White Blood Cell Classification Based on Evolutionary Optimization of Neural Network*

Agnieszka Polowczyk[1,*,†], Alicja Polowczyk[1,†] and Jakub Kancerek[2,†]

[1]Faculty of Applied Mathematics, Silesian University of Technology, Kaszubska 23, 44-100 Gliwice, Poland
[2]Student Scientific Club at the Department of Histology and Cell Pathology, Faculty of Medical Sciences in Zabrze, Medical University of Silesia, Plac Traugutta 2, 41-800 Zabrze, Poland

Abstract

White blood cells (WBC) play a key role in the human immune system and their classification is important in the diagnosis of many diseases and conditions. Previous solutions are mainly based on the use of machine learning algorithms and CNNs. In this work, we propose a non-standard approach using the metaheuristic CMA-ES algorithm as an optimization technique in our classification network, unlike traditional gradient-based approaches. Our solution allows for more effective avoidance of finding oneself in a local minimum and broader exploration. The conducted experiments confirm the superiority of our method in comparison to other algorithms, achieving: Accuracy = 98.75%, Precision = 98.71%, Recall Score = 98.56% and F1 Score = 98.63%.

Keywords

CMA-ES, WBC, Heuristic Optimization, Neural Network

1. Introduction

Leukocytes are dynamic components of the body's immune system, comprising lymphocytes, monocytes, and granulocytes. Lymphocytes and monocytes together form the agranulocyte group. Lymphocytes are the principal cells of the immune system, distributed throughout the body, excluding the central nervous system. They are classified into three primary subtypes: T lymphocytes, B lymphocytes, and natural killer (NK) cells. Their primary function involves the secretion of humoral mediators that facilitate the activation of other immune cells. Monocytes, the second category of agranulocytes, are produced in the bone marrow and circulate in the bloodstream for approximately 8 to 72 hours. Upon migrating to tissues, they differentiate into tissue-specific macrophages, which exhibit characteristics tailored to the respective tissue. Macrophages play a pivotal role in responding to both physiological and pathological stimuli, either produced by host cells or by pathogens. The granulocyte subgroup consists of neutrophil granulocytes - neutrophils, eosinophil granulocytes - eosinophils, and basophil granulocytes - basophils. The functional capabilities of granulocytes are primarily related to their ability to migrate, undergo chemotaxis, degranulate, perform phagocytosis, and generate reactive oxygen species. Neutrophils produce substances that induce smooth muscle contraction and are proficient in phagocytosing bacteria and damaged cells. Neutrophilia, an increased number of neutrophils, is typically observed in bacterial infections. Eosinophils are implicated in allergic responses, inflammatory conditions, and parasitic infections. Basophils play a role in inhibiting coagulation and are also involved in allergic reactions. The leukocyte system is a critical element of the body's defensive mechanisms against antigens that invade the organism. The body faces continuous exposure to viruses, bacteria, parasites, toxins, and foreign substances from the external environment. Therefore, it is crucial to identify which subsets of leukocytes are activated in response to specific antigens, as well as the pathological conditions that may lead to an elevated white blood cell count in the bloodstream [1, 2, 3].

*IVUS 2025: Information Society and University Studies 2025, May 15, Kaunas, Lithuania
*Corresponding author.
†These authors contributed equally.
agnieszkapolowczyk11@gmail.com (A. Polowczyk); alicjapolowczyk47@gmail.com (A. Polowczyk); s88540@365.sum.edu.pl (J. Kancerek)
0009-0008-1583-4493 (A. Polowczyk); 0009-0001-3110-8255 (A. Polowczyk); 0009-0008-6882-1901 (J. Kancerek)

There are many machine learning (ML) methods that play a key role in classification problems, continuous value prediction (regression), or clustering. The key element in the case of using ML methods is the fact that to use these algorithms fully effectively, we need to have a database of images, texts in numerical form. So, in order to analyze various data, they must first be converted into numerical data using various techniques such as: One-Hot Encoding, TF-IDF, Word2Vec. In case of image data, we can flatten each image into a uniform feature vector or extract features from the image using various extraction methods to create a smaller vector and treat it as input to the ML model. The main popular models that are used in data classification are K-Nearest Neighbors, SVC, and Decision Tree. KNN [4] is a lazy algorithm that has no training process. Classification is based on assigning a new, unknown sample to a new class that dominates among its k nearest samples from the training set (k nearest neighbors). Another method mentioned earlier is the Decision Tree [5] which is created based on training data and includes elements such as: nodes (containing conditions), branches (originating from each node) and leaves (containing classification results - classes). To create an effective tree, the entropy measure or Gini index is used. Another popular ML algorithm is SVC (Support Vector Classification) [6]. The main idea of this method is to find a hyperplane that best separates data into classes in the feature space. Moreover, in this algorithm we strive to maximize the margin, i.e. we maximize the distance between this hyperplane and the closest points from both classes. Thanks to this, we have a high level of certainty that new, unknown test data will be correctly classified. Additionally, we can use various kernels that allow us to transform our data to a higher dimension. We use this procedure to make data that are not linearly separable in the original feature space separable in the higher one. Recently, deep learning models using neural networks have been gaining popularity, which can process and analyze data very efficiently. The basic model is the MLP network, which consists of several hidden layers through which the input data passes. This network can model complex and nonlinear dependencies. Extensions of this model are commonly used convolutional neural networks (CNN) [7], recurrent neural networks (RNN) [8], transformers [9], graph neural networks (GNN) [10], which are used in very complex tasks with data typically of spatial (images), irregular (graphs) and sequential structure.

In standard machine learning algorithms, these models do not use gradient optimizers to minimize the cost function. Optimizers are mainly used in neural networks, such as convolutional networks [11] or recurrent networks [12]. Gradient optimizers can be divided into different categories, they are: Gradient Descent [13], Stochastic Gradient Descent (SGD) [14], Adagrad [15], Adadelta [16], RMSProp [17] and Adam [18]. The main task of the above optimizers is to automatically tune the model parameters (weights) during the training process. Gradient Descent (GD) is a basic algorithm that updates the parameters using the entire dataset, in the case of SGD where we use only a specific sample. Adaptive gradient algorithms like Adagrad are characterized by using different learning rates for each iteration, and its improved versions are: Adadelta and RMSProp. Increasingly, researchers are focusing on using metaheuristic algorithms to improve network learning, or they train these networks entirely using different heuristic algorithms, or even use these metaheuristic techniques even to search for appropriate and best network hyperparameters. In [19], scientists proposed to use Particle Swarm Optimization (PSO) to search for the optimal hyperparameter space of the network. In addition, in [20], a genetic algorithm was proposed to optimize the hyperparameters of a convolutional neural network (CNN) in the face recognition problem. Given the growing interest of researchers in using heuristic optimizers in neural networks, we propose a non-standard approach to train our neural network with the Covariance matrix adaptation evolution strategy (CMA-ES) algorithm in the white blood cell classification problem. Our idea assumes complete tuning of weights with an evolutionary algorithm and searching for optimal solutions based on population size. To sum up, our approach focuses on using a lightweight neural network for WBC classification, by extracting the most important features from images, which were then transformed into feature vectors and used as input to our classification network CMA-ES Neural Network. We show high effectiveness of our concept and the influence and selection of appropriate hyperparameters on the quality of network optimization with an evolutionary algorithm, not a classical gradient optimizer. Moreover, our research results confirm the high quality of our idea, outperforming all ML machine learning methods.

2. Methodology

The neural network constructed by us is subjected to the process of optimizing its weights using a non-gradient approach, i.e. we use the metaheuristic CMA-ES method to find the best weights for the white blood cell classification problem excluding lymphocytes. The solution proposed by us assumes the use of a heuristic approach to optimizing weights in a 3-layer neural network with 5540 parameters.

2.1. CMA-ES Optimizer

The CMA-ES method [21] is a covariance matrix adaptation evolution strategy. This algorithm does not operate on derivatives in the optimization problem and is inspired by evolution strategy (ES). The main idea is that individuals with the most favorable features live from generation to generation and pass on good features to the next generation, so evolution takes place on the basis of selection, and the population becomes better and better, i.e. the most adapted to the problem. Evolutionary algorithms assume: mutation, i.e. generating new solutions, selection, i.e. choosing the best solutions, and reproduction and adaptation. In our problem, the initial weight vector is $x \in R^D$, where D is the

Algorithm 1 CMA-ES

Initialize parameters: $x \in R^D$, $\sigma \in R_+^D$, $C = I$, $max\_generations = 1000$;
while $generation < max\_generations$ **do**
  for $k \in 1, ..., \lambda$ **do**
    $z_k \leftarrow \mathcal{N}(0, I)$;
    $x_k \leftarrow x + \sigma C^{\frac{1}{2}} \times z_k$
  end for
  $P \leftarrow$SelectBest$(\mu, \{z_k, f(x_k) \mid 1 \leq k \leq \lambda\})$
  $(x, \sigma, C) \leftarrow$UpdateModel$(x, \sigma, C, P)$
end while
return x

number of all network parameters that initially have the value 0. The next parameter is to set the value of $\sigma \in R_+^D$, i.e. the mutation scale for each parameter (the initial size of the changes). CMA-ES also uses the covariance matrix C, defining the relationship between changes in weights, initially defined as the identity matrix I, i.e. we assume no correlation between weights. The entire scheme of the algorithm is presented in the form of pseudo-code. The algorithm runs until *max_generations* is obtained, i.e. each iteration is a new generation in evolution. First, λ weight samples, i.e. populations, are created. Then, for each new solution x_k, we generate randomness z_k from the normal distribution $\mathcal{N}(0, I)$, and then we create a new version of weights x_k by obtaining mutations, according to the formula:

$$x_k = x + \sigma C^{\frac{1}{2}} \times z_k \tag{1}$$

The next step is to select the best individuals P, i.e. selection, using the SelectBest function, which returns the μ best samples as parents to the next generation, according to the lowest quality score (in our case using Cross Entropy Loss). The last step is to use the UpdateModel function to update the parameters, we take the new x as the new center of the distribution, σ as the new value of the mutation intensity and C the new correlations between the weights, i.e. this process is adaptation. Finally, the best solution x is returned.

2.2. Model architecture

For WBC classification we used a lightweight CMA-ES Neural Network, the structure is presented in Fig. 1. First, each input image was preprocessed to extract the most important n-features. These features were obtained using advanced image processing techniques and we denoted them as: $X \in R^n$. Then, the vector X is the input to the neural network model, which consists of three fully connected layers.

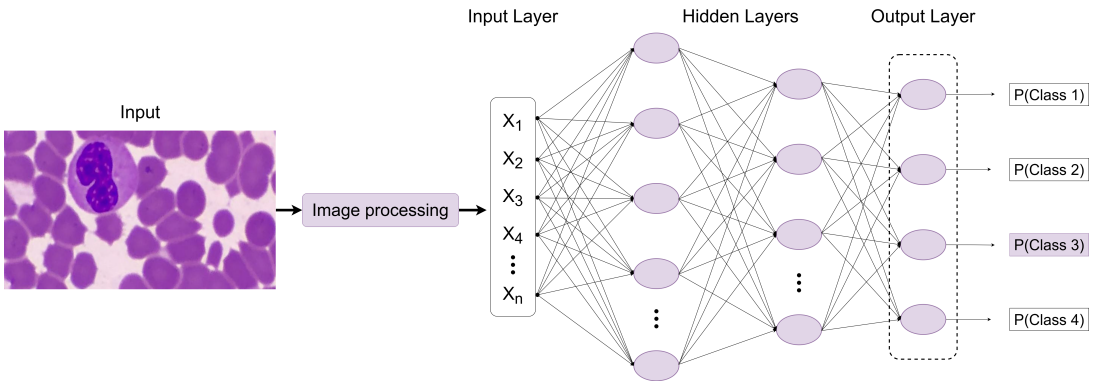

Figure 1: Construction of our CMA-ES Neural Network model, using a three-layer network. The ReLU activation function is used on the hidden layers. Moreover, at the very end, the Softmax activation function is used on the output layer.

The first hidden layer contains 64 neurons with the ReLU activation function. The ReLU activation function is expressed as follows:

$$\mathrm{ReLU}(h) = \max(0, h) \tag{2}$$

where: h is the result of linear transformation of data from the previous layer. The second hidden layer contains 32 neurons also with ReLU activation function. At the very end there is an output layer, which contains $c = 4$ neurons, because we are considering multi-class classification. Additionally, after the output layer, the Softmax activation function was applied, which returned probabilities indicating the affiliation of the classified image to each of the c classes. The sum of these probabilities is 1 and the classification model assigns the image (sample) to the class with the highest probability. We can write the Softmax function using the formula:

$$\mathrm{Softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{c} e^{z_j}} \tag{3}$$

where: z_i is the result from the last layer for the i-th class.

3. Experiments

In this section, we present a detailed description of the dataset, the results of our experiments comparing them with different configurations of the CMA-ES algorithm, and other machine learning algorithms tested on this dataset.

3.1. Dataset

For the study, we used a database containing images of white blood cells from the blood of young individuals of the wild Visayan Warty pig species [22]. In order to increase the number of data, we used augmentation. We used an augmented dataset of images, which contains 4 classes of white blood cells, these are: Monocyte, Basophil, Eosinophil and Neutrophil as presented in Fig. 2. Finally, for our study, we used an augmented database of white blood cells, where only the significant features were extracted from the images and saved in a file in the shared dataset [22]. These features were extracted using advanced image processing techniques.

3.2. Training and Evaluation

For the experiments, we split the data into training and test sets in the ratio of 70:30 and used $batch\_size = 32$. We used the Cross-Entropy Loss function as the objective function needed in the metaheuristic

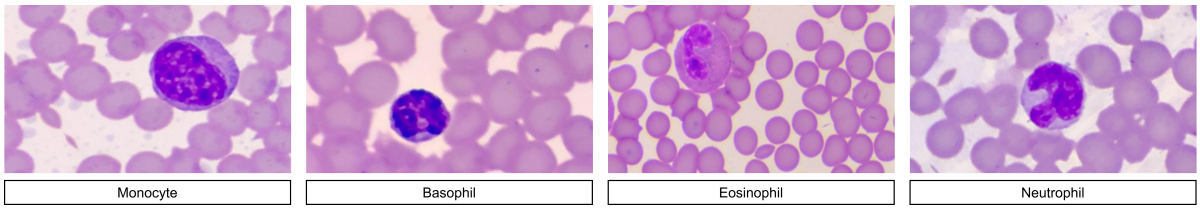

Figure 2: Example images from our image dataset. The database distinguishes 4 classes of white blood cells: Monocyte, Basophil, Eosinophil and Neutrophil. The CMA-ES model was fed with extracted features from images.

algorithm CMA-ES, where y is the true label encoded as one-hot encoding, y' denotes the probabilities obtained by the model:

$$L(y, y') = -\sum_{i=1}^{c} y_i \cdot \log(y_i') \tag{4}$$

3.3. Results

The first of the analyses performed was to properly set the hyperparameters σ and λ. The test results are presented in Table 1. First, the impact of the mutation scale on the accuracy result was analyzed; the larger σ, the more drastic the changes, so it causes instability and difficulty in convergence. For $\sigma = 0.01$, the model trains slowly and fine-tunes the weights more precisely, so the best result was obtained for the value $\sigma = 0.02$. The next modification was to check the impact of the population size λ; the larger, the greater the range of solutions searched for; too small a value can lead to a situation of getting stuck in a local minimum. The best model accuracy was obtained for $\sigma = 0.02$ and $\lambda = 20$, because we achieved a good balance between exploration and exploitation. Additionally, we present the results for these configurations in the form of a loss function and a confusion matrix in Figs. 3 and 5.

Table 1
Results of our CMA-ES Neural Network model for different hyperparameters in evolutionary optimization.

Hyperparameters	Accuracy (%)	Precision (%)	Recall Score (%)	F1 Score (%)
σ = 0.01, λ = 10	96.49	96.91	96.36	96.59
σ = 0.02, λ = 10	97.24	96.86	97.11	96.97
σ = 0.05, λ = 10	90.48	90.59	89.99	90.11
σ = 0.02, λ = 20	**98.75**	**98.71**	**98.56**	**98.63**
σ = 0.02, λ = 50	97.99	98.22	98.11	98.14

Table 2
Results of our CMA-ES Neural Network model compared to other machine learning models. Comparison of all models in terms of performance metrics: Accuracy, Precision, Recall Score, F1 Score.

Models	Accuracy (%)	Precision (%)	Recall Score (%)	F1 Score (%)
Linear Discriminant Analysis	82.96	84.73	83.04	83.71
Gaussian Process Classifier	83.20	86.98	83.78	85.08
KNeighborsClassifier	91.47	89.53	87.83	88.46
NuSVC	92.73	91.50	89.25	89.97
DecisionTreeClassifier	92.98	93.16	93.17	93.11
GradientBoostingClassifier	93.73	92.93	91.96	92.21
LinearSVC	93.98	94.20	93.21	93.53
VotingClassifier	94.98	95.17	94.96	95.03
SelfTrainingClassifier	95.48	95.48	95.51	95.67
SVC	96.99	94.88	93.95	94.27
HistGradientBoostingClassifier	96.99	97.06	96.62	96.81
StackingClassifier	96.99	96.96	96.95	96.81
Adam Neural Network	97.74	97.58	97.91	97.71
CMA-ES Neural Network	**98.75**	**98.71**	**98.56**	**98.63**

A slow trend is noticeable for low values of σ. For $\sigma = 0.02$, a loss function value of 0.4 is reached for epoch 300, for $\sigma = 0.01$, this value appears only at about epoch 600. For $\sigma = 0.05$, aggressive changes of the loss function value are visible, confirming the fact of a too large value, which is inappropriate for this type of problem. In case of changing λ, the effect is not so noticeable between the values: for 10, 20 and 50, the plots differ only in the stability of the loss, the higher the value of λ, the less susceptible the plot is to instabilities and jumps, but therefore leads to higher computational complexity. Fig. 5 presents the classification results for individual classes, each of the models achieved good results, but the best configuration for $\sigma = 0.02$ and $\lambda = 20$ achieved accuracy close to 100% for all classes. Fig. 4 presents a visualization of the multidimensional feature space of data for two states, first before processing by the CMA-ES Neural Network model and after the output. The popular t-SNE feature dimensionality reduction was used, thanks to which it was possible to map the vectors to a two-dimensional space. The presented results confirm that our CMA-ES Neural Network model is able to correctly predict the appropriate classes for samples, before the model the data is scattered, and then after processing by the network, noticeable boundaries between groups were created. Additionally, in Table 2 we present a comparison of our model with other popular machine learning methods. The Linear Discriminant Analysis and Gaussian Process Classifier performed the worst in the whole comparison. Our model achieved Accuracy = 98.75%, Precision = 98.71%, Recall Score = 98.56% and F1 Score = 98.63% and outperformed other methods with higher scores for all performance metrics. The neural network trained using the Adam optimizer (Adam Neural Network) achieved lower average results than the results of the neural network trained using the CMA-ES metaheuristic algorithm (CMA-ES Neural Network).

4. Conclusion

We propose a non-standard and unpopular approach to optimizing neural networks, using the metaheuristic algorithm Covariance Matrix Adaptation Evolution strategy (CMA-ES). We have shown that the use of a lightweight and uncomplicated neural network in the problem of white blood cell clas-

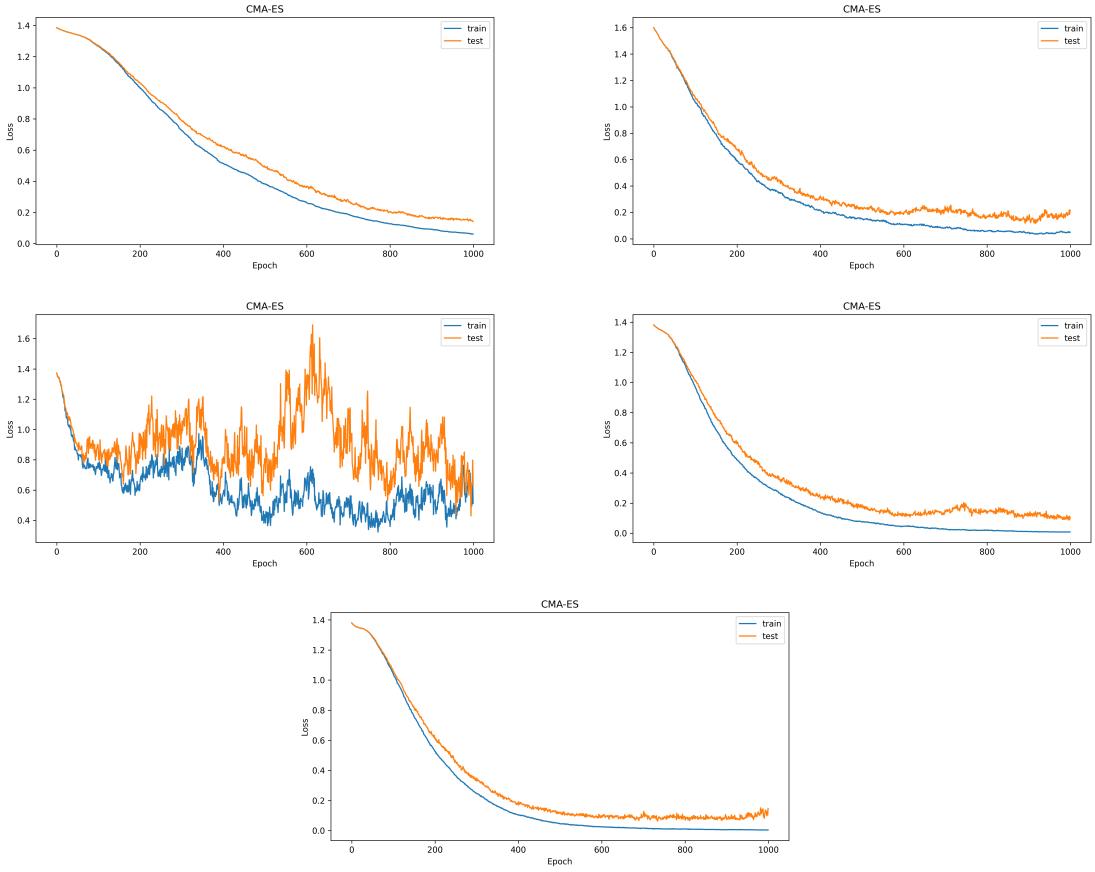

Figure 3: Plots of loss function values for models trained with the CMA-ES metaheuristic algorithm for different hyperparameters. In the first line: left - model with $\sigma = 0.01$ and $\lambda = 10$, right - model with $\sigma = 0.02$ and $\lambda = 10$. Second line: left - model with $\sigma = 0.05$ and $\lambda = 10$, right - model with $\sigma = 0.02$ and $\lambda = 20$. The last plot shows the CMA-ES trained model with $\sigma = 0.02$ and $\lambda = 50$.

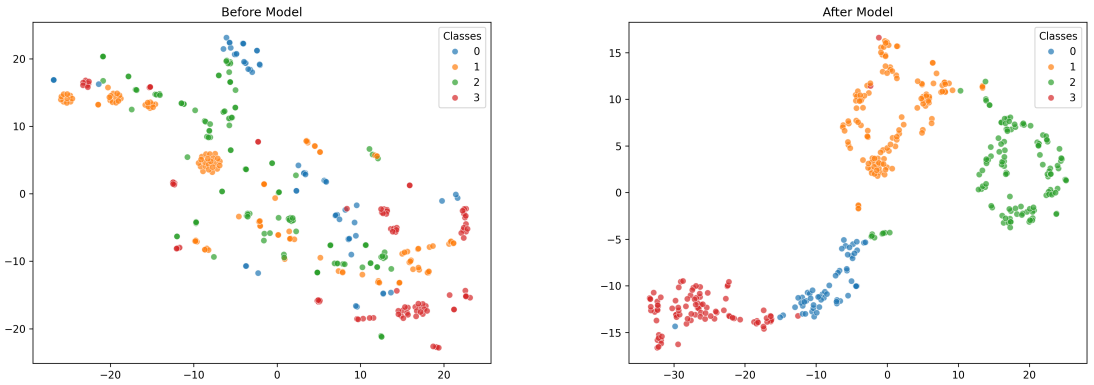

Figure 4: Visualization of high-dimensional data using the t-SNE technique applied before processing by the model and after passing through the network.

sification gives excellent results, outperforming other machine learning methods. Additionally, an important aspect is the selection of appropriate hyperparameters in the CMA-ES algorithm, so that it correctly and quickly finds the most optimal solution (weight values). To sum up, our idea, i.e. the use of metaheuristics to optimize a large number of parameters (about 5000) has obtained excellent results, becoming an alternative method compared to other, classical gradient approaches.

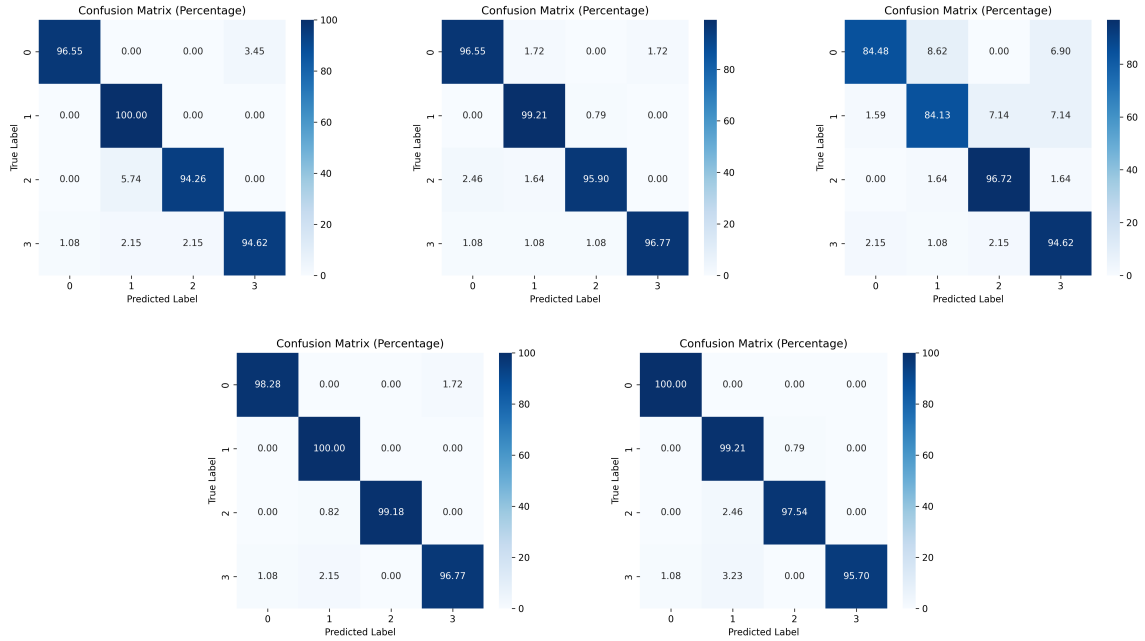

Figure 5: Confusion matrices for models trained with the CMA-ES metaheuristic algorithm for different hyperparameters. In the first line: left - model with $\sigma = 0.01$ and $\lambda = 10$, right - model with $\sigma = 0.02$ and $\lambda = 10$. Second line: left - model with $\sigma = 0.05$ and $\lambda = 10$, right - model with $\sigma = 0.02$ and $\lambda = 20$. The last plot shows the CMA-ES trained model with $\sigma = 0.02$ and $\lambda = 50$.

Acknowledgments

The authors would like to thank the Rector of the Silesian University of Technology for support on this research project under IDUB initiative.

Declaration on Generative AI

The author(s) have not employed any Generative AI tools.

References

[1] J. Gołąb, Immunologia, Wydawnictwo Naukowe PWN, Warsaw, 2023.
[2] S. J. Konturek, Fizjologia, Elsevier Urban Partner, Wroclaw, 2021.
[3] P. Gajewski, A. Szczeklik, Interna Szczeklika 2024, Medycyna Praktyczna, Krakow, 2024.
[4] P. K. Syriopoulos, N. G. Kalampalikis, S. B. Kotsiantis, M. N. Vrahatis, knn classification: a review, Annals of Mathematics and Artificial Intelligence (2023).
[5] K. Mittal, K. S. Gill, D. Upadhyay, V. Singh, S. Aluvala, Applying machine learning for autism risk evaluation using a decision tree classification technique, in: 2024 2nd International Conference on Computer, Communication and Control (IC4), 2024, pp. 1–6.
[6] M. Alizadeh, M. Mehdi Ebadzadeh, Kernel evolution for support vector classification, in: 2011 IEEE Workshop on Evolving and Adaptive Intelligent Systems (EAIS), 2011, pp. 93–99.
[7] R. Yamashita, M. Nishio, R. K. G. Do, K. Togashi, Convolutional neural networks: an overview and application in radiology, Insights into Imaging 9 (2018) 611–629.
[8] J. Oruh, S. Viriri, A. Adegun, Long short-term memory recurrent neural network for automatic speech recognition, IEEE Access 10 (2022) 30069–30079.
[9] W. Ullah, K. Javed, M. A. Khan, F. Y. Alghayadh, M. W. Bhatt, I. S. Al Naimi, I. Ofori, Efficient

identification and classification of apple leaf diseases using lightweight vision transformer (vit), Discover Sustainability 5 (2024) 116.
[10] Z. Wu, Z. Chen, S. Du, S. Huang, S. Wang, Graph convolutional network with elastic topology, Pattern Recognition 151 (2024) 110364.
[11] G. Sathiyapriya, S. A. Shanthi, Image classification using convolutional neural network, in: 2022 First International Conference on Electrical, Electronics, Information and Communication Technologies (ICEEICT), 2022, pp. 1–4.
[12] Y. Chen, J. Li, Recurrent neural networks algorithms and applications, in: 2021 2nd International Conference on Big Data Artificial Intelligence Software Engineering (ICBASE), 2021, pp. 38–43.
[13] Q. Tran-Dinh, M. van Dijk, Chapter 1 - gradient descent-type methods: Background and simple unified convergence analysis, in: L. M. Nguyen, T. N. Hoang, P.-Y. Chen (Eds.), Federated Learning, Academic Press, 2024, pp. 3–28.
[14] J. Azimjonov, T. Kim, Stochastic gradient descent classifier-based lightweight intrusion detection systems using the efficient feature subsets of datasets, Expert Systems with Applications 237 (2024) 121493.
[15] J. C. Duchi, E. Hazan, Y. Singer, Adaptive subgradient methods for online learning and stochastic optimization, J. Mach. Learn. Res. 12 (2011) 2121–2159.
[16] M. D. Zeiler, ADADELTA: an adaptive learning rate method, CoRR abs/1212.5701 (2012).
[17] R. Elshamy, O. Abu-Elnasr, M. Elhoseny, S. Elmougy, Improving the efficiency of rmsprop optimizer by utilizing nestrove in deep learning, Scientific Reports 13 (2023) 8814.
[18] D. P. Kingma, J. Ba, Adam: A method for stochastic optimization, in: 3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings, 2015.
[19] H. S. Salem, M. A. Mead, G. S. El-Taweel, Particle swarm optimization-based hyperparameters tuning of machine learning models for big covid-19 data analysis, Journal of Computer and Communications 12 (2024) 160–183.
[20] K. Namrata, M. Palak, A. Pawanesh, L. P. K., A genetic algorithm based optimized convolutional neural network for face recognition, International Journal of Applied Mathematics and Computer Science 33 (2023) 21–31.
[21] N. Hansen, The cma evolution strategy: A tutorial, 2023. arXiv:1604.00772.
[22] J. R. Alipo-on, F. I. Escobar, J. L. Novia, M. M. Atienza, S. Mana-ay, M. J. Tan, N. AlDahoul, E. Yu, Dataset for machine learning-based classification of white blood cells of the juvenile visayan warty pig, 2022. URL: https://dx.doi.org/10.21227/3qsb-d447. doi:10.21227/3qsb-d447.