

Received July 22, 2020, accepted August 23, 2020, date of publication August 27, 2020, date of current version September 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3019937

Automated System for Chromosome Karyotyping to Recognize the Most Common Numerical Abnormalities Using Deep Learning

MONA SALEM AL-KHARRAZ¹, LAMIAA A. ELREFAEI^{1,2}, (Senior Member, IEEE),
AND MAI AHMED FADEL¹, (Member, IEEE)

¹Computer Science Department, Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah 21589, Saudi Arabia

²Electrical Engineering Department, Faculty of Engineering at Shoubra, Benha University, Cairo 11629, Egypt

Corresponding author: Mona Salem Al-Kharraz (malkharaz0001@stu.kau.edu.sa)

This project was funded by the Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah, under grant no. (DG-21-611-1441). The authors, therefore, gratefully acknowledge the DSR technical and financial support. They would also like to thank Center of Excellence in Genomic Medicine Research (CEGMR) for providing them the dataset and the valuable information.

ABSTRACT Chromosome analysis is an essential task in a cytogenetics lab, where cytogeneticists can diagnose whether there are abnormalities or not. Karyotyping is a standard technique in chromosome analysis that classifies metaphase image to 24 chromosome classes. The main two categories of chromosome abnormalities are structural abnormalities that are changing in the structure of chromosomes and numerical abnormalities which include either monosomy (missing one chromosome) or trisomy (extra copy of the chromosome). Manual karyotyping is complex and requires high domain expertise, as it takes an amount of time. With these motivations, in this research, we used deep learning to automate karyotyping to recognize the common numerical abnormalities on a dataset containing 147 non-overlapped metaphase images collected from the Center of Excellence in Genomic Medicine Research at King Abdulaziz University. The metaphase images went through three stages. The first one is individual chromosomes detection using YOLOv2 Convolutional Neural Network followed by some chromosome post-processing. This step achieved 0.84 mean IoU, 0.9923 AP, and 100% individual chromosomes detection accuracy. The second stage is feature extraction and classification where we fine-tune VGG19 network using two different approaches, one by adding extra fully connected layer(s) and another by replacing fully connected layers with the global average pooling layer. The best accuracy obtained is 95.04%. The final step is detecting abnormality and this step obtained 96.67% abnormality detection accuracy. To further validate the proposed classification method, we examined the Biomedical Imaging Laboratory dataset which is publicly available online and achieved 94.11% accuracy.

INDEX TERMS Convolutional neural network, deep learning, chromosomes classification, data augmentation, transfer learning, object detection.

I. INTRODUCTION

Chromosomes are organized structures that contain the genetic information of the human body. Cytogenetic interests focusing on studying cell activity and chromosome analysis for diagnosing genetic diseases at an early stage [1].

Chromosomes karyotyping become an important clinical process in screening and diagnosing genetic disorders like Edwards syndrome, Turner syndrome, and Down syndrome [1], [2]. It obtained from stained metaphase chromosomes by using staining techniques. It involved two stages:

The associate editor coordinating the review of this manuscript and approving it for publication was Zhiwei Gao.

segmentation and classification of individual chromosomes. Cytogeneticists classified and arranged these chromosomes into 22 pairs for autosomes and 1 pair for sex chromosomes (XY or XX) [3] (Fig. 1). Cytogeneticists label the individual chromosomes to one of the 24 chromosome classes depends on different chromosome features.

Traditional ways for the chromosome classification procedure in most cytogenetic labs are performed manually by experts. This process consumes time and needs efforts from expert operators, therefore expensive.

To make this process easier, many automated karyotyping systems have been developed. These systems mainly follow four steps: image enhancement, segmentation, feature

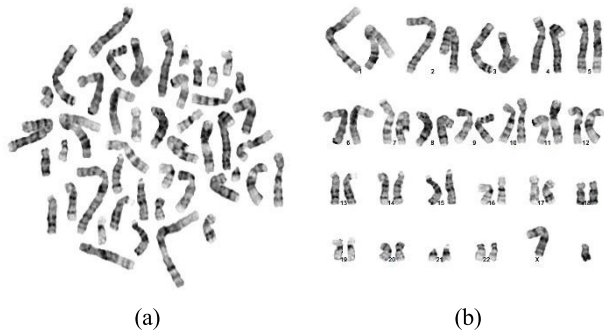


FIGURE 1. (a) Metaphase image, (b) chromosomes karyotype.

extraction, and classification. Chromosomes are classified according to their features and their features can be extracted based on two approaches: handcrafted based features (like chromosomes length, centromere index, and density profile) or learning-based features. Different classification techniques have been used like machine learning and deep learning techniques. Deep learning is considered as the most used technique in the medical image field due to its capabilities of extraction and dealing with complicated features automatically [4], [5].

Recent researches replaced traditional ways of feature extraction based on handcrafted features and classification with a deep learning technique with encouraging results. Convolutional Neural Network (CNN) is one of the public models of deep learning. It is an essential tool for image classification [6] and object detection [7] tasks. It is like the human brain in visually perceiving the world. It consists of neurons that are basic computation units and activated by specific signals. Layers are stacked of neurons and a series of layers are forming CNN [8]–[10]. CNN primarily consists of the following types of layers: convolutional layer is responsible about feature extraction, activation layer is a function that decides if the neuron will fire or not, pooling layer reduces the dimensions (parameters and computations) to avoid overfitting, fully connected layer connects each neuron in the current layer to each neuron in the next layer, and finally, classification layer selects the most probable class [11], [12].

There are two approaches used in deep learning: training a model from scratch and transfer learning. Training a model from scratch needs a huge amount of data to learn a specific task. If the data samples are not large, then the transfer learning can be used in this case especially if there is no available model in this specific domain. It leverages knowledge from previously trained models for training newer models and it also speeds up the training process and improves the overall performance. That means we repurpose a pre-trained model for our own needs by replacing the original classifier with a new classifier that fits our purposes, and finally tuning the model [13]–[15]. ResNet50 and VGG19 are examples of pre-trained models that have been trained on more than a million of images and can classify them into 1000 objects (such as coffee mug, pencil, keyboard, and many animals).

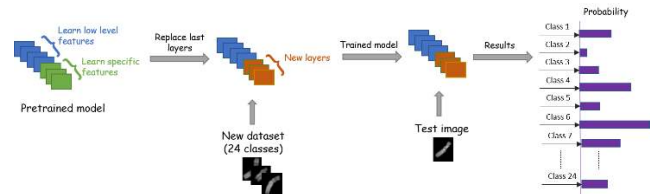


FIGURE 2. Transfer learning concept.

They have been learned huge feature representations from these images [16].

Fig. 2 depicts the concept of transfer learning. It starts by loading the pre-trained model where the convolutional layers (earlier layers) extract features. Then, the last fully connected layer and the final classification layer use to classify the input image. The fully connected layer has the number of classes and it is replaced with a new one that has number of outputs equal to the number of classes in the new dataset. The classification layer specifies the output classes of the network and it is replaced with a new classification layer without class labels. When training the new dataset, automatically the output classes of the classification layer will be sets [17], [18].

We attempt in this article to automate karyotyping steps (segmentation and classification) by detecting and extracting chromosome objects from the metaphase then feeding the chromosome images into CNN for feature extraction and classification. The main contributions are summarized as follows:

1. Provide a categorized review and reporting a comparative summary of the published research on automated karyotyping. the categorization is according to the scope: chromosomes segmentation, chromosome classification, and complete karyotyping systems.
2. Inspired by transfer learning, we fine-tuned YOLO v2 for individual chromosomes detection on a metaphase image with ResNet50 backbone for feature extraction.
3. Fine-tuning the VGG19 by combining the Global Average Pooling (GAP) layer with Fully Connected (FC) layers and by adding Batch Normalization (BN) layer for chromosomes feature extraction and classification.
4. Recognizing the most common numerical abnormalities by determining if there is an extra copy of (13, 18, 21, and X) classes or missing one pair of X chromosomes.

The rest of this article is structured as follows: various segmentation & classification systems are summarized and reported in Section II. Proposed individual chromosomes detection and classification models based on CNN are explained in Section III. Then, experiments and evaluating the performance of the models are discussed in Section IV. Finally, conclusion and future work are presented in Section V.

II. RELATED WORK

The computer-aided systems are required to automate the chromosome analysis and help lab operators to classify

chromosomes and recognize chromosome abnormality. Segmentation and classification of chromosomes are considered the most important and challenging issues in karyotyping and different researches in the literature have been made to automate the karyotyping. Some of these researches focused on chromosome segmentation and they reported in Section II.A, some of them focused on chromosome classification and reviewed in Section II.B, and others focused on whole karyotype system (segmentation and classification) as they summarized in Section II.C.

A. CHROMOSOMES SEGMENTATION

High-quality segmentation is necessary and primarily crucial in developing a computer-aided system because it will affect the accuracy of the classification stage. Various segmentation algorithms are carried out to separate individual chromosomes from the metaphase images.

Minae *et al.* proposed [19] a geometric-based method for partially overlapping and touching chromosomes which proceeded in two phases. The first one detects chromosome clusters that contain touching or partially overlapping chromosomes depending on three geometric criteria that deal with the chromosome's geometry. These criteria are surrounding ellipse, convex hull, and skeleton and end points methods. Next, in the second phase, a cut point is used to separate each chromosome in the clusters and achieved 91.9% accuracy on 62 partially overlapping and touching chromosomes. In the future, they plan to separate a complete overlapping chromosome.

Yilmaz *et al.* [20] started their work by preprocessing and removing unwanted objects then deployed Gaussian-weighted adaptive thresholding to get rid of some dark background parts between chromosomes. Next, they applied binary watershed transform on the binary mask of the image to detect and separate clusters. Geometrical features and skeleton were utilized in examining all connected objects to check it is a single chromosome or a cluster. For separating touched chromosomes, they found appropriate cut points then they calculated a grayscale geodesic distance transform. Whereas for overlapping chromosomes, they created different combinations of possible chromosomes by detecting overlapped regions of clusters and its center then divided the cluster into regions in which each of them has a true cut point. They achieved 97.8% of correctly extracted chromosomes from 145 metaphase images.

The difference of Gaussian (DoG) was used as a sharpening filter in Bashmail *et al.* [21] on 130 non-overlapped metaphase images collected from Diagnostic Genomic Medicine Unit (DGMU) at King Abdulaziz University. Then, they normalized the intensity values to a (0, 1) distribution and finally increased and inversed intensity values for every pixel. For segmentation, they thresholded image using Otsu's method then applied morphological operations like erosion, dilation and hole filling and achieved 99.8% accuracy.

Recently, researches replace traditional segmentation algorithms with deep learning techniques where the network

includes several layers to learn and extract features while training. The number of created features can be practically infinite and without any human bias.

Hu *et al.* [22] applied deep learning and customized U-Net semantic segmentation to distinguish between overlapping chromosomes. They built 13000 grayscale images dataset from raw images available on kaggle and dip4fish blog. They obtained 88-94% intersection over union (IoU) on the non-overlapping chromosome regions and 94.7% for the overlapping region.

The research study done by Saleh *et al.* [23] added an appropriate number of layers to U-Net architecture semantic segmentation which was demonstrated on medical images of cells to help extract more features. Besides, they implemented a Test Time Augmentation (TTA). The dataset is collected from raw images available in Kaggle and Github and it used to build 13,434 greyscale images of overlapping chromosome pairs. The obtained accuracy was 99.68% and IOU for overlapping pixels was 90.63% – 99.94%.

Altinsoy *et al.* [24] used U-net architecture with some modifications: decreasing the feature maps to the half to train the model, changing the input and output image sizes, adding dropout layer after 4th and 5th convolutional blocks, and using adam optimizer instead of SGD. They constructed the dataset that contains 40 metaphase images collected from Renji Hospital. To avoid overfitting, they implemented data augmentation. Before augmentation, the dataset images were divided into 25 for training, 5 for the validation, and 10 for the testing. After augmentation, the number of images increased to 3500 for training and 700 for validation. Elapsed time in segmenting one metaphase image is around 0.25s and they obtained 96.97% Dice similarity coefficient (DSC).

B. CHROMOSOMES CLASSIFICATION

Extensive researches focused on extraction handcrafted features where it based on manually designed features and feature selection methods, like chromosome length and centromere positions features. With the advent of deep learning, other researches applied deep learning in feature extraction.

Moradi and Setarehdan [25] acquired the dataset from Cytogenetic Laboratory of Cancer Institute, Imam Hospital, Tehran, Iran and it contained 303 curved chromosome images of classes 16, 17 and 18 collected and segmented by an expert. They defined new features (width, position) of the two most eye-catching regions of each chromosome and included chromosome length and centromeric index to produce a six-dimensional feature. They feed these features into three-layer artificial neural networks and achieved 98.6% accuracy in classifying these three classes (16, 17, and 18) in group E.

Whereas Mashadi and Seyedin [26] didn't extract features from chromosome images, they instead applied two types of chromosome normalization (intensity and length normalization) and then used image pixels as the input pattern to the Support Vector Machine (SVM) classifier to classify 24 classes. They acquired their dataset from Royan Institute which includes 42000 chromosome images isolated

manually. By experiment, they found using length normalization achieved a better classification accuracy of 95.9%.

As well, Roshtkhari and Setarehdan [27] used the same dataset used in [25] and applied handcrafted features where features have been extracted from the density profile of the chromosome by the discrete wavelet transform. Then for feature reduction, they utilized Linear Discriminant Analysis. They also extracted the centromeric index and the relative length. To classify chromosomes to (16, 17, and 18) classes in group E, they applied a three-layer feed-forward perceptron neural network to classify these classes in group E and obtained 99.3% average correct classification rate.

Chromosome global features (relative length, relative area, and centromeric index) and textural features from GLCM (contrast, inverse difference moment, angular second moment, correlation, variance, and homogeneity) have been extracted by Vanitha and Venmathi [28] from dataset contained 4600 chromosome images segmented manually. The system was divided into two steps. Chromosomes were classified in the first step into seven groups (A-G) using Self Organising Map Neural Network. The correctly classified chromosomes from the first step went to the second step where the seven groups classified into 24 classes using a hybrid neural network approach that combines K-Mean, LVQ, and Naïve Bayes in conjunction with a serial fusion. They achieved 98% classification accuracy and can detect the numerical abnormalities in the metaphase sample, but they didn't mention the exact procedure applied in abnormality detection.

Markou *et al.* [29] constructed the feature vector by following four steps. In the first step, they visited each pixel in the medial axis of the chromosome. Secondly, in each pixel, they calculated the line segment that begins from that particular point and is oriented perpendicularly to the derivative vector at that point. Then, calculating the gray value intensities of the pixels falling under the line segment. Finally, the median of those values has been computed. The dataset employed in this study is from Laboratory of Molecular Biology, General Regional Hospital Papageorgiou, Thessaloniki (Greece) and contains 4554 chromosome images from healthy people and segmented by an expert. They employed a hybrid 2-level classification method in classifying 24 classes. A context-independent SVM classification process took place at the first level followed by a context-aware post-classification stage. They got an overall 6.65% classification errors.

Poletti *et al.* [30] employed Biomedical Imaging Laboratory (BioImLab) dataset containing 5474 chromosome images which is publicly available. They created the feature vector based on the chromosome length, perimeter, area, and 64 samples each for the density and contour profile. These features were fed into ANN for the classification step followed by reassignment to rearrange the 24 classes assigned by the ANN classifier using a greedy approach. The average accuracy of their proposed work was 94% and the average run time was 3s.

Gagula-Palalic and Can [31] proposed a Competitive Neural Network Teams (CNNTs) that ensemble of ANN and nearest neighbor classifiers. This method consists of 462 simple perceptrons. Each perceptron has been trained to distinguish between two classes to form 22 x 21 learning machines. They used Sarajevo dataset containing chromosome features for 3300 chromosomes obtained from the Clinical Center of the University of Sarajevo. The dataset contains chromosome length, length of short p-arm, and ten principal components obtained from band pattern vectors. They were able to classify chromosomes to 22 classes with 1.73% error rate.

The proposed work done by Kusakci *et al.* [32] applied Copenhagen dataset containing features for 4400 chromosomes. The dataset provides chromosome identifier, metaphase index, chromosome type, chromosome length, length of the short arm (p-arm), and the band pattern features. Principal Component Analysis (PCA) was applied to reduce the number of input features. They utilized SVMs where a single SVM was trained to separate a pair of chromosomes. Pattern Search method was used to find optimal parameters for that SVM. The outcome clusters were called Competitive SVM Teams (CSVMTs) and the final output was determined by majority voting. The proposed method achieved correct classification rates of 97.84% in classification 22 chromosomes.

In recent years, deep learning techniques showing very promising results in image classification especially in the medical image field to deal with complicated features. Some of the researches are tending to this field because it facilitates classifying chromosomes to 24 classes.

Chromosomes are segmented manually by an expert and straightening by Swati *et al.* [33] via medial axis extraction and crowdsourcing and via projection vectors methods. They collected the chromosome images from a hospital with 1740 images. They proposed a Siamese Network comprised of twin neural networks and utilized CNN as the base network. The CNN consists of two convolutional layers followed by a maxpooling layer. The result of the final maxpooling is flattened into a vector and fed into a fully connected dense layer followed by the computation of an energy function over the feature representations of the highest level. They applied Multi-layer Perceptron (MLP) as the second training stage and it was trained on the embeddings obtained from Siamese Network. After several experiments, they achieved the highest accuracy (84.6%) when using projection vectors as a straightening method for the chromosomes. They have been classified chromosomes to 24 classes.

Sharma and Vig [8] extracted features from convolutional layers of Residual Neural Networks (ResNet-50). Then, fed into Long Short Term Networks (LSTM - a variant of Recurrent Neural Networks RNN) followed by an attention block. The final layer is the fully connected softmax layer which classifies chromosomes to one of 24 classes. They demonstrated their work on the Biomedical Imaging Laboratory dataset containing 5474 chromosome images which is publicly available. They obtained 90.42% classification accuracy.

Swati *et al.* [34] integrated two networks: convolutional super-resolution and Xception networks. First, they performed length normalization as a preprocessing step. Then, they applied a convolutional super-resolution network to improve the resolution of images by converting them from low to high resolution images. These images were passed to Xception convolutional neural network for the classification task. They validated their system on the Biomedical Imaging Laboratory dataset containing 5474 low resolution chromosome images. They achieved 92.36% classification accuracy. In the future, they plan to detect chromosome structural abnormalities like inversions, translocations, deletions, etc.

The proposed system by Qin *et al.* [4] followed three steps. First, global features like chromosome's length, shape, and size were extracted via the G-Net then local features like texture patterns of local parts were extracted via the L-Net. For the second step, they built two MLP classifiers that utilized the fused features. In the third step, they assigned each chromosome to its type by a dispatch strategy. Their dataset contained 87831 separated chromosomes collected from the Xiangya Hospital of Central South University, China. Two kinds of data augmentation were occurred on the dataset: rotation and flipping. They evaluated their system based on different performance metrics: accuracy, F1-score, the average accuracy of the complete karyotyping per patient case (Acc. per Case), and the average accuracy of the complete karyotyping per patient case using the dispatch strategy (Acc. per Case-D). The conducted work obtained 98.9% accuracy, 98.7% F1, 98.9% Acc. per Case, 99.2% Acc. per Case-D.

C. WHOLE KARYOTYPE SYSTEM

In this section, we summarized the recent studies that include both karyotype stages: segmentation and classification. As well, the studies divided into two types hand-crafted based and deep learning-based feature extraction studies.

Rungruangbaiyok and Phukpattaranont [35] started their work by image preprocessing which includes removing noises and improving contrast and image quality. They segmented 60 metaphase pictures by thresholding and Otsu's algorithm. Dilation and erosion processing algorithms were performed to enhance the images. Measuring the performance of the segmentation process is not mentioned in the paper. After segmentation, they extracted area, band's area, perimeter, band profile, and singular value decomposition features. They adopted the Probabilistic Neural Network (PNN) and divided their system into two steps. The one was classified chromosomes into six groups based on extracted features. The next step classified the six groups into 24 classes. They got 68.18% accuracy result for female and 61.30% for male.

Saranya *et al.* [36] preprocessed image by the median filter then segmented this image by Fuzzy c mean algorithm but neither they mentioned the dataset size nor the result of segmentation. Texture features were extracted based on a GLCM algorithm. They generated a symmetrical normalized GLCM and make features "rotation" invariant by using the isotropic

GLCM. They utilized SVM as a classifier in classifying chromosomes to 24 classes and obtained 95.89% accuracy result.

Also, Neethu *et al.* [37] preprocessed the metaphase picture by thresholding and noise removal. Segmentation was occurred based on a 4-connectivity labelling algorithm and bounding box and they able to segment 1628 chromosome images. Even this article didn't measure the segmentation process performance. They extracted different groups of features like chromosome global features (area, medial axis length, average gray value of each chromosome, and contour length), chromosome centromeric features (contour length p/q ratio and medial axis length ratio of p/q), and textural features from Gray Level Co-occurrence Matrix GLCM (correlation, contrast, homogeneity, entropy, and energy). Two feed-forward Artificial Neural Network (ANN) took place for firstly classifying chromosomes to the Denver group based on global and chromosome centromeric features then, from the Denver group, recognize each chromosome based on its textural features. They obtained 75% accuracy and they claimed the accuracy will improve if they increase the number of samples.

Somasundaram [38] utilized dataset contained 1000 touching chromosomes, 1000 overlapping, and 500 multiple overlapping chromosomes with normal and abnormal cases. They preprocessed the metaphase picture by applying a median filter to reduce the noise and applying morphological operations (dilation, erosion, opening, and closing). Two segmentation methods are carried out, the first one was MOGAC method, but it could not separate overlapping and touching chromosomes, so they employed Hypothesis analysis for isolating these overlapped chromosomes. Furthermore, no segmentation performance was evaluated in this research. PNN and SVM were carried out for classifying chromosomes into 24 classes. For classification by SVM, chromosome length, centromere index, and similarity index are considered input features to the classifier and obtained 97% accuracy. Whereas for PNN, they used chromosome length, centromere position, and perimeter features and obtained 96% accuracy. The research showed that PNN performed better than SVM.

Zhang *et al.* [39] collected the dataset from a local company consisting of 224 karyotyping images and preprocessed these images by utilizing the area filter to remove noises. They isolated each chromosome by using regionprop on the Matlab to generate bounding boxes on these chromosomes and they didn't evaluate segmentation performance. CNN has been constructed for feature extraction and classification. It consisted of five types of layers: convolution, pooling, dropout, flatten, and dense layers. They achieved 92.5% accuracy and 91.3% proportion of well-classified karyotype (PWCK). They trained this network on vertical chromosomes and in the future, they will consider the different orientations of chromosomes.

Wu *et al.* [40] utilized Extremal Regions (ER) with some filters used geometric properties and intensity distribution to segment out individual chromosomes. Whereas

for overlapping and touching chromosomes they segmented them by approximating chromosome shapes with ellipses. In some cases, they failed in segment overlapped and touching chromosomes, so they developed a modification tool to enable clicking on segment chromosomes to form an individual chromosome. Although they still failed in other cases, so they let the technicians segment these chromosomes manually. Their proposed segmentation method obtained 95.9% accuracy and 94.8% recall on 120 metaphase images collected from a private company. Multiple Distribution Generative Advertising Network (MD-GAN) for augmentation has been proposed by them to divers and increase training samples after segmentation. Next, they applied a pre-trained CNN model (VGG16) for classifying metaphase to 24 classes by freezing earlier layers and fine-tuning the higher- layers. They got 63.5% precision.

Segmentation by Somasundaram [41] is done based on cut points and multilevel object-based algorithm on more than 500 normal and 500 abnormal images. For touching chromosomes, they adjusted the level sets to separate each chromosome whereas overlapped chromosomes were separated by the B-spline method. No segmentation results were shown in the paper. For feature extraction and classification, CNN has been constructed and consisted of five types of layers: convolution, pooling, flatten, dense, and dropout layers. Three kinds of data augmentation were occurred on the dataset: rotation, flipping, and scaling. They obtained 98.9% accuracy for normal and abnormal classification.

Sharma *et al.* [42] collected their dataset from a hospital containing 400 healthy patient metaphases where non-expert crowd from CrowdFlower was utilized to isolate chromosomes. They performed some preprocessing steps on the chromosomes before classification including straightening of chromosomes, finding bending orientation, finding the bending center of curved chromosomes, stitching of the two arms of the chromosome, and reconstruction, normalizing chromosome length. They constructed CNN for feature extraction and classification consisted of four blocks where every block contains two convolutional layers, one dropout, and one maxpooling layer. Followed by two fully connected layers and a softmax layer with 24 units. Without preprocessing steps, they obtained 68.5% classification accuracy and by preprocessing steps, they improved the accuracy to 86.7%.

Dataset used by Xie *et al.* [43] for segmentation contains original and synthetic images. They improved the 'cut and paste' image synthesis method to synthesize the annotated images. They followed these steps: collecting 5000 chromosome images from hospital with 20 backgrounds, and 50 kinds of distractors, then generating masks for chromosomes by binarization and morphological transformations. Finally, they combined 48 chromosomes and 2 to 6 distractors with the background. Segmentation was occurred using Mask R-CNN which includes object detection and pixel-level segmentation. They applied geometric algorithms like cropping, rotating, and straightening for highly curved chromosomes. After conducting several experiments, they achieved

95.644% average precision (AP) on IoU threshold 0.5. They proposed a multi-input CNN which takes three inputs at a time: original, cropped, and straightened chromosome images. They applied ResNet-50 as a backbone. The original and straighten images are passed through max-polling layer followed by a convolution layer whereas cropped images fed directly to the ResNet. To combine these features, they created a concatenation layer with 4096 nodes followed by a MLP which includes two FC layers with 1024 nodes and one softmax layer. The network has been trained on data contains 480000, 90000, and 90000 training, validation, and testing respectively and achieved 95.70% accuracy.

Further contributions are required to automate abnormality detection. In this article, we propose to automatically detect the individual chromosomes on the metaphase image and classify the chromosomes based on CNN deep learning. It eliminates manually selection and extraction features and therefore guaranteeing the accuracy of the selected chromosome features to ensure the accurate detection of abnormality.

III. THE PROPOSED SYSTEM

The proposed system mainly consists of three major stages. The first stage is individual chromosomes detection via object detection using You Only Look Once (YOLO) v2 followed by chromosomes post-processing. The input to this stage is the non-overlapped metaphase and the output is the detected and separated chromosomes. The second stage is chromosome classification via VGG19 whereas the input is the detected and separated chromosomes coming from previous stage and the output is the classified chromosomes. The final stage is abnormality detection based on the result of the classification stage and the input to this stage is the classified chromosomes for single metaphase and the output is the diagnosis. The details of the stages are shown in the sections below. The overall proposed system is shown in Fig. 3.

A. FIRST STAGE: INDIVIDUAL CHROMOSOMES DETECTION

In our study we used non-overlapped metaphase images therefore, we utilized deep learning object detection to locate the presence of chromosomes and surround each one by a bounding box.

YOLO is a kind of object detection and is addressed as a regression problem to spatially separate bounding boxes and associated class probabilities. It is much faster than R-CNN therefore, it designed for speed and real-time use. R-CNN approaches like Faster RCNN predict detections based on a specific region, whereas YOLO uses features from the whole image in predicting boundaries. It involved classification and localization tasks and it can be trained on different pre-trained CNN. The input image is divided into a grid of cells each of them is responsible for predicting class probabilities, bounding box locations, and box confidence scores (objectness). Pre-defined bounding boxes (anchor boxes) with suitable shapes and sizes are determined during training [44], [45].

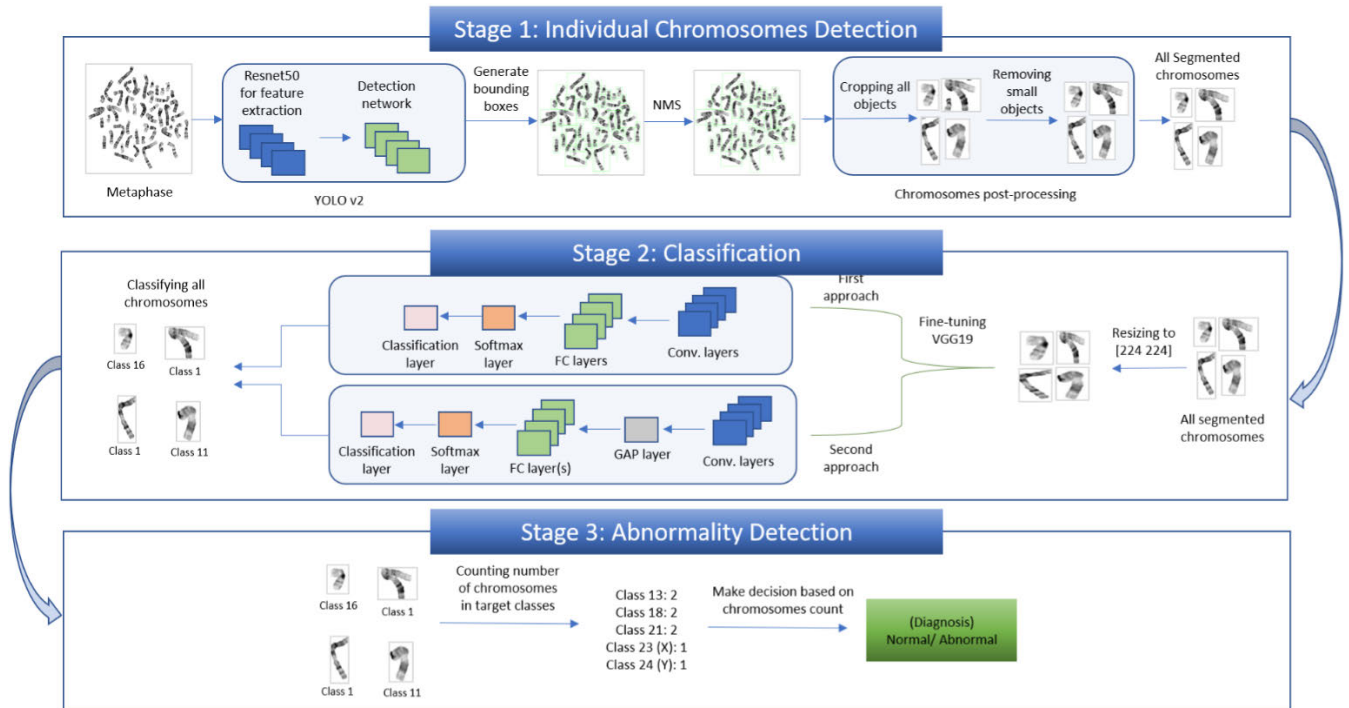


FIGURE 3. Framework of our proposed system.

1) YOLO v2 ARCHITECTURE

YOLO v2 composes of two subnetworks: network for feature extraction and a detection network. The network for feature extraction is a pre-trained CNN model while the detection network consists of a few convolutional layers and specific layers for YOLO v2 [45].

Image input size, the number of anchor boxes, and the base of the pre-trained CNN model for feature extraction have been specified to create YOLO v2 network.

Choosing the number of anchor boxes is considered as training parameters that should be selected carefully. We used mean IoU distance metric with k-means clustering algorithm to identify the top-k boundary boxes that have the best fit of the training data. If the mean IoU is greater than 0.5, this means the anchor boxes overlap well with boxes in the training data [45], [46].

We used ResNet50 as the feature extractor. It includes 50 residual layers and introduces an “identity shortcut connection” that skips blocks of convolutional layers to form blocks named residual blocks. These residual blocks solve the degradation of training accuracy presented in deep networks [47], [48]. Fig. 4 illustrated the architecture of ResNet50.

We extracted features from ‘activation_40_relu’ layer and removed all layers after ‘activation_40_relu’ then added the detection subnetwork. The detection subnetwork consists of groups of connected convolutions, batch normalization, and ReLU layers. The final part of the detection subnetwork includes convolution, YOLO v2 transform, and YOLO v2 output layers as seen in Table 1. The main goal of the

convolution layer is to predict object class probabilities, x and y location offset, width, and height offset for each anchor box. The feature map of this layer is $(57, 57, 54)$ where 57×57 is the grid of cells. Each prediction includes 4 parameters for the boundary box, 1 box confidence score, and 1 class probabilities (we just have one class which is a chromosome). After the experiment, we found the best boundary boxes are 9 with $(4+1+1)$ parameters so the total number of parameters per grid cell are 54 parameters. Transform Layer extracts activations of the last convolutional layer and transforms the bounding box predictions to be within the bounds of the ground truth. It improves the stability of the network by constraining the location predictions [49]. Whereas the output layer provides the refined bounding box locations of the target chromosome objects [50].

The first row in Fig. 5 presents samples of original metaphases and the second row presents YOLOv2 results where each chromosome is surrounding by a bounding box.

2) CHROMOSOME POST-PROCESSING

Every chromosome is cropped from the original image according to its bounding box. After cropping, some chromosome images contain another chromosome especially the short chromosomes as seen in Fig. 6. Fig. 6 (a) is showing the original metaphase image, Fig. 6 (b) is showing the detected boxes where every chromosome is surrounded by a bounding box and according to bounding boxes, all chromosomes are cropped. Fig. 6 (c) is showing one of the chromosomes after cropping where it contains another short chromosome object.

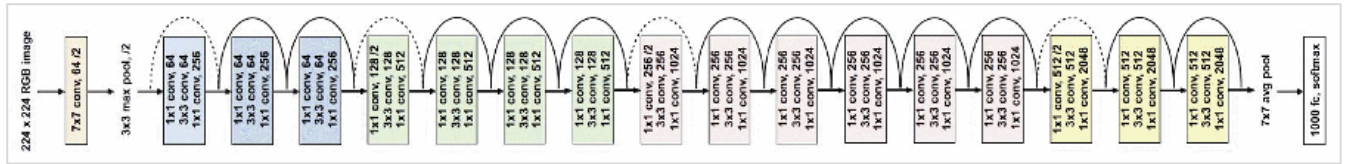


FIGURE 4. ResNet50 architecture [47].

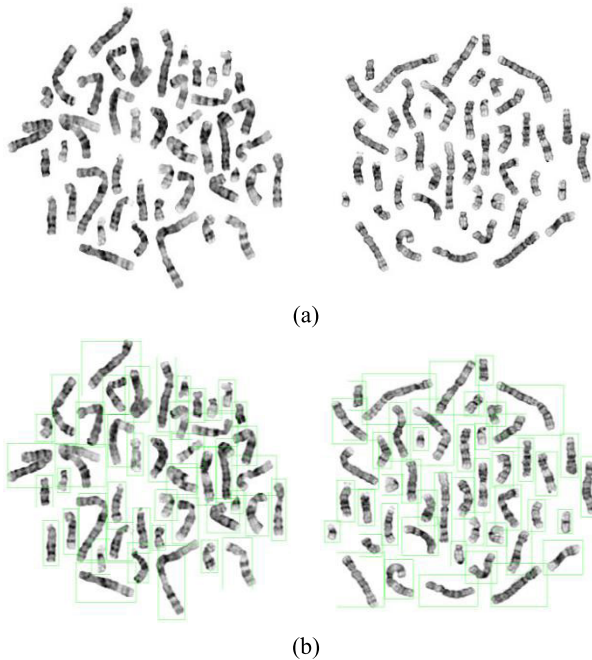


FIGURE 5. Samples of YOLOv2 results: (a) original metaphases (b) detected chromosomes on metaphases.

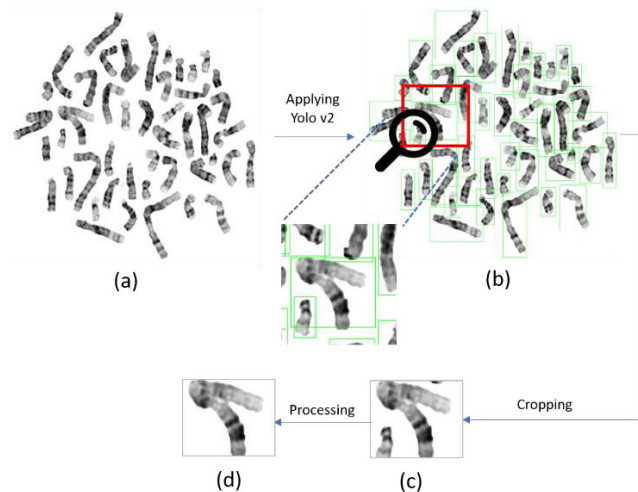


FIGURE 6. Chromosomes post-processing after object detection.

The problem here is not from the detected bounding box, the bounding box fit correctly on the chromosome object, the problem is from the short chromosome where it can

be in the bounding box of the tall and bended chromosome. Although the short chromosome is detected correctly. According to this situation, we did some post-processing on the chromosome images that contain this case as seen in Fig. 6 (d).

TABLE 1. Detection subnetwork architecture of YOLO v2.

Layer Name	Type	# of Filters	Feature Map	Stride
yolov2Conv1	Convolution	1024	(57, 57, 1024)	[1 1]
yolov2Batch1	Batch Normalization			
yolov2Relu1	ReLU			
yolov2Conv1	Convolution	1024	(57, 57, 1024)	[1 1]
yolov2Batch1	Batch Normalization			
yolov2Relu1	ReLU			
yolov2ClassConv	Convolution	54	(57, 57, 54)	[1 1]
yolov2Transform	YOLO v2 Transform Layer		(57, 57, 54)	
yolov2OutputLayer	YOLO v2 Output			

The post-processing step starts by creating a binary image, then counting the number of objects in the image by labelling connected components [51]. If the number of objects is more than one, then the process continues to the next step which determines the largest object in the image. Specifying the largest object is done by counting the number of pixels belong each object using connected components analysis that search for the next unlabeled pixel p, then use a flood-fill algorithm to label all the pixels in the connected component containing p. The analysis repeats this process until all the pixels are labeled. Finally, filling the image regions and holes based on morphological reconstruction [52] and making outside the image the mean of what's above the threshold. Since we need the white area around the chromosome, without affecting the light bands of the chromosome, but this area is not a clear white (255) so after different experiments, we found the best threshold is between 235-245 that will not affect the light bands. In our case, we choose the threshold to be 240 which is the mid-point between 235-245.

B. SECOND STAGE: CHROMOSOME CLASSIFICATION

The input to this stage is the separated chromosomes. In this research, the feature extraction and classification are based on fine-tuning VGG19. VGG19 includes 19 layers divided

into 16 convolutional layers with a very small receptive field 3×3 convolution filters and 3 fully connected (FC) layers. The width of the convolutional layers starts from 64 in the first layer and increases by a factor of two after each max-pooling layer, until it reaches 512. The first two layers of FC have 4096 neurons and the final FC layer has 1000 neurons (one neuron for each class) followed by the softmax layer. All hidden layers are equipped with the Rectified Linear Unit (ReLU). It accepts RGB images in size 224-by-224 pixels [16], [47]. The architecture of VGG19 is illustrated in Fig. 7 [47].

One important thing in a deep learning network is the generalization ability that reduces the overfitting. Different regularization techniques were proposed, and they proved their efficiency in avoiding overfitting and make the model more stable.

- **Batch Normalization (BN):** speed up the convergence, increase the stability, and regularize the model. It can be added after a convolutional layer and it performs some operations on the output of the preceding activation layer. These operations include standardization, normalization, scaling, and shifting operations [53], [54].
- **Dropout:** during training, neurons' weights are tuned for specific features providing some specialization then the neighbors' neurons rely on this specialization which will result in a model that fits on the training data. Dropout randomly ignored (dropped) selected neurons along with their connections from the network. This prevents neurons from complex co-adaptations on training data and forces them to learn features on their own [54], [55].
- **Global Average Pooling (GAP):** the proposed work on Network in Network (NIN) [56] is following where GAP partially or fully replaces the traditional fully connected layers and it reduces the dimension of each tensor by taking the average output of each feature map in the previous layer then, feeding the result vector directly to a softmax layer. If a tensor has a dimension ($h \times w \times d$), GAP reduces the dimension to ($1 \times 1 \times d$) by calculating the average of each feature map ($h \times w$). GAP idea is generating one feature map for each corresponding category of the classification task. The advantage of GAP is avoiding overfitting because there is no parameter needs to optimize.

We use the above techniques in modifying the feature extraction & classification parts of the base model. Also, freezing the weights of some earlier layers by setting the learning rates to zero to speed up the training because the gradients of these layers do not need to be computed and because our dataset is small and different from the pre-trained model's dataset [14], [17]. We also, modify the last fully connected and classification layers of the model according to our number of classes (24 classes). The modifications are done with varying model depth and varying feature size in two approaches [57]:

Approach 1: Adding FC layer(s) with different number of neurons after 'fc8' layer. Fig. 8 demonstrates the concept of this approach.

Approach 2: Removing the top classification layers then, combining the GAP layer with FC layer(s) followed by the softmax layer. More precisely we remove layers from 'fc6' to 'fc8', then we add the new layers. Fig. 9 shown the concept of this approach of fine-tuning the VGG19.

We examined different schemes with the above approaches to modify the pre-trained VGG19 model with varying model depth (hidden layers) and varying feature size (number of neurons) and these schemes are listed below:

1. No additional FC layers are added. In approach 1, we kept the base model as it is whereas in approach 2, we connected the GAP layer to the softmax layer.
2. We examined varying feature size by adding FC layer with a different number of neurons: 2048, 4096, 5000, and 6000.
3. Moreover, we examined varying model depth by adding regularization technique layers that reduce over-fitting to investigate their efficiency and effectiveness on the network. We combined dropout layer and FC layer with ReLU activation layer in this order: FC + ReLU + dropout + FC + ReLU + dropout. As well we examined the effect of BN layer in two ways. The first one is examining it in classification part by combining it with FC layer followed by ReLU in this order: FC + BN + ReLU + FC + BN + ReLU. The second way is to examine it in the feature extraction part by adding BN after each convolutional layer without adding any extra layer in the classification part.

Table 2 presents the examined approaches with different schemes and Fig. 10 visualizes these combinations of approaches and schemes. The output of this stage is the classified chromosomes.

C. THIRD STAGE: ABNORMALITY DETECTION

The next step after classification is detecting and diagnosing the abnormality. The input to this stage is the classified chromosomes for single metaphase. Collected testing metaphases contain different normal and numerical abnormal cases as shown in Table 3.

The decision is made according to the chromosomes count and is based on a traditional conditional statement. It starts by counting the total number of the detected chromosomes for the single metaphase and the number of chromosomes on each target class (13, 18, 21, X, Y). If the total number of chromosomes is 47 and the number of chromosomes in class 13 is three, then this case is diagnosing as Trisomy 13 (Patau Syndrome). If it is not, then it will check the number of chromosomes in class 18. If it has three copies of chromosomes 18 then it considered as Trisomy 18 (Edwards Syndrome). The same thing is done for Trisomy 21. Whereas for Trisomy XXY (Klinefelter Syndrome), it detected if the above conditions are not met and the total number of chromosomes

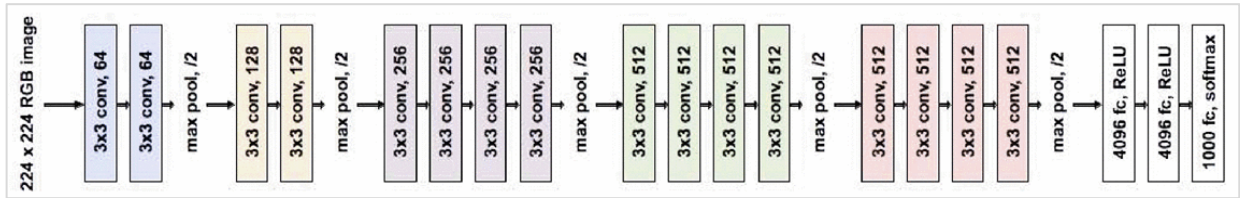


FIGURE 7. VGG19 architecture [47].

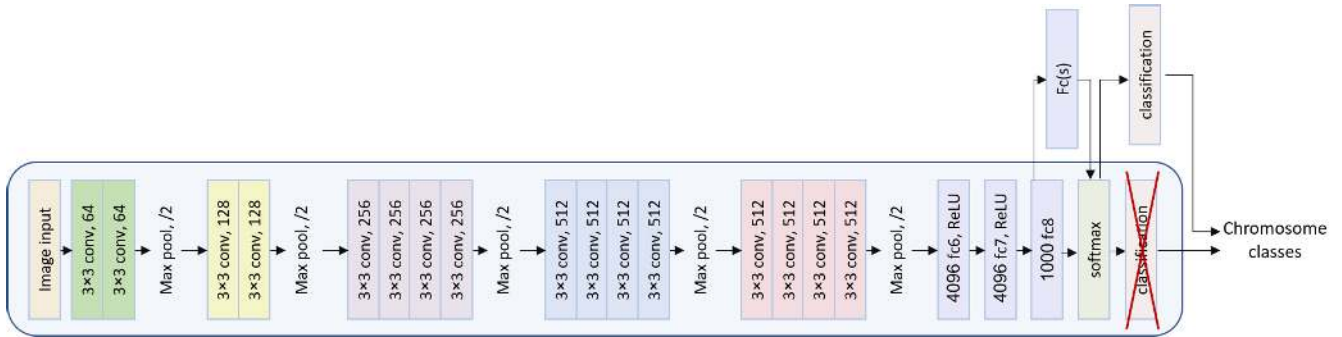


FIGURE 8. Approach 1 of fine-tuning VGG19.

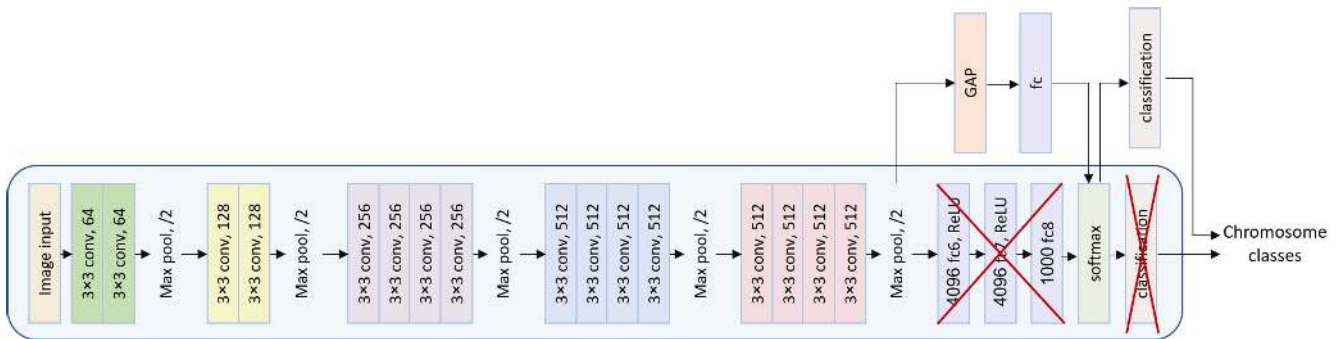


FIGURE 9. Approach 2 of fine-tuning VGG19.

is 47, and the number of chromosomes in classes 23 (X class) and 24 (Y class) is two and one respectively. If the total number of chromosomes is 45 and there is only one chromosome in class 23 and nothing in class 24, then this case is Monosomy X. Otherwise the case is predicted as a normal case. Algorithm 1 summarizes this stage procedure.

IV. EXPERIMENTS AND RESULTS DISCUSSION

A. TOOLS

We implemented the proposed system in MATLAB version R2019b with Deep Learning Toolbox on a computer with Intel(R) Core (MT) i7-6700K CPU @ 4.00GHz processor, NVIDIA GTX 1070 graphics card, 2T HDD, and 16GB RAM.

B. DATASET

To evaluate our system and approaches, we perform experiments on two different datasets:

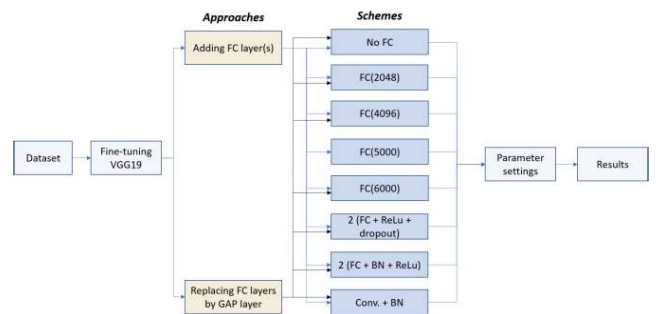


FIGURE 10. Framework of fine-tuning VGG19 with a combination of proposed approaches and schemes.

Cytogenetic Service Unit dataset: from Center of Excellence in Genomic Medicine Research (CEGMR) at King Abdulaziz University (KAU) [58]. We collected 147 jpg file format metaphase images contain normal and abnormal

TABLE 2. Examined proposed approaches with different schemes.

Scheme No.	Scheme Name	Description
First approach (Adding FC layer(s) with different number of neurons)		
1	No FC	No extra FC layer(s) added. This scheme is the VGG19 base
2	FC(2048)	Extra FC layer with 2048 neurons comes after 'fc8' layer
3	FC(4096)	Extra FC layer with 4096 neurons comes after 'fc8' layer
4	FC(5000)	Extra FC layer with 5000 neurons comes after 'fc8' layer
5	FC(6000)	Extra FC layer with 6000 neurons comes after 'fc8' layer
6	2 (FC + ReLu + dropout)	FC(4096) + ReLu + dropout + FC(4096) + ReLu + dropout layers come after 'fc8' layer
7	2 (FC + BN + ReLu)	FC(4096) + BN + ReLu + FC(4096) + BN + ReLu layers come after 'fc8' layer
8	Conv. + BN	BN after each convolutional layer in feature extraction part
Second approach (combining the GAP layer with FC layer(s))		
9	No FC	GAP layer with no FC layer(s)
10	FC(2048)	FC layer with 2048 neurons comes after GAP layer
11	FC(4096)	FC layer with 4096 neurons comes after GAP layer
12	FC(5000)	FC layer with 5000 neurons comes after GAP layer
13	FC(6000)	FC layer with 6000neurons comes after GAP layer
14	2 (FC + ReLu + dropout)	FC(4096) + ReLu + dropout + FC(4096) + ReLu + dropout layers come after GAP layer
15	2 (FC + BN + ReLu)	FC(4096) + BN + ReLu + FC(4096) + BN + ReLu layers come after GAP layer
16	Conv. + BN	BN after each convolutional layer in feature extraction part with GAP layer in the classification part

TABLE 3. Distribution of testing cases.

Case type	No. of cases
Normal	5
Trisomy 13 (Patau Syndrome)	5
Trisomy 18 (Edwards Syndrome)	5
Trisomy 21 (Downs Syndrome)	5
Trisomy XXY (Klinefelter Syndrome)	5
Monosomy X (Turner Syndrome)	4

numerical cases with 910×910 pixels quality resolution. The selected images are free overlap, sever some bending, and have different band resolutions between 550 to 650. The difference in the band resolutions depends on the sample type and quality. This dataset used for individual chromosomes detection, classification, and abnormality detection stages. The individual chromosomes detection process produces separated 6807 chromosomes with different sizes like 31×62 , 51×67 , 100×200 , 69×82 , and 43×62 . Fig. 11 shows sample of metaphase images. The distribution of chromosomes over classes is shown in Table 4.

Biomedical Imaging Laboratory (BioImLab): which is publicly available online [59]. The dataset contains 5474 grayscale separated chromosome images and the distribution of these chromosomes over classes is tabulated in Table 5. The images are in bmp file format with also different sizes. We use this dataset to validate our proposed

Algorithm 1 Decision Making and Diagnosis

//Input: Detected and classified chromosome images for single metaphase

//Output: Case diagnosis

1. TotalChro: count total no. of individual chromosomes for the single metaphase
2. ch13, ch18, ch21, ch23, ch24: count no. of chromosomes in class 13, 18, 21, 23, and 24 respectively
3. **if** TotalChro==47 & ch13>=3
4. 'Trisomy 13'
5. **else if** TotalChro==47 & ch18>=3
6. 'Trisomy 18'
7. **else if** TotalChro==47 & ch21>=3
8. 'Trisomy 21'
9. **else if** TotalChro==47 & ch23>=2 & ch24>=1
10. 'Trisomy XXY'
11. **else if** TotalChro==45 & ch23 == 1 & ch24 == 0
12. 'Monosomy X'
13. **else**
14. 'Normal'
15. **End**

TABLE 4. Chromosome distribution over classes for CEGMR dataset.

Chromosome Class	No. of Chromosomes
1-12, 14-17, 19-20, 22	294 for each class
13	306
18	308
21	312
23 (X)	231
24 (Y)	64

classification stage. Fig. 12 shows sample of images from different classes.

TABLE 5. Chromosome distribution over classes for BioImLab dataset.

Chromosome Class	No. of Chromosomes
1-22	238 for each class
23 (X)	194
24 (Y)	44

C. FIRST STAGE: INDIVIDUAL CHROMOSOMES DETECTION EXPERIMENTS & RESULTS

1) DATASET PREPARATION

The CEGMR dataset is used to train the YOLO v2. It contains 147 metaphase images and the ground truth for these images are generated using image labeler app provided by MATLAB. This app enables to mark a rectangular region of interest (ROI) labels on each metaphase image. Out of these images we utilized 80% as training (118 metaphase images) and 20% as testing (29 metaphase images).

2) FINE-TUNING YOLO v2

YOLO v2 object detection is used to localize every chromosome on metaphase image and after conducting different experiments, we found the best training parameters are



FIGURE 11. Sample of CEGMR metaphase images.

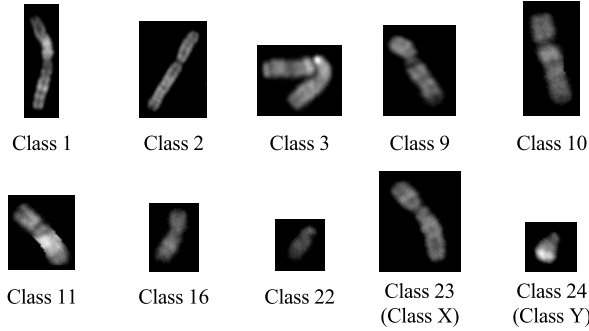


FIGURE 12. Sample images for different classes in BiolMLab dataset.

stochastic gradient descent (SGD) optimizer, 10^{-3} learning rate, 0.9 momentum. We trained the model for 60 epochs on 4 mini-batch size. We found the best number of anchors is 9 which produces IoU=0.8 on the training samples as seen in Fig. 13.

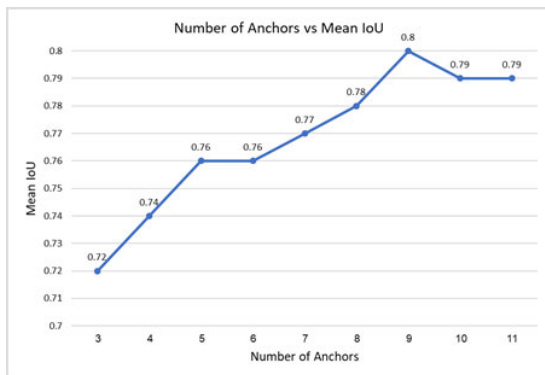


FIGURE 13. Number of anchors vs mean IoU.

To improve the accuracy and boost the performance of the deep learning network, the dataset should have a large amount of training data but unfortunately, most of the applications domain do not have access to a large amount of data especially the medical domain.

Data augmentation is a regularization technique used to increase the diversity of training data, without having to increase the number of labeled training samples. It used to help preventing the network from overfitting caused by using small datasets and preventing the network from memorizing the exact details of the training samples therefore, it improves network accuracy. In our study, we used online data augmentation (also, called augmentation on the fly) where no

need to save the results on disk as in offline augmentation. Fig. 14 depicts online data augmentation for the individual chromosomes detection stage. At each epoch during training, the augmented image dataset has a random combination of transformations for images in the mini-batch of the training sample. So, each epoch uses a slightly different dataset and at the same time in each epoch the actual number of training images does not change [60], [61].

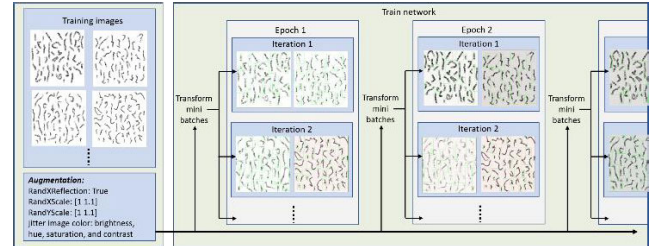


FIGURE 14. Online augmentation for individual chromosomes detection.

The augmentation occurred by randomly flipping the original data image and associated box labels horizontally, randomly scaling by a scale factor that is between [1, 1.1], and randomly jittering images color for brightness, hue, saturation, and contrast during training.

3) FIRST STAGE RESULTS & DISCUSSION

The most widely used and common metric for object detection is the IoU (Jaccard’s index). It computes the similarity between the ground truth box and the predicted box. It is defined as the intersection between boxes divided by the union of these boxes and a returned value is between 0 and 1 where 1 means the perfect fit between boxes. The metric is defined in (1).

$$Intersection\ over\ union\ (IoU) = \frac{area\ of\ intersection}{area\ of\ union} \quad (1)$$

AP (Average precision) is another common metric in measuring the performance of object detection. To compute this metric, we first compute recall and precision values. Recall defined in (2) is a ratio of correctly detected positive objects to the total number of objects in the dataset.

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

where TP is the True Positive and FN is the False Negative. Equation 3 represents the precision that is the ratio of correctly detected positive objects to the total detected positive objects.

$$Precision = \frac{TP}{TP + FP} \quad (3)$$

where FP is False Positive. After computing recall and precision, AP is calculating by taking the average value of the precision at the set of 11 equally spaced recall values as in (4).

$$AP = \frac{1}{11} \sum_{Recall_i} Precision(Recall_i) \quad (4)$$

Accuracy in (5) is the ratio between the number of correctly separated chromosomes to the total number of chromosomes. Also, we compute the individual chromosomes detection time for each metaphase.

$$Accuracy = \frac{\text{correctly segmented chromosomes}}{\text{total number of chromosomes}} \times 100 \quad (5)$$

For some metaphases, YOLO v2 may be produced more than one box for the same detected chromosome object as seen in Fig. 15. To solve this, Non-maximum Suppression (NMS) comes up. It keeps the box that has the highest confidence score and removes the other [44], [46].

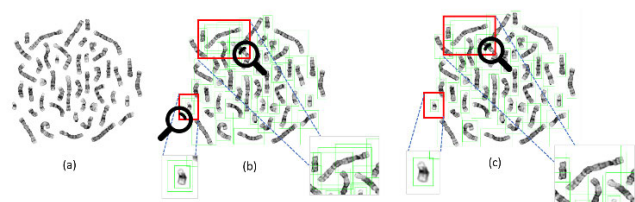


FIGURE 15. Applying Non-maximum Suppression on predicted metaphase bounding boxes. (a) shows the original metaphase, (b) shows the predicted bounding boxes, (c) shows predicted bounding boxes after applying NMS.

The experimental results on CEGMR testing dataset get the mean IoU=0.84. To compute AP, we define a prediction of chromosome object to be a TP if the IoU for that object ≥ 0.5 , and a FP if the IoU < 0.5 . If the model missed detecting the object, then it considered as FN. After calculating recall and precision, precision/recall (PR) curve is plotted as in Fig. 16 to calculate AP and as we see our model achieved AP=0.9923. We measure the total individual chromosomes detection time for each metaphase and we find the individual chromosomes detection time is between 1.2990 s - 1.6138 s.

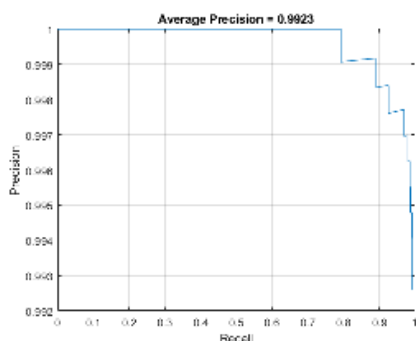


FIGURE 16. PR curve.

Fig. 17 visualizes the original metaphase images, ground truth, predicted bounding boxes, and the overlapping between the ground truth and the predicted. The predicted boxes acceptably fit the chromosome objects. Some parts of some chromosomes go outside the boxes especially the tall chromosomes but the number of chromosomes that fall in this case is rare. Fig. 18 presents one of this case where the left image is the ground truth and the right image is the

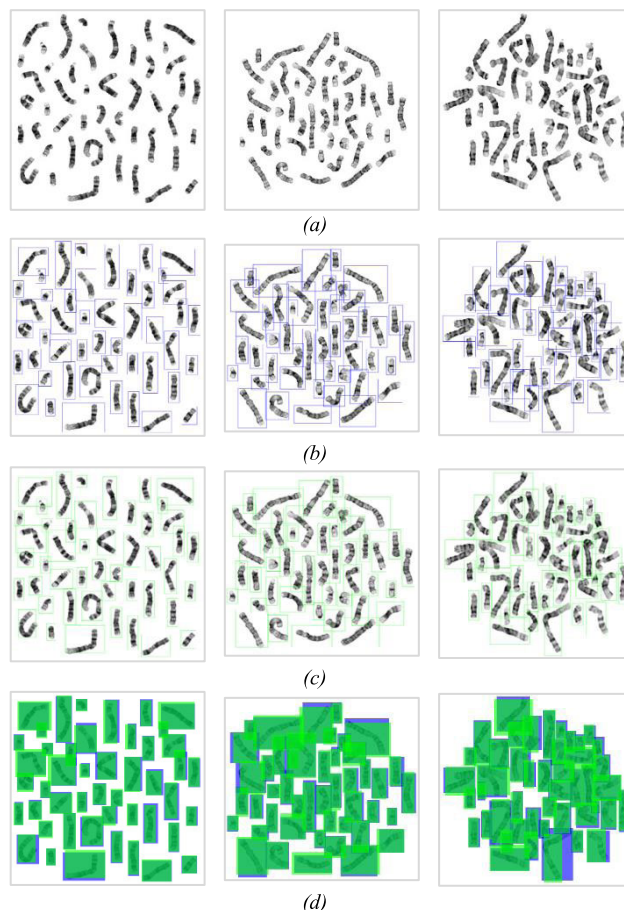


FIGURE 17. Samples of detected boxes where blue color represents the ground truth and the green color represents the predicted: (a) original metaphase images (b) ground truth images (c) predicted (d) overlapped regions.

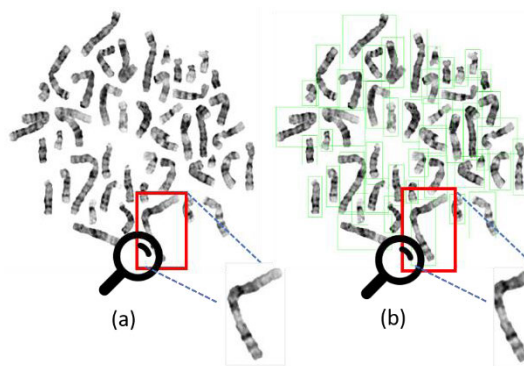


FIGURE 18. Segmentation result for chromosome which its predicted box is not fit the whole chromosome.

separated. However, the classifier classifies this chromosome correctly. Although, the proposed individual chromosomes detection stage successfully separated all 1350 chromosomes on 29 metaphases and as a result, the individual chromosomes detection achieves 100% accuracy. All 1350 separated chromosomes from this stage are used as it is in the second stage even if it has some missing parts as seen in Fig. 18 (b).

4) COMPARISON WITH THE STATE-OF-THE-ART SEGMENTATION METHOD

Since [21] is the only research used the same dataset collected from CEGMR [58] except it used 130 metaphases and we increased the metaphases to be 147. Table 6 summarizes the differences between the two studies that used CEGMR dataset and as seen our individual chromosomes detection technique obtained better result, whereas Table 7 compares our proposed method with the state-of-the-art methods that used deep learning for chromosome segmentation. The comparison is relative because the used techniques and datasets are different. We notice that three papers [22], [32], and [33] used U-Net semantic segmentation and paper [43] used another type of segmentation which is instance segmentation (Mask R-CNN). Each pixel in segmentation is labeled with the its class while in our research we applied object detection technique (YOLO v2) where each chromosome is distinguished by a bounding box. As we see our proposed method achieved comparable result with the state of the art.

TABLE 6. Comparing the proposed individual chromosomes detection stage with a state-of-the-art method on CEGMR dataset.

Research paper	Technique	Dataset	Performance
Bashmail <i>et al.</i> (2019) [21]	Thresholding and Otsu's algorithm	130 metaphase images	Accuracy: 99.8%
Our proposed method	YOLO v2 + chromosomes post-processing	147 metaphase images (118 for training and 29 for testing)	Accuracy: 100% IoU: 0.84 AP: 0.9923 individual chromosomes detection time per metaphase: 1.2990 s - 1.6138 s

D. SECOND STAGE: CHROMOSOME CLASSIFICATION EXPERIMENTS & RESULTS

1) DATASETS

Two different datasets used in this stage. The first one is coming from the previous stage and contains 6807 separated chromosomes. Out of these, we used 5457 chromosomes for training VGG19 and 1350 for testing. The testing images are belonging to the 29 metaphases used in the testing previous stage. The second dataset is BioImLab and contains 5474 chromosome images. We converted these images to RGB format and utilized 4370 for training and 1104 for testing. Besides the two datasets, we combined them to increase the number of chromosomes in each class. To ensure the balance splitting among the datasets, we first split each dataset separately to training and testing then merge the training from each dataset to make a new training dataset contains 9827 chromosomes and a new testing dataset contains 2454 chromosomes. Table 8 summarizes the size of training and testing for each dataset. Different experiments are performed on the three datasets to get the highest performance model.

TABLE 7. Relative comparison with state-of-the-art methods that addressed chromosomes separation using deep learning (segmentation/object detection).

Research paper	Technique	Dataset	Performance
Hu <i>et al.</i> (2017) [22]	U-Net (semantic segmentation)	Built 13000 images from raw images available on kaggle and dip4fish blog	IOU on the non-overlapping chromosome: 88-94% IOU for the overlapping region: 94.7%
Saleh <i>et al.</i> (2019) [32]	U-Net (semantic segmentation)	built 13,434 images of overlapping chromosome pairs from raw images available in Kaggle and Github	Accuracy: 99.68% IOU for overlapping pixels: 90.63% – 99.94%
Altinsoy <i>et al.</i> (2019) [33]	U-Net (semantic segmentation)	40 metaphase images from Renji Hospital	DSC: 96.97%
Ning <i>et al.</i> (2019) [43]	Mask R-CNN (instance segmentation)	Built dataset from 5000 chromosome images + 20 backgrounds + 50 kinds of distractors collected from hospital	AP: 95.644%
Our proposed method	YOLO v2 (object detection) + chromosomes post-processing	147 metaphase images	Accuracy: 100% IoU: 0.84 AP: 0.9923 individual chromosomes detection time per metaphase: 1.2990 s - 1.6138 s

All images are resized to 224-by-224 to match the required input size for the first layer of the VGG19.

TABLE 8. Size of training and testing datasets used in the classification stage.

Dataset	Training Size	Testing Size
CEGMR	5457	1350
BioImLab	4370	1104
Combined (CEGMR and BioImLab)	9827	2454

2) FINE-TUNING VGG19

For the three datasets, we train the network using the combination of the proposed approaches and schemes listed in Table 2 with a learning rate of 10^{-3} , stochastic gradient descent (SGD) optimizer, and other parameters are set to the default values. We conducted a different number of experiments and found the best results found at 10 mini-batch size and 60 epochs. Freezing some earlier layers and online data augmentations were taken place. The augmentation occurred by randomly translating images up to 30 pixels and randomly flipping them along the vertical axis and scaling them horizontally and vertically up to 10% (Fig. 18).

3) SECOND STAGE RESULTS & DISCUSSION

The pre-trained VGG19 model was evaluated based on various performance metrics (accuracy, recall, precision, and

F1 score) also, we computed the model training time and needed time for predicting one chromosome image. To compute these metrics, we first define the following criteria:

- True positive (TP): image classified correctly as a positive class.
- False positive (FP): image classified incorrectly as a positive class.
- False negatives (FN): image classified incorrectly as a negative class.
- True negatives (TN): image classified correctly as a negative class.

Accuracy is calculated in (6) as the ratio between the number of correctly classified images and the total number of images.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (6)$$

Recall and precision are computed according to Equation 2, 3 respectively. F1 score in (7) is the harmonic mean between precision and recall.

$$F1score = \frac{2 * (Recall * Precision)}{(Recall + Precision)} \quad (7)$$

The sections below display the results of the proposed two approaches with different schemes presented in Table 2 on the three datasets.

4) CEGMR DATASET RESULT

Table 9 tabulated the CEGMR dataset classification results and we notice the best results of accuracy equals to 95.04%, recall is 94.84%, precision is 94.90%, and F1 score is 94.87% found on scheme 16 that adds a BN layer after each Conv. layer in the feature extraction part and replaces all FC layers in the classification part with GAP layer. Scheme 9 achieved the best training time 2.48h but in general, all schemes in the second approach except the last scheme achieve better training time comparing with the first approach even for predicting time per chromosome image. Adding BN with GAP layers boost the performance of base VGG19 around 1.11% but it increases training time comparing with other schemes in approach 2. Dropout and BN are techniques of regularization and we examined their effect on VGG19 performance on schemes 6, 7, 14, and 15. BN outperforms dropout in this dataset experiments.

5) BioImLab DATASET RESULT

From Table 10 it is clear that scheme 3 provides the best results for accuracy which is 94.11%, recall of 93.86%, precision of 94.51%, and F1 score of 94.18%. Scheme 3 adds FC layer with 4096 neurons after 'f8' layer in the classification part of VGG19. The smallest training time 2.21h obtained using scheme 14 which replaces the FC layers with GAP layer followed by these layers FC(4096) + ReLu + dropout + FC(4096) + ReLu + dropout. In general approach 2 (adding GAP) produces the smallest prediction time per chromosome image but it didn't reduce training time for this dataset as it reduces in the CEGMR dataset. Dropout outperforms BN in approach 1 whereas in approach 2, BN outperforms dropout.

a: COMBINED DATASET RESULT

As seen from Table 11 the best accuracy result is 94.70%, precision is 94.75%, and F1 score is 94.68% found on scheme 16 that adds a BN layer after each Conv. layer in the feature extraction part and replaces all FC layers in the classification part with GAP layer. Furthermore, scheme 13, which adds extra FC layer with 6000 neurons after 'fc8', accomplished the best recall which is 94.69%. The smallest training time 5.07h has been produced by scheme 10 where extra FC layer with 2048 neurons is added after 'fc8' layer. GAP layer reduces training time for this dataset as it reduces in the CEGMR dataset, but the training time was increased when BN was added in the feature extraction part. Dropout outperforms BN in this dataset experiments.

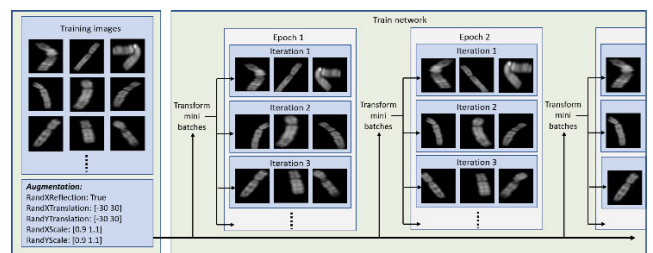


FIGURE 19. Online augmentation for classification.

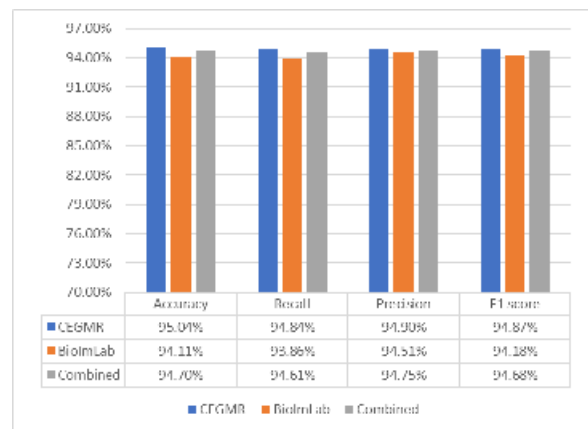


FIGURE 20. Best Classification result on each dataset (CEGMR, BioImLab, combined).

From the above experiments conducted on the three datasets, we observe the proposed two approaches improve the performance over the VGG19 baseline. Fig. 20 visualizes the best result found on each dataset and as we see the best result is obtained on the CEGMR dataset using scheme 16. We use this trained model in chromosomes classification for the third stage. In general, adding GAP layer to VGG19 decreases the training time. Moreover, connecting each convolutional layer in the feature extraction part of VGG19 with BN layer lead to enhance the performance. However, batch normalization and dropout should be used in the network only with caution and experimentation [54].

TABLE 9. Classification results on CEGMR dataset.

Scheme No.	Scheme Name	Accuracy	Recall	Precision	F1 score	Training time (h)	Prediction time per chromosome image (s)
Approach 1							
1	No FC (base model)	93.93%	93.72%	94.20%	93.96%	3.33	0.0113
2	FC(2048)	92.89%	92.97%	93.08%	93.02%	3.29	0.0115
3	FC(4096)	93.85%	93.55%	93.96%	93.75%	3.26	0.0114
4	FC(5000)	94.30%	93.78%	94.13%	93.95%	3.28	0.0114
5	FC(6000)	94.37%	94.14%	94.40%	94.27%	3.27	0.0115
6	2 (FC + ReLu + dropout)	92.22%	91.78%	92.03%	91.90%	3.41	0.0116
7	2 (FC + BN + ReLu)	94.44%	93.74%	94.75%	94.24%	3.41	0.0116
8	Conv. + BN	93.56%	93.23%	94.26%	93.74%	3.92	0.0118
Approach 2							
9	No FC	94.00%	93.46%	94.18%	93.82%	2.48	0.0106
10	FC(2048)	93.70%	93.16%	93.83%	93.50%	2.50	0.0106
11	FC(4096)	94.00%	93.76%	93.70%	93.73%	2.50	0.0106
12	FC(5000)	93.63%	92.94%	93.95%	93.44%	2.51	0.0106
13	FC(6000)	94.15%	93.51%	94.50%	94.00%	2.51	0.0106
14	2 (FC + ReLu + dropout)	93.93%	93.80%	93.67%	93.73%	2.55	0.0106
15	2 (FC + BN + ReLu)	94.59%	94.22%	94.77%	94.49%	2.56	0.0105
16	Conv. + BN	95.04%	94.84%	94.90%	94.87%	3.09	0.0108

TABLE 10. Classification results on BiolmLab dataset.

Scheme No.	Scheme	Accuracy	Recall	Precision	F1 score	Training time (h)	Prediction time per chromosome image (s)
Approach 1							
1	No FC (base model)	91.67%	91.20%	91.80%	91.50%	3.41	0.0136
2	FC(2048)	91.67%	91.44%	92.26%	91.85%	2.77	0.0135
3	FC(4096)	94.11%	93.86%	94.51%	94.18%	3.97	0.0135
4	FC(5000)	92.12%	92.33%	92.69%	92.51%	2.79	0.0135
5	FC(6000)	91.30%	91.13%	91.82%	91.47%	2.79	0.014
6	2 (FC + ReLu + dropout)	92.75%	92.62%	92.96%	92.79%	2.95	0.0135
7	2 (FC + BN + ReLu)	90.76%	90.59%	90.87%	90.73%	3.01	0.0136
8	Conv. + BN	93.66%	93.05%	93.65%	93.35%	4.37	0.014
Approach 2							
9	No FC	91.39%	91.67%	91.48%	91.58%	3.89	0.0125
10	FC(2048)	92.03%	92.26%	92.30%	92.28%	2.57	0.0125
11	FC(4096)	93.39%	92.77%	93.71%	93.24%	2.96	0.0126
12	FC(5000)	92.48%	92.65%	93.21%	92.93%	3.83	0.0126
13	FC(6000)	92.39%	92.47%	92.64%	92.55%	3.11	0.0127
14	2 (FC + ReLu + dropout)	92.30%	92.09%	92.37%	92.23%	2.21	0.0126
15	2 (FC + BN + ReLu)	93.21%	92.62%	93.59%	93.10%	2.22	0.0126
16	Conv. + BN	93.84%	93.26%	94.14%	93.70%	2.98	0.0128

TABLE 11. Classification results on the combined dataset (CEGMR and BiolmLab).

Scheme No.	Scheme	Accuracy	Recall	Precision	F1 score	Training time (h)	Prediction time per chromosome image (s)
Approach 1							
1	No FC (base model)	93.64%	93.40%	94.00%	93.70%	6.57	0.0124
2	FC(2048)	93.36%	92.40%	93.42%	92.91%	6.24	0.0123
3	FC(4096)	92.91%	92.74%	93.19%	92.96%	6.26	0.0124
4	FC(5000)	93.24%	92.93%	93.20%	93.07%	6.23	0.0124
5	FC(6000)	93.24%	92.82%	93.34%	93.08%	6.27	0.0124
6	2 (FC + ReLu + dropout)	94.09%	93.71%	93.96%	93.84%	6.47	0.0124
7	2 (FC + BN + ReLu)	93.52%	93.20%	93.77%	93.49%	6.54	0.0124
8	Conv. + BN	94.30%	93.98%	94.43%	94.20%	7.81	0.0127
Approach 2							
9	No FC	92.26%	91.99%	92.82%	92.40%	6.12	0.0117
10	FC(2048)	94.13%	93.89%	94.10%	94.00%	5.07	0.0117
11	FC(4096)	93.32%	93.10%	93.46%	93.28%	5.09	0.0117
12	FC(5000)	93.19%	92.69%	93.46%	93.07%	5.11	0.0119
13	FC(6000)	94.66%	94.69%	94.64%	94.66%	5.12	0.0118
14	2 (FC + ReLu + dropout)	93.93%	93.63%	94.00%	93.82%	5.22	0.0118
15	2 (FC + BN + ReLu)	93.03%	92.66%	93.06%	92.86%	5.28	0.0123
16	Conv. + BN	94.70%	94.61%	94.75%	94.68%	6.12	0.0118

TABLE 12. Relative comparison with state-of-the-art methods that addressed chromosomes classification using deep learning.

Research paper	Feature extraction	Classifier	Dataset	Performance
Sharma <i>et al.</i> (2018) [8]	ResNet50	LSTM followed by an attention block	5474 chromosome images from BioImLab	• Accuracy: 90.42%
Swati <i>et al.</i> (2018) [34]	Super-Xception network		5474 chromosome images from BioImLab	• Accuracy: 92.36%
Swati <i>et al.</i> (2017) [33]	Siamese Network comprised of twin neural networks and utilized CNN as the base network	1. Siamese Network 2. MLP	1740 chromosome images from a hospital	• Accuracy: 84.6%
Qin <i>et al.</i> (2019) [4]	<ul style="list-style-type: none"> G-Net extracted global features: chromosome's length, shape, and size. L-Net extracted local features: texture patterns of local parts. 	Two MLP	87831 chromosome images from the Xiangya Hospital	<ul style="list-style-type: none"> • Accuracy: 98.9% • F1: 98.7% • Acc. per Case: 98.9% • Acc. per Case-D: 99.2%
Zhang <i>et al.</i> (2018) [39]	CNN consisted of five types of layers: convolution, pooling, dropout, flatten, and dense layers		224 karyotyping images	<ul style="list-style-type: none"> • Accuracy: 92.5% • PWCK: 91.3%
Monika <i>et al.</i> (2017) [42]	CNN consisted of four blocks where every block contains two convolutional layers, one dropout, and one maxpooling layer. Followed by two fully connected layers and a softmax layer with 24 units		400 metaphase images from a hospital	• Accuracy: 86.7%
Yirui <i>et al.</i> (2018) [40]	VGG16 (used Multiple Distribution Generative Advertising Network for augmentation)		120 metaphase images from a private company	• Precision: 63.5%
Somasundaram (2019) [41]	CNN consisted of five types of layers: convolution, pooling, flatten, dense, and dropout layers		more than 500 normal and 500 abnormal images	• Accuracy: 98.9%
Ning <i>et al.</i> (2019) [43]	<ul style="list-style-type: none"> Applying ResNet-50 as backbone for feature extraction. Original and straighten images passed through max-polling layer followed by a convolution layer then Resnet. Cropped images fed directly to the ResNet. Concatenation features. 	MLP which includes two FC layers with 1024 nodes and one softmax layer.	Built dataset from 5000 chromosome images + 20 backgrounds + 50 kinds of distractors collected from hospital	• Accuracy: 95.70%
	VGG19 (using scheme 3)		5474 chromosome images from BioImLab	<ul style="list-style-type: none"> • Accuracy: 94.11% • Recall: 94.84% • Precision: 94.90% • F1 score: 94.87%
Our method	VGG19 (using scheme 6)		147 metaphase images from CEGMR	<ul style="list-style-type: none"> • Accuracy: 95.04% • Recall: 93.86% • Precision: 94.51% • F1 score: 94.18%

6) COMPARISON WITH THE STATE-OF-THE-ART CLASSIFICATION METHODS

Table 12 compares the proposed classification method with other classification systems that extract features using deep learning. The comparison is relative because they used different classification methods and different datasets. Paper [8] used different networks for feature extraction and classification while paper [34] and our proposed method applied the same network for feature extraction and classification on BioImLab dataset. The performance of our proposed method is 94.11% using scheme no. 3 and as we notice we achieved better performance than [8] and [34]. Papers [4], [33], and [43] used different deep learning networks for feature extrac-

tion and classification on their own collected dataset whereas paper [39], [41], and [42] built their own CNN and used it for both feature extraction and classification on their own collected dataset. As obvious from Table 12, our classification method on the CEGMR dataset using scheme no. 16 achieved accuracy of 95.04% which is comparable when compared with the state of the art that used private dataset.

E. THIRD STAGE: ABNORMALITY DETECTION AND INTEGRATED SYSTEM TEST

The model achieved the best result in the above sections, which is scheme 16 trained on CEGMR dataset, is used in classifying chromosomes to recognize the abnormalities. The

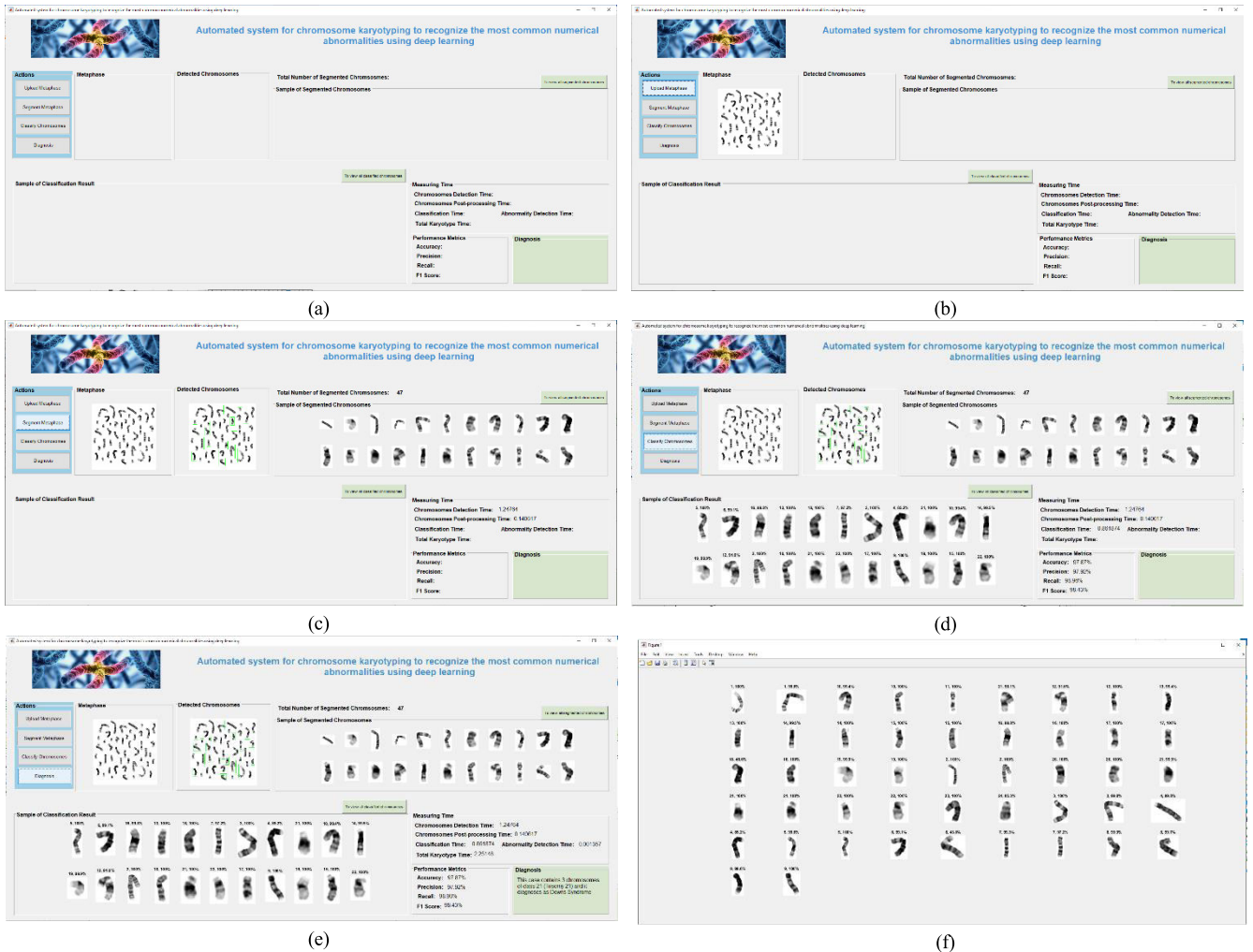

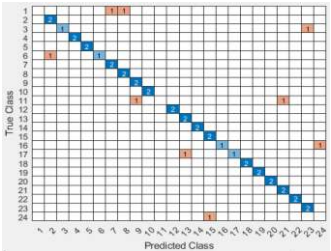


FIGURE 21. GUI for testing the proposed system, (a) main interface (b) uploading metaphase image (c) chromosomes individual chromosomes detection result (d) chromosomes classification result (e) diagnosis result (f) showing all classified chromosomes.

TABLE 13. Successful cases tested by our system.

Metaphase					
Confusion Matrix					
Classification Accuracy	91.49%	97.87%	95.74%	95.74%	97.78%
Actual/ Predicted diagnosis	Trisomy 13	Trisomy 21	Trisomy 18	Trisomy XXY	Monosomy X

TABLE 14. Failed case tested by our system.

Metaphase	Confusion Matrix	Classification Accuracy	Actual diagnosis	Predicted diagnosis
		80.85%	Trisomy XXY	Trisomy 13

input to this stage is the classified chromosomes and the output is the diagnosis. For detecting abnormality, we measured the abnormality detection accuracy in (8) as the ratio of the correctly diagnosed cases to the total number of cases.

$$\begin{aligned}
 & \text{Abnormality detection accuracy} \\
 &= \frac{\text{Correctly diagnosed cases}}{\text{Total number of cases}} \times 100 \quad (8)
 \end{aligned}$$

To test the whole system, we diagnosed 29 metaphases through the system and it can detect the abnormality on all cases with 100% detection accuracy except Trisomy XXY it detects 80% of its cases. Table 13 shows the results of some of the successful test cases and Table 14 shows example result of the failed Trisomy XXY case.

Simulations GUI for the Three Stages: We provide a graphical user interface (GUI) for testing the proposed system as shown in Fig. 21 (a). The interface contains four main operations: upload image, individual chromosomes detection, classify chromosomes, and diagnosis buttons. Initially, the metaphase image that needs to be diagnosed is uploaded using upload image button (Fig. 21 (b)). Clicking individual chromosomes detection button will call YOLOv2 model to localize each chromosome then cropping and post-processing are performed on all chromosomes as seen in Fig. 21 (c).

Classify chromosomes button fed all the separated chromosomes into the classifier to classify them to 23/24 classes and as a result, the confusion matrix and the performance metrics are displayed in Fig. 21 (d). Diagnosing the abnormality based on the number of chromosomes on each of the target classes is seen after pressing Diagnosis button (Fig. 21 (e)). Pressing To view all classified chromosomes button will showing all classified chromosomes as clear in Fig. 21 (f).

V. CONCLUSION AND FUTURE WORK

Recently, the need for automating karyotyping system to recognize abnormalities increased to assist cytogenetics lab technicians and to save their valuable time. In this article, we propose a system to detect individual chromosomes and classify them using deep learning techniques. YOLO v2 has been investigated with some chromosome post-processing to separate each chromosome from the metaphase image.

In the classification stage, we utilized transfer learning in VGG19 pre-trained model for feature extraction and classification which compares well with the state-of-the-art methods that used the Biomedical Imaging Laboratory dataset. We observed via experimentation that the classification accuracy on CEGMR dataset from all experiments give the best accuracy of 95.04%. The fine-tuning VGG19 done by connecting each convolutional layer with the batch normalization layer in the feature extraction part and replacing the top classification layers (fully connected layers) with the global average pooling layer. The final step is abnormality detection and it obtained 96.67% detection accuracy.

For the future work, we will increase the number of metaphase images in the dataset, as, if the training size increases, the accuracy can be improved. Moreover, we will try to work on segment out the overlapped and touched chromosomes as well, we will work to detect structural abnormalities. We plan to use a newer version of YOLO and semantic segmentation using U-Net.

ACKNOWLEDGMENT

This project was funded by the Deanship of Scientific Research (DSR), King Abdulaziz University, Jeddah, under grant no. (DG-21-611-1441). The authors, therefore, gratefully acknowledge the DSR technical and financial support. They would also like to thank Center of Excellence in Genomic Medicine Research (CEGMR) for providing them the dataset and the valuable information.

REFERENCES

- [1] S. L. Gersen, M. B. Keagle, and R. Hall, *The Principles of Clinical Cytogenetics*. Totowa, NJ, USA: Humana Press, 1999.
- [2] D. Patterson, "Molecular genetic analysis of down syndrome," *Hum. Genet.*, vol. 126, no. 1, pp. 195–214, Jul. 2009.
- [3] M. V. Munot, P. M. Joshi, P. Kulkarni, and M. A. Joshi, "Efficient pairing of chromosomes in metaphase image for automated karyotyping," in *Proc. IEEE-EMBS Conf. Biomed. Eng. Sci.*, Dec. 2012, pp. 916–921.
- [4] Y. Qin, J. Wen, H. Zheng, X. Huang, J. Yang, N. Song, Y.-M. Zhu, L. Wu, and G.-Z. Yang, "Varifocal-net: A chromosome classification approach using deep convolutional networks," *IEEE Trans. Med. Imag.*, vol. 38, no. 11, pp. 2569–2581, Nov. 2019.
- [5] F. Abid and L. Hamami, "A survey of neural network based automated systems for human chromosome classification," *Artif. Intell. Rev.*, vol. 49, no. 1, pp. 41–56, Jan. 2018.
- [6] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

- [7] D. Cireşan, U. Meier, and J. Schmidhuber, "Multi-column deep neural networks for image classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2012, pp. 3642–3649.
- [8] M. Sharma, Swati, and L. Vig, "Automatic chromosome classification using deep attention based sequence learning of chromosome bands," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, Jul. 2018, pp. 1–8.
- [9] P. N. Druzhkov and V. D. Kustikova, "A survey of deep learning methods and software tools for image classification and object detection," *Pattern Recognit. Image Anal.*, vol. 26, no. 1, pp. 9–15, Jan. 2016.
- [10] I. Goodfellow, Y. Bengio, and A. Courville, "Deep learning," in *Book in Preparation*, vol. 1. Cambridge, MA, USA: MIT Press, 2016. [Online]. Available: <http://www.deeplearningbook.org>
- [11] A. Lumini, L. Nanni, and G. Maguolo, "Deep learning for plankton and coral classification," *Appl. Comput. Informat.*, to be published, doi: 10.1016/j.aci.2019.11.004.
- [12] S. Hijazi, R. Kumar, and C. Rowen, "Using convolutional neural networks for image recognition," Cadence Des. Syst., San Jose, CA, USA, Tech. Rep., 2015. [Online]. Available: http://site.eet-china.com/webinar/pdf/Cadence_0425_webinar_WP.pdf
- [13] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Proc. Adv. Neural Inf. Process. Syst.*, 2014, pp. 3320–3328.
- [14] P. Marcelino. (Oct. 23, 2018). *Transfer Learning From Pre-Trained Models*. [Online]. Available: <https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751>
- [15] S. J. Pan and Q. Yang, "A survey on transfer learning," *IEEE Trans. Knowl. Data Eng.*, vol. 22, no. 10, pp. 1345–1359, Oct. 2010.
- [16] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: <http://arxiv.org/abs/1409.1556>
- [17] (Feb. 15, 2020). *Train Deep Learning Network to Classify New Images*. [Online]. Available: <https://www.mathworks.com/help/deeplearning/examples/train-deep-learning-network-to-classify-new-images.html>
- [18] (Feb. 15, 2020). *Get Started with Transfer Learning*. [Online]. Available: <https://www.mathworks.com/help/deeplearning/gs/get-started-with-transfer-learning.html>
- [19] S. Minaee, M. Fotouhi, and B. H. Khalaj, "A geometric approach to fully automatic chromosome segmentation," in *Proc. IEEE Signal Process. Med. Biol. Symp. (SPMB)*, Dec. 2014, pp. 1–6.
- [20] I. C. Yilmaz, J. Yang, E. Altinsoy, and L. Zhou, "An improved segmentation for raw G-Band chromosome images," in *Proc. 5th Int. Conf. Syst. Informat. (ICSAI)*, Nov. 2018, pp. 944–950.
- [21] R. Bashmail, L. A. Elrefaie, and W. Alhalabi, "Automatic segmentation of chromosome cells," in *Proc. Int. Conf. Adv. Intell. Syst. Inform. Cham, Switzerland: Springer*, 2019, pp. 654–663.
- [22] R. Hu, J. Karnowski, R. Fadel, and J.-P. Pommier, "Image segmentation to distinguish between overlapping human chromosomes," *CoRR*, vol. abs/1712.07639, 2017. [Online]. Available: <http://arxiv.org/abs/1712.07639>
- [23] H. M. Saleh, N. H. Saad, and N. A. M. Isa, "Overlapping chromosome segmentation using U-net: Convolutional networks with test time augmentation," *Procedia Comput. Sci.*, vol. 159, pp. 524–533, 2019.
- [24] E. Altinsoy, C. Yilmaz, J. Wen, L. Wu, J. Yang, and Y. Zhu, "Raw G-band chromosome image segmentation using U-net based neural network," presented at the Int. Conf. Artif. Intell. Soft Comput., Jun. 16, 2019.
- [25] M. Moradi and S. K. Setarehdan, "New features for automatic classification of human chromosomes: A feasibility study," *Pattern Recognit. Lett.*, vol. 27, no. 1, pp. 19–28, Jan. 2006.
- [26] N. Tabatabaey Mashadi and S. Alireza Seyedin, "Direct classification of human G-banded chromosome images using support vector machines," in *Proc. 9th Int. Symp. Signal Process. Appl.*, Feb. 2007, pp. 1–4.
- [27] M. J. Roshtkhari and S. K. Setarehdan, "Linear discriminant analysis of the wavelet domain features for automatic classification of human chromosomes," in *Proc. 9th Int. Conf. Signal Process.*, Oct. 2008, pp. 849–852.
- [28] L. Vanitha and A. R. Venmathi, "Automatic abnormality detection in chromosomes using hybrid multi-layer neural network," *Int. J. Neural Netw. Appl.*, vol. 4, pp. 37–45, 2011.
- [29] C. Markou, C. Maramis, A. Delopoulos, C. Daiou, and A. Lambropoulos, "Automatic chromosome classification using support vector machines," in *Pattern Recognition: Methods and Applications*. Hong Kong: iConcept Press, 2012, pp. 1–24.
- [30] E. Poletti, E. Grisan, and A. Ruggeri, "A modular framework for the automatic classification of chromosomes in Q-band images," *Comput. Methods Programs Biomed.*, vol. 105, no. 2, pp. 120–130, Feb. 2012.
- [31] S. Gagula-Palalic and M. Can, "Human chromosome classification using competitive neural network teams (CNNT) and nearest neighbor," in *Proc. IEEE-EMBS Int. Conf. Biomed. Health Informat. (BHI)*, Jun. 2014, pp. 626–629.
- [32] A. O. Kusakci, B. Ayvaz, and E. Karakaya, "Towards an autonomous human chromosome classification system using competitive support vector machines teams (CSVMT)," *Expert Syst. Appl.*, vol. 86, pp. 224–234, Nov. 2017.
- [33] Swati, G. Gupta, M. Yadav, M. Sharma, and L. Vig, "Siamese networks for chromosome classification," in *Proc. IEEE Int. Conf. Comput. Vis. Workshops (ICCVW)*, Oct. 2017, pp. 72–81.
- [34] S. Swati, M. Sharma, and L. Vig, "Automatic classification of low-resolution chromosomal images," in *Proc. Eur. Conf. Comput. Vis. (ECCV) Workshops*, Cham, Switzerland: Springer, 2019, pp. 315–325.
- [35] S. Rungruangbaiyok and P. Phukpattaranont, "Chromosome image classification using a two-step probabilistic neural network," *Songklanakarin J. Sci. Technol.*, vol. 32, no. 3, pp. 255–262, 2010.
- [36] S. Saranya, V. Loganathan, and P. S. RamaPraba, "Efficient feature extraction and classification of chromosomes," in *Proc. Int. Conf. Innov. Inf. Comput. Technol.*, Feb. 2015, pp. 1–7.
- [37] R. S. Remya and K. Sabeena, "Automated karyotyping of metaphase chromosome images based on texture features," in *Proc. Int. Conf. Inf. Sci. (ICIS)*, Aug. 2016, pp. 103–106.
- [38] D. Somasundaram, "Structural similarity and probabilistic neural network based human G-band chromosomes classification," *Int. J. Hum. Genet.*, vol. 18, no. 3, pp. 228–237, Apr. 2018.
- [39] W. Zhang, S. Song, T. Bai, Y. Zhao, F. Ma, J. Su, and L. Yu, "Chromosome classification with convolutional neural network based deep learning," in *Proc. 11th Int. Congr. Image Signal Process., Biomed. Eng. Informat. (CISP-BMEI)*, Oct. 2018, pp. 1–5.
- [40] Y. Wu, Y. Yue, X. Tan, W. Wang, and T. Lu, "End-to-end chromosome Karyotyping with data augmentation using GAN," in *Proc. 25th IEEE Int. Conf. Image Process. (ICIP)*, Oct. 2018, pp. 2456–2460.
- [41] D. Somasundaram, "Machine learning approach for homolog chromosome classification," *Int. J. Imag. Syst. Technol.*, vol. 29, no. 2, pp. 161–167, Jun. 2019.
- [42] M. Sharma, O. Saha, A. Sriraman, R. Hebbalaguppe, L. Vig, and S. Karande, "Crowdsourcing for chromosome segmentation and deep classification," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops (CVPRW)*, Jul. 2017, pp. 34–41.
- [43] N. Xie, X. Li, K. Li, Y. Yang, and H. T. Shen, "Statistical karyotype analysis using CNN and geometric optimization," *IEEE Access*, vol. 7, pp. 179445–179453, 2019.
- [44] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 779–788.
- [45] J. Redmon and A. Farhadi, "YOLO9000: Better, faster, stronger," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jul. 2017, pp. 6517–6525.
- [46] J. Hui. (2018). *Real-time Object Detection with YOLO, YOLOv2 and now YOLOv3*. Accessed: May 19, 2020. [Online]. Available: https://medium.com/@jonathan_hui/real-time-object-detection-with-yolo-yolov2-28b1b93e2088
- [47] M. M. Leonardo, T. J. Carvalho, E. Rezende, R. Zucchi, and F. A. Faria, "Deep feature-based classifiers for fruit fly identification (diptera: Tephritidae)," in *Proc. 31st SIBGRAPI Conf. Graph., Patterns Images (SIBGRAPI)*, Oct. 2018, pp. 41–47.
- [48] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 770–778.
- [49] Mathworks. (May 19, 2020). *YOLOv2TransformLayer*. [Online]. Available: <https://www.mathworks.com/help/vision/ref/nnet.cnn.layer.yolov2transformlayer.html>
- [50] Mathworks. (May 19, 2020). *YOLOv2OutputLayer*. [Online]. Available: <https://www.mathworks.com/help/vision/ref/nnet.cnn.layer.yolov2outputlayer.html>
- [51] R. M. Haralick and L. G. Shapiro, *Computer and Robot Vision*. Reading, MA, USA: Addison-Wesley, 1992.
- [52] P. Soille, *Morphological Image Analysis: Principles and Applications*. New York, NY, USA: Springer, 2013.
- [53] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," 2015, *arXiv:1502.03167*. [Online]. Available: <http://arxiv.org/abs/1502.03167>

- [54] C. Garbin, X. Zhu, and O. Marques, "Dropout vs. Batch normalization: An empirical study of their impact to deep learning," *Multimedia Tools Appl.*, vol. 79, nos. 19–20, pp. 12777–12815, May 2020.
- [55] 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.*, vol. 15, no. 1, pp. 1929–1958, 2014.
- [56] M. Lin, Q. Chen, and S. Yan, "Network in network," 2013, *arXiv:1312.4400*. [Online]. Available: <http://arxiv.org/abs/1312.4400>
- [57] A. Al-Shannaq and L. Elrefaei, "Age estimation using specific domain transfer learning," *Jordanian J. Comput. Inf. Technol.*, vol. 6, no. 2, Jun. 2020.
- [58] C. o. E. I. G. M. R. (CEGMR). (2015). *Cytogenetic Service Unit*. [Online]. Available: <https://cegmr.kau.edu.sa/Pages-Cytogenetic-Service-Unit-en.aspx>
- [59] E. Poletti, E. Grisan and A. Ruggeri, "Automatic classification of chromosomes in Q-band images," in *Proc. 30th Annu. Int. Conf. IEEE Eng. Med. Biol. Soc.*, Vancouver, BC, Canada, Aug. 2008, pp. 1911–1914.
- [60] B. Raj. (2018). *Data Augmentation | How to Use Deep Learning When You have Limited Data—Part 2*. Accessed: Feb. 28, 2020. [Online]. Available: <https://medium.com/nanonets/how-to-use-deep-learning-when-you-have-limited-data-part-2-data-augmentation-c26971dc8ced>
- [61] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *J. Big Data*, vol. 6, no. 1, Dec. 2019.

MONA SALEM AL-KHARRAZ received the B.Sc. degree (Hons.) in computer science from King Abdulaziz University, Jeddah, Saudi Arabia, in 2008, where she is currently pursuing the master's degree with the Department of Computer Science, FCIT. She has been a System Developer with the Center of Excellence in Genomic Medicine Research, since 2009. Her research interests include machine learning and deep learning.



LAMIAA A. ELREFAEI (Senior Member, IEEE) received the B.Sc. degree (Hons.) in electrical engineering (electronics and telecommunications) and the M.Sc. and Ph.D. degrees in electrical engineering (electronics) from the Faculty of Engineering at Shoubra, Benha University, Egypt, in 1997, 2003, and 2008, respectively. She held a number of faculty positions at Benha University, as a Teaching Assistant, from 1998 to 2003, an Assistant Lecturer, from 2003 to 2008, and has been a Lecturer, since 2008. She is currently an Associate Professor with the Faculty of Computing and Information Technology, King Abdulaziz University, Jeddah, Saudi Arabia. Her research interests include computational intelligence, biometrics, multimedia security, wireless networks, and nano networks.

MAI AHMED FADEL (Member, IEEE) received the Ph.D. degree in computer science from the Department of Computer Science, University of Exeter, U.K. She was the Head of the Information System (IS) and the Computer Science (CS) Departments, between 2009 and 2016. She is currently a Lecturer in software engineering with the Department of Computer Science, King Abdulaziz University. She has taught courses in software engineering, web development, algorithms, and Java programming. Her research interests include design patterns, and software engineering in general, cloud computing, and news credibility in social networks and high-performance computing (HPC). She is a member of ACM.

• • •