



The model architecture search system for chromosome image classification

Nurullah Calik¹

Received: 13 December 2023 / Revised: 3 February 2024 / Accepted: 10 February 2024 / Published online: 14 March 2024
© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2024

Abstract

The classification of chromosome images holds immense significance in the fields of genetics, clinical diagnostics, and medical research. It plays a pivotal role in the precise identification of genetic abnormalities, allowing for early and accurate diagnosis of various genetic disorders and birth defects. The automation of this process offers significant advantages in terms of time and human resource savings. This study introduces the Model Architecture Search System (MASS), designed to adapt itself to classify chromosome images for karyotyping. The MASS framework aims to construct an optimal model architecture for the specific classification task by leveraging predefined CNN backbones, activation functions, and loss functions. There are 12 pre-trained networks, 5 activation functions, and 2 loss functions in the selection set of the MASS. The proposed framework utilizes the Tree-structured Parzen Estimator (TPE) algorithm based on Bayesian Optimization, eliminating the need for manual model searching processes and finding optimal model architecture. The suitable model structure for the relevant dataset is generated from these groups automatically by using TPE. Experiments conducted on two distinct datasets demonstrate the superior performance achieved by this proposed mechanism.

Keywords Chromosome images · Deep learning · Classification · Bayesian optimization · Tree-structured Parzen estimator

1 Introduction

The karyotyping is a lab procedure used to examine chromosomal count, size, and shape in cytogenetics. It is a useful method for detecting genetic diseases and chromosomal anomalies [1]. Geneticists and medical professionals can diagnose chromosomal diseases, such as Turner syndrome, Down syndrome, and many others by investigating karyotypes [1, 2]. To detect abnormalities, karyotyping involves arranging pairs of chromosomes from 1 to 22, followed by gonosomes, or sex chromosomes (XX: female and XY: males), based on their size. However karyotyping offers an invaluable opportunity to extract information from chromosomal images, it is a labor-and time-intensive procedure. Sorting through the chromosomes, which are rolled up and overlapped, one by one, and making decisions based on band transitions is a tedious process. As the volume of genetic and cytological data increases, there is a huge demand for

automated and accurate chromosome identification systems. The many research on automatic karyotyping has been conducted to reduce this workload [3–7]. At the beginning of an end-to-end karyotyping system, the chromosome detection problem is handled, and then the images are classified for pairing [8]. Deep learning models have achieved significant performances in the classification stage in the karyotyping process [9–11].

Pre-trained DNN architectures are employed for diverse image classification challenges, leveraging their intrinsic capabilities in domain adaptation. Each model, characterized by a distinctive graph architecture, yields varying outcomes across a spectrum of tasks. Hence, the identification of a suitable model for the relevant problem necessitates numerous iterative trial-and-error procedures, entailing a labor-intensive workload. The main motivation of this paper is to build optimal DNN models for karyotyping by mitigating this iterative burden. The primary objective of this research is to build an automated approach, based on Bayesian Optimization (BO) principles, for the systematic searching of an optimal model tailored to chromosome classification datasets. Within this framework, users are afforded a streamlined process by which they input the names of their

✉ Nurullah Calik
nurullah.calik@medeniyet.edu.tr

¹ Department of Biomedical Engineering, Istanbul Medeniyet University, 34700 Istanbul, Turkey

preferred pre-trained models, along with the desired training and testing ratios, into the system. Subsequently, the system autonomously selects a suitable pre-trained model for the specific training dataset. The Model Architecture Search System (MASS) conducts the seeking mechanism for an appropriate pre-trained model. Moreover, the essential properties of MASS are extended beyond model selection by covering the identification of an optimal loss function and activation function deemed suitable for the given dataset. The consideration of loss and activation functions is crucial, because those associated with the pre-trained model may not be optimal with the task at hand.

1.1 Related works

Within the area of automatic karyotyping investigations, the field is typically divided into two categories: the first being the detection/segmentation of chromosomes, and the second involving the classification of chromosomes. In this study, because the primary focus is the classification task, greater emphasis has been given to the examination of image classification studies.

Minaee et al. [12] introduced a method characterized by its geometric approach for the segmentation of partially overlapping chromosomes. This method comprises two separate phases. The initial phase is dedicated to the detection of chromosome clusters that cover chromosomes exhibiting either contact or partial overlap. This detection process relies on the application of three fundamental geometric criteria related to the geometry of the chromosomes, specifically, those based on the surrounding ellipse, convex hull, and skeleton and end points methodologies. Then, in the second phase, a separation strategy utilizing a defined cut point is employed to isolate individual chromosomes within the identified clusters. The application of this method yielded an accuracy rate of 91.9% when applied to a dataset comprising 62 partially overlapping and touching chromosomes. Another segmentation work is presented in [13]. The authors proposed a U-net architecture [14] with specific adaptations to enhance model training. The dataset used in their study comprised 40 metaphase images gathered from Renji Hospital. To mitigate overfitting, the authors implemented data augmentation techniques. Prior to augmentation, the dataset images were divided into subsets of 25 for training, 5 for validation, and 10 for testing. Following the augmentation process, the dataset size expanded to 3500 images for training and 700 images for validation. Their approach yielded 96.97% Dice similarity coefficient achievement. One of the leading studies on chromosome classification is given in [15]. Swati et al. introduced a novel approach based on Siamese networks, where these networks base their training on comparing pairs of chromosomes. The proposed model achieved 84.6% accuracy over a dataset that contains 1740 images. The Chromenet model,

as introduced by Menaka et al. [9], is designed for the classification of metaphase Q-banded chromosome images with a resolution of 64x64 pixels. The authors conducted a comprehensive comparison of commonly used optimizers in the literature and found that the ADAM optimizer outperformed others. Their study utilized the dataset provided by BioImLab [11], and it reported 93.1% accuracy as the highest achieved score, validated through a tenfold cross-validation process. In another significant paper [16] concerning chromosome classification, Lin et al. presented an impactful study that utilized the original CIR dataset and the Inception-Resnet-v2 model for their analysis. CIR-Net uses a rotation-based data augmentation method to increase model performance. The presented model is designed to process images with dimensions of 224×224 pixels. In the paper, for the highest achieved metrics, 95.98% accuracy and 96% F1 score are reported. Given the relatively low resolution of the BioImLab dataset, addressing the image classification across 24 distinct classes is a challenging task. To overcome this struggle, Menanka et al., as proposed the use of a Laplacian pyramidal super-resolution network (LaPSRN) as a preprocessing step for the images before classification in [17]. This novel approach enhanced the image quality for classification by upscaling them to higher resolution levels. The authors noted a significant performance improvement of 3% over the baseline model with the implementation of this pre-processing technique. Moreover, the study included a hybrid classification approach that utilizes feature vectors extracted from the CNN model and then classifies them by using a support vector machine (SVM). Remarkably, this framework yielded the highest performance rate, achieving 94.6% accuracy in classification tasks on the BioImLab dataset. In this study, it is also stated that the Swish [18] activation function is preferred instead of the usual ReLU as the activation function. In another study [10], both CIR and BioImLab data sets are used. The CNN models that form the basis of the study are ResNet50 and SENet50. Among the residual layers, the authors chose to use the Leaky ReLU activation function. With the proposed framework, the highest 92.97% accuracy, 93.00% recall, and 92.98% precision are obtained for the CIR dataset, while these values are expressed as 92.58%, 93.08%, and 92.57% for the BioImLan dataset. Another significant study was presented by Al-Kharraz et al. [8]. This research introduced a comprehensive end-to-end system designed for karyotyping. The methodology involved the initial detection of chromosomes within metaphase images through the implementation of YOLOv2. After this detection process, a fine-tuned version of the VGG19 model was developed for the classification of these detected chromosomes. The results of the study demonstrated an achievement of 94.11% accuracy over the BioImLab dataset.

1.2 Contributions

Contributions of this paper are listed as follows:

- The proposed fully automated method seeks a proper pre-trained model for chromosome classification. User-defined parameters are only the lists of pre-trained models, activation functions, and loss functions.
- The MASS strives to find a suitable model over the set given for training using TPE that is based on Bayesian Optimisation.
- The proposed framework also searches for the activation function that is optimum for the problem instead of the activation that comes with the pre-trained model.
- The loss function is selected by MASS and end-to-end model architecture search is performed.

2 Materials and methods

2.1 Data set

Two distinct datasets were employed in this research to carry out the study cases. The first of them, referred to as the CIR-Net dataset, is detailed in [16], while the second one is provided by the BioImLab Group in [11].

The CIR dataset, as described in [16], was generated at the Medical Genetic Center and the Maternal and Children Metabolic-Genetic Key Laboratory of Guangdong Women and Children Hospital. It aims to advance the field of automated chromosome karyotyping and facilitate research endeavors. The dataset contains 65 normal karyotypes. Each normal karyotype comprises a complement of 46 chromosomes. Thus, the dataset involves a total of 2990 chromosome samples which are excreted from 32 male and 33 female subjects. Chromosomes 1 to 22 are identified as paired, while chromosome 23, indicative of gender, is considered singular. The images of chromosomes were captured at a resolution of 8 bits per pixel, and their dimensions conform to a standardized size of 224×224 pixels. Some samples of the CIR dataset are given in Fig. 1a. The number under the images presents the chromosome ID.

The BioImLab chromosome dataset for classification (BIL) was presented in [11]. The dataset was collected by expert cytologists with specialized knowledge and expertise in the field. The dataset contains 5,474 chromosome images, organized into 119 specific cell samples. Within each cell folder, a set of 46 chromosomes is represented, reflecting the typical karyotype complement found in human cells. Chromosome image sizes are different from each other. Hence, in this paper, each chromosome is placed in the middle of a 224×224 template and brought to the sizes required by pre-trained models. Illustrations of the BIL dataset are depicted

in Fig. 1b. It is discernible from these exemplars that the BIL dataset exhibits a comparatively lower resolution. The characteristic band transitions that serve as identifying features of chromosomes are notably faint. This makes their visual classification substantial complexity and demands manual effort.

2.2 TPE-based fully-automated model architecture search

2.2.1 The theoretical background of TPE

The TPE [19–21] is a Bayesian Optimization [22] algorithm widely used for hyperparameter tuning and global optimization of expensive-to-evaluate functions. It is a probabilistic model-based optimization technique that uses a tree-structured algorithm to efficiently search for the optimal set of hyperparameters. The algorithm iteratively refines its probability models by sampling new configurations, evaluating them, and updating the probability densities. It selects the next candidate parameters by optimizing the acquisition function. In contrast to the grid and random search strategies, TPE distinguishes itself by not evaluating specific statistical points within the hyperparameter space. TPE, as an instance of SMBO, adopts a Gaussian Process (GP)-based [23] approach, incorporating a tree structure. Within TPE, different functions are used for constructing the surrogate model, evaluating between regions above and below a specified threshold value [19, 20, 24].

Specifically, TPE's surrogate model is established over the predefined domain of the optimization problems which encompasses the pre-trained model selection, activation function determination, and loss function identification. The principal objective of the GP within TPE is to construct a surrogate model capable of diminishing the variance in the output response at unobserved points residing over the optimization domain. As each iteration progresses, TPE seeks to estimate the optimal hyperparameter point based on the growing ensemble of previously observed data points. Hence, as the number of observations accumulates, the surrogate model's average response tends to converge [24].

The main goal of a hyperparameter estimation algorithm is to identify the optimal set of hyperparameters that yield $\theta^* = \underset{\theta \in \Theta}{\operatorname{argmin}} \mathcal{L}(\theta)$. To accomplish this, TPE relies on sequential observation pairs (\mathcal{S}_i, y_i) . Herein, $\mathcal{S} \triangleq \{\theta_1, \theta_2, \theta_3, \dots, \theta_K\}$ represents K hyperparameter set of θ s, y refers the loss value, i indicates index of trials. The set of observed pair is $\mathcal{D}_{1:t} \triangleq \{(\mathcal{S}_1, y_1), (\mathcal{S}_2, y_2), \dots, (\mathcal{S}_t, y_t)\}$. TPE aims to find next the best hyperparameter set \mathcal{S}_{t+1} by using Bayesian approach over $P(\mathcal{S}_{t+1} | \mathcal{D}_{1:t})$. The TPE is fundamentally built on probability density estimation. It models the objective function as a probability distribution over the hyperparam-

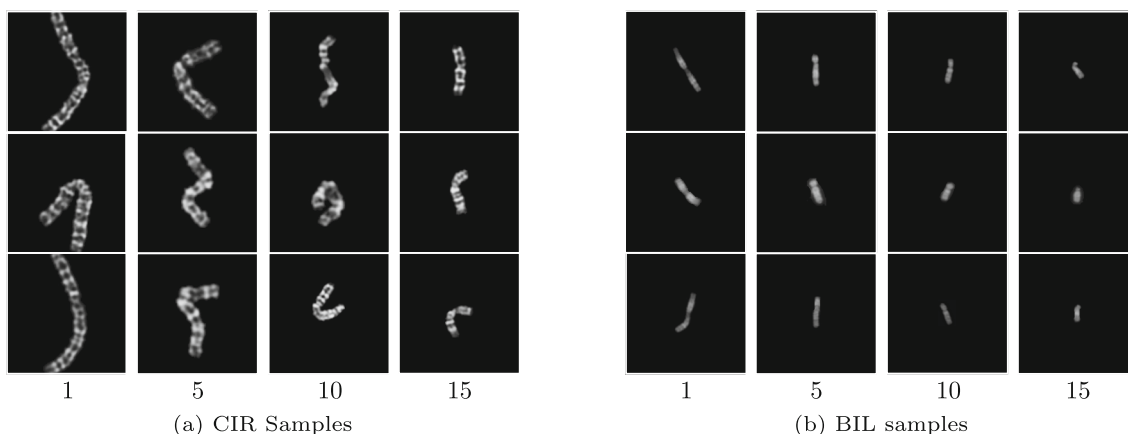


Fig. 1 Examples of chromosomes 1, 5, 10, and 15 from the CIR and BIL datasets. CIR dataset has a higher resolution than BIL. The bands within the chromosome can be seen more clearly. In addition, while the

images are 224×224 in the CIR dataset, they are in different sizes in the BIL. To make both data sets the same size, BIL samples are placed in the middle of a 224×224 template

ter space. The key idea is to maintain two probability density functions given in Eq. 1.

$$p(\mathcal{S} | \mathcal{D}) = \begin{cases} \ell(\mathcal{S}) & \text{if } y < y^* \\ g(\mathcal{S}) & \text{if } y \geq y^* \end{cases} \quad (1)$$

$\ell(\mathcal{S})$ represents the conditional probability density of observing a configuration \mathcal{S} given that its objective function value y is less than or equal to the best observed value y^* . This models the better region where good results can be obtained. However, $g(\mathcal{S})$ is used worst region. These conditional probability density functions are updated iteratively as more data points are collected during the optimization process. Due to its ability to recommend more promising candidate hyperparameters for evaluation, TPE facilitates a clearly swifter recovery of the loss function's value in comparison to conventional random or grid search methods. This accelerated convergence contributes to an overall reduction in the loss function's evaluation. Although TPE allocates additional computational resources to select the next set of hyperparameters in order to enhance model performance, it emerges as a more sensible approach when considering the aggregate time expended in comparison to grid and random search methods. Hence, it has a crucial role in decreasing the effect of human intervention in the model design process, effectively removing the manual searching from the loop [19, 25]. The TPE is also used for categorical parameters. These parameters are paired with an integer number and given to the algorithm. In this study, TPE uses pre-trained model, activation, and loss function flags as $\mathcal{S} \triangleq \{f_m, f_a, f_l\}$. The Hyperopt [20] Python library is preferred to establish the TPE mechanism.

2.2.2 The model architecture search system based on TPE

There are two separate stages in the MASS structure, where the user enters parameters and the framework searches for a model for chromosome classification.

In the Parameter Definition Stage (PDS) (Fig. 2a), users are tasked with the specification of sets comprising pre-trained models, activation functions, and loss functions. Users are only required to provide a list of parameter names corresponding to these sets. In this framework, the MASS automatically retrieves models from the Pytorch Image Models (TIMM) [26] library. Additionally, users are expected to specify a training-to-test data ratio denoted as P%, which is utilized to create training and test datasets. Notably, efforts are made to preserve this ratio within each class, even in the presence of imbalanced data sets. Furthermore, users have the flexibility to indicate the number of folds to be employed for cross-validation at this stage. For the specific investigation in this work, a twofold cross-validation approach is adopted. A total of 12 pre-trained models, 5 activation functions, and 2 loss functions are selected to constitute the sets (Table 1). These selections are represented by a set of integer flags, denoted as f . For pre-trained model set, $f_m \in [1, 12]$, activation function set $f_a \in [1, 5]$ and loss functions $f_l \in \{1, 2\}$ notation is used. The union of these three flags constitutes the set denoted as \mathcal{S} . Within the set \mathcal{S} , the parameters associated with the corresponding flag values are utilized to build the DNN model. The objective of TPE is to identify the most suitable subset, denoted as \mathcal{S}^* for the given classification task.

In the Model Search Stage (MSS), initially, TPE randomly selects an initial point denoted as \mathcal{S}^0 . A model is built based on the parameters within \mathcal{S}^0 . After this process, a twofold cross-validation process is initiated. In this phase, the model

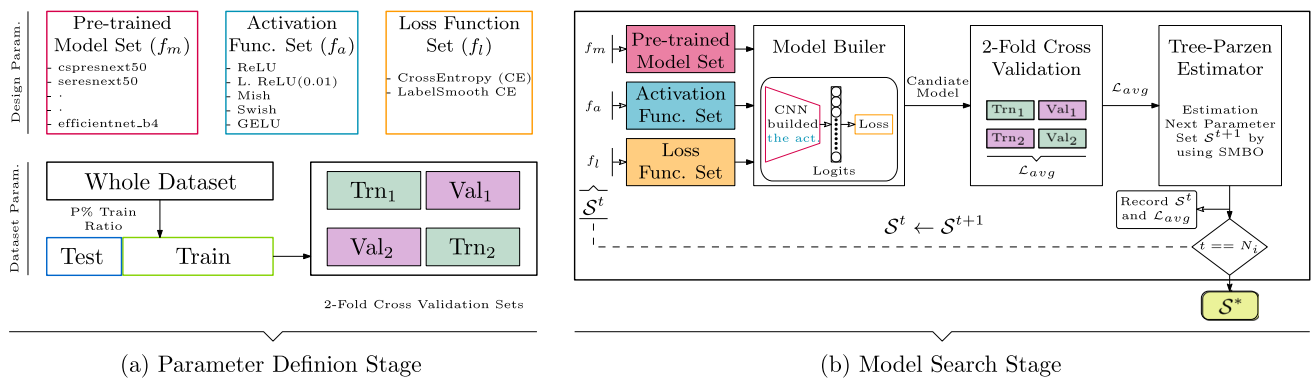


Fig. 2 The Model Architecture Search System (MASS) block diagram. **a** In the User Defined Parameter stage, MASS needs two types of parameters from the user. In stage **b**, MASS starts the search process using

the Tree-Parzen Estimator over the determined design param. sets. At the end of each iteration, the best one among all recorded S^t sets is exported as the optimum model parameters (S^*)

Table 1 The design parameter sets that necessitate the definition by the user

Model	Flag ID	Activations	Flag ID	Loss Func.	Flag ID
cpresnext50	1	ReLU	1	Cross entropy	1
seresnext50_32x4d	2				
tv_resnext50_32x4d	3	LeakyReLU	2		
densenet121	4				
inception_v4	5	Mish	3		
inception_resnet_v2	6				
efficientnet_b4	7			Label smoothing	2
tf_efficientnetv2_m_in21k	8	SiLU	4		
seresnext26d_32x4d	9				
tf_efficientnet_b0	10				
regnetx_064	11	GELU	5		
regnety_064	12				

Specifically, the work encompasses 12 different pre-trained models, 5 activation functions, and 2 loss functions. These sets can be configured for other classification tasks. The Tree-Parzen Estimator methodology operates on the basis of integer flag ID numbers to generate predictive outcomes

is trained according to hyperparameters set to 10 epochs, 24 mini-batch sizes, and a learning rate of 0.001. In addition, while each mini-batch size is acquired during the training phase, the input images are rotated with the degree obtained from a uniform distribution between $[-30, 30]$ degrees and applied in data augmentation. The TPE loss is used as $\mathcal{L}_{avg} = 1 - F_{avg}$ during model validation. This means that the average of the validation F-score values obtained from the models forms the basis of TPE’s loss. To facilitate the minimization of the TPE loss function, the avg F-score is subtracted from 1. In a single iteration of TPE, the loss value obtained through twofold cross-validation is associated with the S^0 parameters and recorded. Then, TPE begins its operation to estimate S^1 . The same process continues after calculating the appropriate set of S for the next step. In this study, a value of 30 is chosen for the TPE iteration. After the MSS is completed, TPE provides the output in the form of model architecture parameters (S^*) that yield the best validation loss.

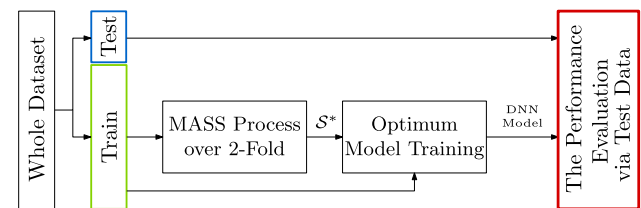


Fig. 3 The evaluation of the models, which have been identified through the MASS procedure, is conducted over test data. Following the training of these optimal models on the complete training dataset, their performance is assessed using the independent test dataset

3 Experimental results and discussion

The MASS approach is tested on two commonly employed datasets within the literature. The first dataset, denoted as the CIR dataset, comprises a total of 2990 data instances, as previously noted. The second dataset, referred to as the

BIL dataset, consists of 5474 chromosome samples for 24 classes. As depicted in Fig. 1, the BIL dataset presents a more intricate band interweaving pattern that makes the problem harder. Otherwise, images in the CIR dataset have more clear chromosome band transitions. Hence, an initial phase of the evaluation procedure focuses on assessing the intrinsic performance characteristics of the MASS methodology. Then, comparative evaluation vis-à-vis existing similar deep learning-based studies in the literature.

Following the completion of both the PDS and MSS, the process of determining the optimal model for the training data is finalized. Subsequently, the model undergoes a testing phase wherein its performance is evaluated (Fig. 3). During this testing phase, the model is reconfigured with the optimized parameters denoted as (S^*). In this step, the model is trained using the entire training dataset. Then, performance metrics and results are acquired through the evaluation of the model's predictive capacity when tested against the independent test dataset.

3.1 The methodology for experimental investigations

While performing evaluations on the previously described datasets, the allocation of training and testing samples is established based on the configuration recommended by previous works, specifically adhering to an 80:20 split for the CIR dataset and a 70:30 division for the BIL dataset, as mentioned in [16] and [9] respectively. In order to evaluate the robustness of the MASS, performance metrics are reported across 10 individual random trials. The partitioning of data is executed by using the scikit-learn Python library [27], utilizing the seed parameter of the random number generator (RNG). This seed parameter, when set to a fixed value, ensures a similar separation of training and testing samples during each execution on the same seed value. This is an important approach for programs to be reproducible. In the context of this study, integer values within the range [0,9] are selected as seeds for the RNG. The MASS procedure is run independently for each randomized trial. This approach facilitated an investigation into the model structures built by MASS under varying data distributions.

As comparison metrics, accuracy (ACC), precision (PRE), recall (RCL), and macro F-Score (F_{macro}) are used. These values are obtained from the confusion matrix. The benefit of the confusion matrix ($\mathbf{K} \in \mathbb{N}^{24 \times 24}$) lies in illustrating the model's ability to accurately classify the true positive class from the others. In the MASS training, there is an optimum model training phase after finding S^* (Fig. 3). The pre-trained model selected at the end of TPE is trained over a total of 20 epochs. 0.001 is used for the initial learning rate, and this value is multiplied by 0.75 after each epoch. Adam is preferred for the optimizer. Mini-batch size is handled as 24.

The specifications of the platform used for training can be listed as i7-6700K 4.00 GHz CPU, 16 GB RAM, and Nvidia 1080Ti GPU.

3.2 The performance analysis and comparison of MASS

The initial performance evaluation of the MASS system is evaluated by the Model Search Stage (MSS), commencing after the establishment of key parameters. In this phase, the TPE algorithm endeavors to identify the optimal model configuration for the prevailing data distribution through a twofold cross-validation process applied to the training set. Figure 4 illustrates the progression of average F-score values generated in the MSS across three different RNG seeds (0, 2, and 9).

At the beginning of the MSS, TPE initiates its search by randomly selecting the S^0 configuration and iteratively improves its approach. It is noteworthy that TPE does not consistently converge to a superior solution with each iteration, as evidenced by the oscillations evident in Fig. 4. Within each TPE iteration, a triad consisting of a pre-trained model, an activation function, and a loss function is experimented with. Once the predefined iteration count of $N_t = 30$, as preferred in this paper, after ending the TPE process, the optimal model parameters are reported as the final output. The whole training data is used in the last training phase where the S^* model is built. The circular markers on Fig. 4 show the iteration in which the S^* model is found.

After the completion of the MSS phase, the resultant models, trained for each RNG seed are handled for comprehensive testing. The outcomes derived from the proposed MASS algorithm, when applied to the CIR and BIL datasets, are detailed in Tables 2 and 3, respectively. These results present the architectural configuration of the model determined by MSS for each RNG seed, along with the performance metrics generated by this particular model. It should be noted that due to the differing partition of training and testing datasets at varying RNG values, the MASS algorithm yields divergent model architectures.

It is considerable that, despite the variability in training and testing data, the preferred models identified by MASS consistently exhibit a superior level of performance. This is also manifested in the results, which not only showcase a notable degree of accuracy (ACC) but also demonstrate a balance between precision (PRE) and recall (RCL) values.

Table 2 demonstrates a significant facet of the study, namely that the MASS framework focuses only on the SE-ResNext model and the CSP-ResNext model among 12 models for the CIR data set. This observation suggests that MASS consistently identifies and endorses these two models as the most optimal pre-trained models for the CIR dataset across multiple trials. The preference for these models

Fig. 4 The search process of TPE to identify the optimal model. The illustrative instances are provided for three RNG seeds. TPE systematically constructs models using various architecture combinations throughout 30 iterations. When all iterations are finished, TPE extracts the parameters of the model in the step given by colored dots as \mathcal{S}^*

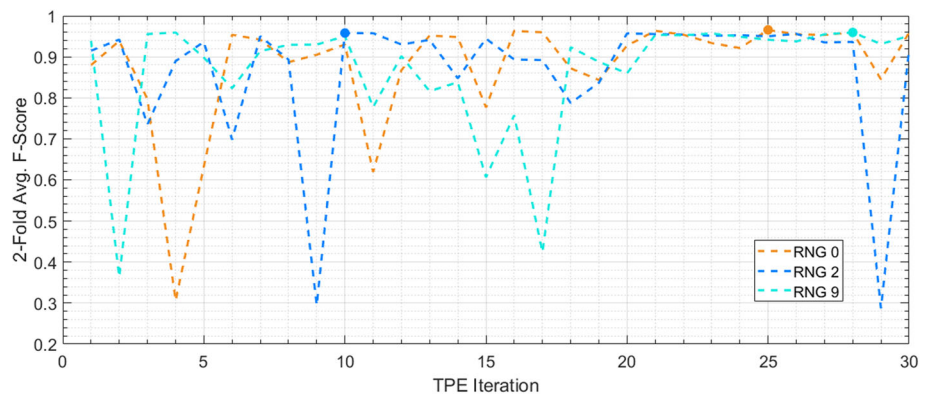


Table 2 Through 10 different random experiments carried out on the CIR dataset, the MASS consistently generates reliable architectural configurations

RNG seed	Selected model	Loss Func	Act. Func	ACC	PRE	RCL	F _{macro}
0	seresnext50 (2)	LS (2)	L.ReLU (2)	0.9866	0.9872	0.9818	0.9840
1	seresnext50 (2)	LS (2)	Mish (3)	0.9833	0.9841	0.9840	0.9838
2	csresnext50 (1)	LS (2)	GELU (5)	0.9866	0.9872	0.9765	0.9803
3	seresnext50 (2)	LS (2)	GELU (5)	0.9766	0.9779	0.9669	0.9706
4	csresnext50 (1)	LS (2)	GELU (5)	0.9849	0.9862	0.9798	0.9824
5	seresnext50 (2)	CEL (1)	Mish (2)	0.9799	0.9808	0.9696	0.9736
6	csresnext50 (1)	LS (2)	SiLU (4)	0.9833	0.9844	0.9723	0.9767
7	seresnext50 (2)	LS (2)	ReLU (1)	0.9799	0.9812	0.9754	0.9776
8	csresnext50 (1)	LS (2)	Mish (3)	0.9799	0.9812	0.9749	0.9775
9	csresnext50 (1)	LS (2)	L.ReLU (2)	0.9816	0.9823	0.9819	0.9818
Overall evaluation	csresnext50-5 seresnext50-5	CEL-1 LS-9	GELU-3 Mish-3	Average scores			
				0.9823	0.9833	0.9763	0.9788

These configurations cover pre-trained model choices, loss functions, and activation functions. The consistent performance exhibited by these configurations on test data attests to their suitability for the CIR dataset. In the "Overall Evaluation" row, the two most frequently selected parameters are reported. The values in parentheses are the flag ID of the relevant parameter

implies their strong compatibility with the CIR dataset. Furthermore, a detailed examination of the chosen loss functions reveals that label smoothing (LS) is consistently favored in 9 out of 10 experimental trials, while Cross Entropy Loss (CEL) is chosen only once. This pattern in loss function selection indicates the robustness and effectiveness of Label Smoothing in the context of the relevant data. Additionally, the investigation of activation functions shows that the most suitable choices for the CIR dataset are GELU and Mish. Each of them is elected by the MASS 3 times, while the others are preferred less frequently. This finding exhibits that the MASS consistently achieves superior performance by strategically aligning with the most appropriate configurations for the CIR dataset.

Table 3 presents the experimental outcomes on the BIL dataset, and a distinct pattern emerges when compared to the findings in the context of the CIR dataset. In contrast to the CIR dataset, MASS exhibits a propensity to select not just two, but a variety of models across different trials for the BIL

dataset. Particularly, within the subset of models chosen, the SE-ResNext and CSP-ResNext models emerge as prominent selections, with the SE-ResNext model displaying a higher prevalence in the BIL dataset. Analyzing the loss function aspect, a significant difference is observed in comparison to the CIR dataset. While LS continues to maintain its prominence, there is a reduction in the disparity between CEL and LS. Specifically, CEL is selected by MASS four times across different trials. However, LS is again the most selected loss function. Regarding activation functions, whereas Mish and GELU are optimally selected functions for the CIR dataset, MASS determines that ReLU and Leaky ReLU are better suited for the BIL dataset. It is worth noting that although the architectural configurations identified by MASS vary when applied to training and test data with different random distributions, the performance on the test data remains relatively stable. This underscores MASS's ability to consistently pinpoint the most fitting model architectures for chromosome datasets.

Table 3 Evaluation of MASS on BIL dataset over 10 experiments

RNG seed	Selected model	Loss func	Act. func	ACC	PRE	RCL	F _{macro}
0	csresnext50 (1)	LS (2)	Mish	0.9598	0.9617	0.9610	0.9610
1	csresnext50 (1)	LS (2)	L.ReLU	0.9507	0.9530	0.9474	0.9496
2	seresnext50 (2)	LS (2)	L.ReLU	0.9580	0.9603	0.9592	0.9594
3	tf_effi_netv2 (8)	CEL (1)	Mish	0.9483	0.9488	0.9496	0.9484
4	seresnext50 (2)	LS (2)	L.ReLU	0.9562	0.9586	0.9552	0.9562
5	seresnext50 (2)	CEL (1)	SiLU	0.9428	0.9455	0.9424	0.9433
6	seresnext50 (2)	LS (2)	ReLU	0.9525	0.9543	0.9536	0.9538
7	inc_resnet_v2 (6)	LS (2)	ReLU	0.9428	0.9454	0.9398	0.9419
8	csresnext50 (1)	CEL (1)	ReLU	0.9544	0.9559	0.9533	0.9542
9	seresnext50 (2)	CEL (1)	SiLU	0.9537	0.9534	0.9551	0.9538
Overall evaluation	csresnext50-3	CEL-4	ReLU-3	Average scores			
	seresnext50-5	LS-6	L.ReLU-3	0.9519	0.9537	0.9517	0.9522

According to CIR, the proposed framework also shows a tendency to choose different models over this dataset. However, SE-ResNext still maintains a superior advantage. In the activation function, the MASS tends to the selection of ReLU and Leaky ReLU

Table 4 Examination of the architectures found by MASS for two datasets across all chromosome classes according to Specificity, Sensitivity, and AUC values

ID	CIR dataset			BIL dataset		
	Specificity	Sensitivity	AUC	Specificity	Sensitivity	AUC
1	0.992±0.02	1.000±0.00	1.000±0.00	0.990±0.01	1.000±0.00	1.000±0.00
2	0.995±0.01	0.999±0.00	1.000±0.00	0.982±0.02	0.999±0.00	0.999±0.00
3	0.979±0.02	1.000±0.00	0.996±0.01	0.962±0.02	0.999±0.00	0.992±0.01
4	0.992±0.01	1.000±0.00	1.000±0.00	0.972±0.02	0.999±0.00	0.994±0.01
5	0.972±0.02	0.999±0.00	0.996±0.01	0.978±0.01	0.999±0.00	0.992±0.01
6	0.987±0.02	0.998±0.00	1.000±0.00	0.935±0.03	0.997±0.00	0.985±0.01
7	0.979±0.02	1.000±0.00	0.999±0.00	0.951±0.02	0.998±0.00	0.988±0.01
8	0.995±0.01	0.999±0.00	1.000±0.00	0.940±0.03	0.997±0.00	0.987±0.01
9	0.992±0.01	0.999±0.00	1.000±0.00	0.954±0.03	0.997±0.00	0.990±0.01
10	0.974±0.03	0.999±0.00	0.996±0.01	0.958±0.02	0.998±0.00	0.990±0.01
11	0.982±0.02	0.999±0.00	0.997±0.01	0.958±0.02	0.998±0.00	0.992±0.01
12	0.990±0.01	0.999±0.00	1.000±0.00	0.964±0.02	0.999±0.00	0.990±0.01
13	0.990±0.01	1.000±0.00	1.000±0.00	0.959±0.02	0.998±0.00	0.991±0.01
14	0.969±0.02	0.999±0.00	0.992±0.01	0.926±0.03	0.996±0.00	0.982±0.00
15	0.982±0.02	0.999±0.00	0.997±0.01	0.931±0.03	0.997±0.00	0.984±0.01
16	0.982±0.02	0.999±0.00	0.997±0.01	0.919±0.04	0.997±0.00	0.984±0.01
17	0.987±0.02	0.999±0.00	1.000±0.00	0.947±0.04	0.997±0.00	0.984±0.01
18	0.987±0.02	0.999±0.00	1.000±0.00	0.957±0.03	0.999±0.00	0.989±0.01
19	0.974±0.03	0.999±0.00	0.999±0.00	0.901±0.04	0.998±0.00	0.978±0.01
20	0.977±0.02	0.999±0.00	0.995±0.01	0.962±0.02	0.998±0.00	0.992±0.01
21	0.977±0.03	0.998±0.00	0.996±0.01	0.971±0.03	0.999±0.00	0.992±0.01
22	0.969±0.03	0.998±0.00	0.992±0.01	0.940±0.03	0.995±0.00	0.989±0.01
23	0.969±0.04	0.999±0.00	0.996±0.01	0.924±0.03	0.997±0.00	0.991±0.01
24	0.933±0.08	1.000±0.00	0.994±0.02	0.964±0.05	1.000±0.00	0.993±0.02

The metrics are provided based on the average of ten random trials. A standard deviation of zero indicates that there is no significant value beyond the second decimal point

While the overall superiority of the models discovered by MASS is evident when assessing their general achievement, Table 4 is provided to scrutinize the performance of

each class individually and identify classes where challenges arise over specificity, sensitivity, and area under curve (AUC) values. The findings represent the average across 10 exter-

Table 5 The Comparison of MASS performance over the CIR dataset by using 80% training and 20% test ratios

Model	Train test ratio	ACC	PRE	RCL	F _{macro}	Source
Lin et al. [16]	80:20	0.9598	0.9600	0.9600	0.9600	Table.1
Wang et al. [10]	85:5:10	0.8611	0.8025	0.8322	N/A	Table.1
Proposed MASS	80:20	0.9823	0.9833	0.9763	0.9788	Table.3

The source column indicates from where the relevant performance values are extracted

Table 6 The Comparison of MASS performance over the BIL dataset by using 70% training and 30% test ratios

Model	Train test ratio	ACC	PRE	RCL	F _{macro}	Source
Wang et al. [10]	85:5:10	0.9258	0.9257	0.9308	N/A	Table.1
Menaka et al. [9]	70:30	0.9310	N/A	N/A	N/A	Table.10
Al-Kharraz et al. [8]	80:20	0.9386	0.9411	0.9451	0.9418	Table.10
Menaka et al. [17]	70:30	0.9460	N/A	N/A	N/A	Table.3
Proposed MASS	70:30	0.9519	0.9537	0.9517	0.9522	Table.4

nally conducted random trials beside Table 3. In the context of sensitivity, the models demonstrate a notable capability in accurately detecting true negative samples. The standard deviation appears as zero due to the precision of values expressed with two digits after the decimal point. Upon examination of specificity values, consistently high results are observed. In particular, classes 14, 22, 23, and 24 are identified as factors contributing to the reduced performance of the model on the CIR dataset. When the BIL dataset is considered, chromosomes 14, 15, 16, 19, and 23 adversely affect the overall average performance.

Tables 5 and 6 present the average results derived from 10 separate trials conducted with the MASS, and they are compared with findings from counterpart studies in the literature over the CIR and BIL datasets, respectively. To ensure methodological consistency, the selection of train-test ratio values adhered to the predominant ratios advocated in prior literature for the respective datasets. Specifically, the values in Tables 5 and 6 present a division of 80% for training and 20% for testing, as well as 70% and 30%, respectively. To facilitate an enhanced understanding of the comparisons, the "Source" column is added at the end of tables to indicate the specific source from the relevant studies where the reported values were extracted. This approach offers a more transparent and fair basis for assessing the performance of the MASS model relative to existing proposals in the field.

When Table 5 is examined, the proposed framework performs better than the other two studies on the CIR dataset. It is evident that higher results are produced not only in the ACC context but also in PRE and RCL values. The results given in Table 6 belong to the BIL dataset. Despite this challenging data set, it is clear that MASS surpasses other studies.

4 Discussion and conclusion

The primary constraint of the presented method lies in the extended trial and error procedures inherent in dealing with large-scale datasets. MASS effectively alleviates researchers from engaging in these trial-and-error tasks by assuming control over the entire process. The computational workload, measured in FLOP (floating-point operations), is directly linked to the user-chosen models and significantly influences their operational time. Another limitation is that MASS refrains from intervening with pre-trained architectures. This is deliberate since the algorithm is specifically crafted to leverage the adaptability of pre-trained models when confronted with new tasks. The results given in the tables indicate the consistent capability of the MASS framework to effectively build model architectures for chromosome data. In the present work, a deliberate choice was made to focus on the selection of 12 pre-trained models, 2 loss functions, and 6 activation functions. It should be noted that these parameter sets can be enlarged in terms of number. Over an expanded parameter set, performance values could be higher by using TPE iteration size N_t that is bigger than 30. Furthermore, the MASS framework offers the flexibility to incorporate various parameters. For instance, the selection of an optimizer (Adam, RMSProp, Nadam, etc.) is a critical factor that affects the performance across datasets, and this choice could also be integrated within the MASS framework.

In summary, chromosome categorization is a labor-intensive technique that is an integral part of chromosomal karyotyping research. It stands as a prominent computer vision challenge that necessitates effective solutions. The fine-tuning strategy of pre-trained models offers an opportunity for considerable performance in the classification of chromosome images. However, identifying the most suitable model from the tens of times of existing options typically involves a manual and time-consuming trial-and-error pro-

cess. In the context of this research, the Model Architecture Search System (MASS) framework is introduced, which leverages the power of the Tree-Parzen Estimator that is based on well-established the Bayesian Optimization mechanism. In addition to the TPE algorithm integrated into the enhancement of the MASS framework, other optimization algorithms such as Asynchronous Successive Halving (ASHA) [28] and BOHB [29] play a crucial role for optimization of machine learning models. These algorithms focus on optimizing resources among workers to address complex and time-consuming problems, particularly in the context of large language models. The utilization of ASHA and BOHB will provide valuable insights and directions for future research endeavors in this domain. The TPE-based MASS framework has been designed to facilitate the automated search for an optimal model architecture built for the specific task of chromosome image classification. Moreover, the capabilities of MASS were extended beyond the mere selection of a pre-trained model to identify the optimal activation and loss functions pertinent to the classification task. This approach, guided by the TPE criterion of expected improvement, strives to construct the most suitable model architecture, and it resolves the need for extensive manual experimentation for chromosome image classification.

Author Contributions NC led the whole process of the paper

Funding Not applicable.

Data availability The used datasets are open source and available in [11] and [16]

Declarations

Conflict of interest Not applicable.

Ethical Approval Not applicable.

References

1. Wang, P.-H., Chen, C.-Y., Yang, M.-J., Chao, H.-T.: Detection of chromosome abnormalities: beyond conventional karyotyping. *LWW* (2015)
2. Bihunyak, T.V., Bondarenko, Y.I., Ulyanda, O.O., Charnosh, S.M., Sverstiuk, A.S., Bihuniak, K.O.: Chromosomal diseases in the human pathology. *Int. J. Med. Med. Res.* **6**, 50–60 (2020)
3. Piper, J., Granum, E.: On fully automatic feature measurement for banded chromosome classification. *Cytom. J. Int. Soc. Analyt. Cytol.* **10**(3), 242–255 (1989)
4. Agam, G., Dinstein, I.: Geometric separation of partially overlapping nonrigid objects applied to automatic chromosome classification. *IEEE Trans. Pattern Anal. Mach. Intell.* **19**(11), 1212–1222 (1997)
5. Hu, X., Yi, W., Jiang, L., Wu, S., Zhang, Y., Du, J., Ma, T., Wang, T., Wu, X.: Classification of metaphase chromosomes using deep convolutional neural network. *J. Comput. Biol.* **26**(5), 473–484 (2019)
6. Popescu, M., Gader, P., Keller, J., Klein, C., Stanley, J., Caldwell, C.: Automatic karyotyping of metaphase cells with overlapping chromosomes. *Comput. Biol. Med.* **29**(1), 61–82 (1999)
7. Johnston, D.A., Tang, K., Zimmerman, S.: Band features as classification measures for g-banded chromosome analysis. *Comput. Biol. Med.* **23**(2), 115–129 (1993)
8. Al-Kharraz, M.S., Elrefaei, L.A., Fadel, M.A.: Automated system for chromosome karyotyping to recognize the most common numerical abnormalities using deep learning. *IEEE Access* **8**, 157727–157747 (2020)
9. Menaka, D., Vaidyanathan, S.G.: Chromenet: a cnn architecture with comparison of optimizers for classification of human chromosome images. *Multidimens. Syst. Signal Process.* 1–22 (2022)
10. Wang, C., Han, M., Wu, Y., Wang, Z., Ma, F., Su, J.: Cnn based chromosome classification architecture for combined dataset. In: 2021 6th International Conference on Communication, Image and Signal Processing (CCISP), pp. 69–74 (2021)
11. Poletti, E., Grisan, E., Ruggeri, A.: Automatic classification of chromosomes in q-band images. In: 2008 30th Annual International Conference of the IEEE Engineering in Medicine and Biology Society, pp. 1911–1914 (2008)
12. Minaee, S., Fotouhi, M., Khalaj, B.H.: A geometric approach to fully automatic chromosome segmentation. In: 2014 IEEE Signal Processing in Medicine and Biology Symposium (SPMB), pp. 1–6. IEEE (2014)
13. Altinsoy, E., Yilmaz, C., Wen, J., Wu, L., Yang, J., Zhu, Y.: Raw g-band chromosome image segmentation using u-net based neural network. In: Artificial Intelligence and Soft Computing: 18th International Conference, ICAISC 2019, Zakopane, Poland, June 16–20, 2019, Proceedings, Part II 18, pp. 117–126. Springer (2019)
14. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. In: Medical Image Computing and Computer-Assisted Intervention—MICCAI 2015: 18th International Conference, Munich, Germany, October 5–9, 2015, Proceedings, Part III 18, pp. 234–241. Springer (2015)
15. Swati, Gupta, G., Yadav, M., Sharma, M., Vig, L.: Siamese networks for chromosome classification. In: 2017 IEEE International Conference on Computer Vision Workshops (ICCVW), pp. 72–81 (2017)
16. Lin, C., Zhao, G., Yang, Z., Yin, A., Wang, X., Guo, L., Chen, H., Ma, Z., Zhao, L., Luo, H., Wang, T., Ding, B., Pang, X., Chen, Q.: Cir-net: automatic classification of human chromosome based on inception-resnet architecture. *IEEE/ACM Trans. Comput. Biol. Bioinf.* **19**, 1285–1293 (2020)
17. Menaka, D., Vaidyanathan, S.G.: A hybrid convolutional neural network-support vector machine architecture for classification of super-resolution enhanced chromosome images. *Expert Syst.* **40** (2022)
18. Ramachandran, P., Zoph, B., Le, Q.V.: Swish: a self-gated activation function. [arXiv: Neural and Evolutionary Computing](https://arxiv.org/abs/1710.05921) (2017)
19. Bergstra, J., Bardenet, R., Bengio, Y., Kégl, B.: Algorithms for hyper-parameter optimization. In: NIPS (2011)
20. Bergstra, J., Yamins, D., Cox, D.D.: Making a science of model search: hyperparameter optimization in hundreds of dimensions for vision architectures. In: International Conference on Machine Learning (2013)
21. Watanabe, S.: Tree-structured parzen estimator: Understanding its algorithm components and their roles for better empirical performance. [arXiv:2304.11127](https://arxiv.org/abs/2304.11127) (2023)
22. Shahriari, B., Swersky, K., Wang, Z., Adams, R.P., Freitas, N.: Taking the human out of the loop: a review of Bayesian optimization. *Proc. IEEE* **104**, 148–175 (2016)

23. Rasmussen, C.E., Williams, C.K.I.: Gaussian processes for machine learning (adaptive computation and machine learning) (2005)
24. Ozaki, Y., Tanigaki, Y., Watanabe, S., Onishi, M.: Multiobjective tree-structured parzen estimator for computationally expensive optimization problems. In: Proceedings of the 2020 Genetic and Evolutionary Computation Conference (2020)
25. Erwianda, M.S.F., Kusumawardani, S.S., Santosa, P.I., Rimadana, M.R.: Improving confusion-state classifier model using xgboost and tree-structured parzen estimator. In: 2019 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), pp. 309–313 (2019)
26. Wightman, R.: PyTorch image models. GitHub (2019). <https://doi.org/10.5281/zenodo.4414861>
27. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: machine learning in python. *J. Mach. Learn. Res.* **12**, 2825–2830 (2011)
28. Li, L., Jamieson, K., Rostamizadeh, A., Gonina, E., Ben-Tzur, J., Hardt, M., Recht, B., Talwalkar, A.: A system for massively parallel hyperparameter tuning. *Proc. Mach. Learn. Syst.* **2**, 230–246 (2020)
29. Falkner, S., Klein, A., Hutter, F.: Bohb: Robust and efficient hyperparameter optimization at scale. In: International Conference on Machine Learning, pp. 1437–1446. PMLR (2018)

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Springer Nature or its licensor (e.g. a society or other partner) holds exclusive rights to this article under a publishing agreement with the author(s) or other rightsholder(s); author self-archiving of the accepted manuscript version of this article is solely governed by the terms of such publishing agreement and applicable law.