, 2018. https://doi.org/10.1109/ICCIA.2018.00036
40. N. Tajbakhsh, J. Y. Shin, S. R. Gurudu, R. T. Hurst, C. B. Kendall, M. B. Gotway, and J. Liang. Convolutional neural networks for medical image analysis: Full training or fine tuning? IEEE Transactions on Medical Imaging, 35(5):1299–1312, 2016.
41. M. Claro, L. Vogado, R. Veras, A. Santana, J. Tavares, J. Santos, and V. Machado. Convolution neural network models for acute leukemia diagnosis. In 2020 International Conference on Systems, Signals and Image Processing (IWSSIP), pages 63–68, 2020. https://doi.org/10.1109/IWSSIP48289.2020.9145406
42. A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In Advances in neural information processing systems, pages 1097–1105, 2012.
43. K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pages 770–778, June 2016. https://doi.org/10.1109/CVPR.2016.90
44. A. T. Sahlol, P. Kollmannsberger, and A. A. Ewees. Efficient classification of white blood cell leukemia with improved swarm optimization of deep features. Scientific Reports, 10(1):2536, 2020. https://doi.org/10.1038/s41598-020-59215-9
45. P. K. Das and S. Meher. An efficient deep convolutional neural network based detection and classification of acute lymphoblastic leukemia. Expert Systems with Applications, 183:115311, 2021. https://doi.org/10.1016/j.eswa.2021.115311
46. S. K. Zhou, H. Greenspan, C. Davatzikos, J. S. Duncan, B. Van Ginneken, A. Madabhushi, J. L. Prince, D. Rueckert, and R. M. Summers. A review of deep learning in medical imaging: Imaging traits, technology trends, case studies with progress highlights, and future promises. Proceedings of the IEEE, 109(5):820–838, 2021. https://doi.org/10.1109/JPROC.2021.3054390
47. R. D. Labati, V. Piuri, and F. Scotti. All-IDB: The acute lymphoblastic leukemia image database for image processing. In 2011 18th IEEE International Conference on Image Processing, pages 2045–2048, 2011. https://doi.org/10.1109/ICIP.2011.6115881
48. R. K. Srivastava, K. Greff, and J. Schmidhuber. Training very deep networks. CoRR, abs/1507.06228, 2015.
49. J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In 2009 IEEE Conference on Computer Vision and Pattern Recognition, pages 248–255, June 2009. https://doi.org/10.1109/CVPR.2009.5206848
50. N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. J. Mach. Learn. Res., 15(1):1929-1958, Jan. 2014.
51. Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. Proceedings of the IEEE, 86(11):2278–2324, 1998. https://doi.org/10.1109/5.726791
52. B. Polyak. Some methods of speeding up the convergence of iteration methods. USSR Computational Mathematics and Mathematical Physics, 4(5):1 – 17, 1964. https://doi.org/10.1016/0041-5553(64)90137-5
53. J. H. Holland. Adaptation in Natural and Artificial Systems: An Introductory Analysis with Applications to Biology, Control and Artificial Intelligence. MIT Press, Cambridge, MA, USA, 1992.
54. H. Youssef, S. M. Sait, and H. Adiche. Evolutionary algorithms, simulated annealing and tabu search: a comparative study. Engineering Applications of Artificial Intelligence, 14(2):167 – 181, 2001. https://doi.org/10.1016/S0952-1976(00)00065-8
55. R. Kumar et al. Blending roulette wheel selection & rank selection in genetic algorithms. International Journal of Machine Learning and Computing, 2(4):365–370, 2012.
56. B. L. Miller and D. E. Goldberg. Genetic algorithms, selection schemes, and the varying effects of noise. Evolutionary Computation, 4(2):113–131, 1996. https://doi.org/10.1162/evco.1996.4.2.113
57. P. A. Devijver and J. Kittler. Pattern Recognition: A Statistical Approach. Prentice-Hall, 1982.
58. R. O. Duda, P. E. Hart, and D. G. Stork. Pattern Classification (2Nd Edition). Wiley-Interscience, New York, NY, USA, 2000.

59. F. Chollet et al. Keras. https://keras.io, 2015.
60. T. O'Malley, E. Bursztein, J. Long, F. Chollet, H. Jin, L. Invernizzi, et al. Kerastuner. https://github.com/keras-team/keras-tuner, 2019.
61. S. Kilicarslan, M. Celik, and Şafak Sahin. Hybrid models based on genetic algorithm and deep learning algorithms for nutritional anemia disease classification. Biomedical Signal Processing and Control, 63:102231, 2021. https://doi.org/10.1016/j.bspc.2020.102231
62. E. L. da Rocha, L. Rodrigues, and J. F. Mari. Maize leaf disease classification using convolutional neural networks and hyperparameter optimization. In Anais do XVI Workshop de Visão Computacional, pages 104–110, Porto Alegre, RS, Brasil, 2020. SBC. https://doi.org/10.5753/wvc.2020.13489
63. M. da Silva, L. Rodrigues, and J. F. Mari. Optimizing data augmentation policies for convolutional neural networks based on classification of sickle cells. In Anais do XVI Workshop de Visão Computacional, pages 46–51, Porto Alegre, RS, Brasil, 2020. SBC. https://doi.org/10.5753/wvc.2020.13479
64. A. Hizukuri, R. Nakayama, M. Nara, M. Suzuki, and K. Namba. Computer-Aided Diagnosis Scheme for Distinguishing Between Benign and Malignant Masses on Breast DCE-MRI Images Using Deep Convolutional Neural Network with Bayesian Optimization. Journal of Digital Imaging, 34(1):116–123, 2021. https://doi.org/10.1007/s10278-020-00394-2
65. S. Kaur, H. Aggarwal, and R. Rani. Diagnosis of parkinson's disease using deep CNN with transfer learning and data augmentation. Multimedia Tools and Applications, 80(7):10113–10139, 2021. https://doi.org/10.1007/s11042-020-10114-1
66. M. Nishio, S. Noguchi, H. Matsuo, and T. Murakami. Automatic classification between covid-19 pneumonia, non-covid-19 pneumonia, and the healthy on chest x-ray image: combination of data augmentation methods. Scientific Reports, 10(1):17532, 2020. https://doi.org/10.1038/s41598-020-74539-2
67. M. H. Saleem, J. Potgieter, and K. M. Arif. Plant disease classification: A comparative evaluation of convolutional neural networks and deep learning optimizers. Plants, 9(10), 2020. https://doi.org/10.3390/plants9101319
68. J. Bergstra and Y. Bengio. Random search for hyper-parameter optimization. Journal of Machine Learning Research, 13(10):281–305, 2012. http://jmlr.org/papers/v13/bergstra12a.html
69. M. Pelikan, D. E. Goldberg, and E. Cantú-Paz. BOA: The Bayesian Optimization Algorithm. In Proceedings of the 1st Annual Conference on Genetic and Evolutionary Computation - Volume 1, GECCO-99, page 525-532, San Francisco, CA, USA, 1999. Morgan Kaufmann Publishers Inc.