

Vol. 33, No. 3/4, 2024

Machine GRAPHICS & VISION

International Journal

Published by
The Institute of Information Technology
Warsaw University of Life Sciences – SGGW
Nowoursynowska 159, 02-776 Warsaw, Poland

in cooperation with
The Association for Image Processing, Poland – TPO

BRAIN TUMOR CLASSIFICATION USING FEATURE EXTRACTION AND ENSEMBLE LEARNING

Iliass Zine-dine* , Anass Fahfouh , Jamal Riffi , Khalid El Fazazy ,
Ismail El Batteoui , Mohamed Adnane Mahraz  and Hamid Tairi 

*Laboratory of Informatics, Signals, Automatics and Cognitivism (LISAC),
Faculty of Sciences Dhar El Mehraz (FSDM), Sidi Mohamed Ben Abdellah University (USMBA),
Fès, Morocco*

**Corresponding author: Iliass Zine-dine (zinedine.iliass@gmail.com)*

Abstract Brain tumors (BT) are considered the second-principal cause of human death on our planet. They pose significant challenges in the field of medical diagnosis. Early detection is crucial for effective treatment and improved patient outcomes. As a result, researchers' studies that deal with tumor detection play a vital role in early disease prediction in the field of medicine. Despite advancements in medical imaging technologies, accurate and efficient classification of BT remains a complex task. This study aims to address this challenge by proposing a novel method for brain tumor classification utilizing ensemble learning techniques combined with feature extraction from neuroimaging data. In the present paper, we present a novel approach for brain tumor classification that contains ensemble learning methods following the extraction of important features from brain tumor images. Our methodology involves the preprocessing of neuroimaging data, followed by feature extraction using descriptor techniques. These extracted features are then utilized as inputs to ensemble learning classifiers. Experimental results demonstrate the efficacy of the proposed approach in accurately classifying brain tumors with high precision and recall rates. The ensemble learning framework, combined with feature extraction, outperforms several benchmark models and methods commonly used in brain tumor classification, including AlexNet, VGG-16, and MobileNet, in terms of classification accuracy and computational efficiency. The proposed method that integrates ensemble learning techniques with feature extraction from neuroimaging data offers a promising solution for improving the accuracy and efficiency of brain tumor diagnosis, thereby facilitating timely intervention and treatment planning. The findings of this study contribute to the advancement of medical imaging-based classification systems for brain tumors, with implications for enhancing patient care and clinical decision-making in neuro-oncology.

Keywords: brain tumor, Histogram of Oriented Gradients, Discrete Wavelet Transform, ensemble learning.

1. Introduction

The brain is the principal organ of the nervous system, and it is the most complex organ in the human body. It consists of nerve cells and tissues to control the most basic functions of the body, such as muscular movement, breathing, and senses. For the mentioned reasons, early detection of brain tumors represents a crucial task in the medical field [45]; a brain tumor is a form of cancer that affects the central nervous system, according to the definition provided by the World Health Organization (WHO). It was categorized as a deadly disease in 2016 [2]. In general, a brain tumor is described

as a collection of abnormally growing brain cells. The purpose of this study is to classify cancer in MRI images.

The classification of brain tumors is one of the most challenging tasks in the medical field because various criteria are dependent on the structure of the nervous system's tissue and cells [50]. Currently, the development of technology and the digitalization of medical devices help doctors properly detect and classify brain tumors in the early stages; the field of machine learning is focusing on this task. Rajan and Sundar [36] present an architecture for classifying brain cancers using support vector machines (SVM) classifiers and feature extraction like K-means clustering. This classifier is integrated with Fuzzy C- Means (KMFCM) and active contour by level set for tumor segmentation. In another study, Murugan et al. [5] proposed a system including enhancement, transformation, feature extraction, and classification with machine learning models.

The architectures of deep learning are a subset of machine learning that allows computers to make predictions and describe conclusions based on data thanks to their capacity to learn data representations. These methods are widely utilized in medical imaging categorization and are considered one of the most important computational intelligence techniques. In this context, Das et al. [11] presented an approach that consists of two principal steps. First, preprocess the images using different image processing techniques, then classify the preprocessed images using convolutional neural networks (CNN). In this regard, Paul et al. [34] applied two types of neural networks, fully connected and convolutional neural networks, which were used to classify brain images with different tumor types. Thus, Shree et al. [22] proposed a probabilistic neural network (PNN) approach that relies on feature extraction techniques (such as noise suppression, gray level co-occurrence matrix (GLCM), and the growth of brain tumor region segmentation based on DWT) to reduce complexity and improve performance. However, the field of deep learning has several limitations. Firstly, there is a strong use of neural network methods in this field of brain tumor classification, with the lack of feature extraction functions; secondly, deep learning architectures do not work perfectly with small datasets; they need greedy datasets; and finally, deep network training necessitates the meticulous tuning of numerous parameters, and suboptimal tuning can lead to overfitting or underfitting. These restrictions do not allow researchers to find better performance metrics. In consequence, we thought about a method that gives us better prediction scores.

Recently, new research has been based on the extraction of features from the image, which will be fed to the classifiers. Mircea et al. [15] have proposed an approach using different wavelet transforms and support vector machines to detect and classify the brain tumor. In this regard, Nabizadeh and Kubat [30] have studied a method based on the extraction of features with the Gabor wavelet that is able to detect slices that include tumors and delineate the tumor area. Another type of image feature extraction based on the extraction of information from image textures is widely used in medical image analysis. Singh et al. [43] proposed a hybrid technique for automatic classification of MRI

images by first extracting the features using Principal Component Analysis (PCA) and Gray-Level Co-occurrence Matrix (GLCM). The extracted features are fed as input to a Support Vector Machine (SVM) classifier, which classifies the brain image as normal or abnormal. Thus, LBP (Local Binary Pattern) is a commonly used technique for texture description and pattern recognition in images [46]. Another well-known feature extraction technique is the histogram of oriented gradients, which is a computer vision and image processing technique used to detect objects [25].

In this study, we have proposed an approach based on ensemble learning using the stacking model, we utilized two feature descriptors, HOG and DWT, to capture discriminative information from images for use as inputs to classification algorithms. This not only accelerates the training of these models but also helps prevent overfitting. The extracted characteristics were combined into a vector, evaluated, and selected using a RandomForestRegressor to evaluate and select the most essential features. This process reduces the dimensionality of our dataset, improves the interpretability and reliability of our model's decisions, and provides refined inputs for subsequent models. This process ultimately enhances the overall effectiveness and robustness of the classification framework. The proposed system recorded a more satisfactory classification performance reaches 94%. It appears that the functionalities used in this classification task have an important and effective role. Therefore, our approach demonstrates significant value in the early prediction of AD through the advancement of computer vision and machine learning methods, applied within the medical domain.

The following is how the rest of the paper is organized. The related studies are presented in the second Section. Also, our main contribution introduced in this study is described in the final part of that Section. It is followed by a description of the methods and techniques used in our approach in Section 3. Within this Section, in Subsections 3.1 and 3.1.1 the dataset used is presented and described, and in Subsection 3.1.2 the data pre-processing, including data augmentation, is presented. In the following Subsection 3.2 the features derived from images are described, including the Histogram of Oriented Gradients in Subsection 3.2.1, the Discrete Wavelet Transform in Subsection 3.2.2, and their concatenation and feature selection with the RandomForestRegressor in Subsection 3.2.3. The classification methodology is presented in Subsection 3.3, divided into machine learning methods in Subsection 3.3.1 and ensemble learning methods in Subsection 3.3.2. The whole Section 3 on materials and methods is concluded with its discussion in Subsection 3.4.

The experiments and their results are presented in Section 4, with four Subsections: 4.1 on the experimental setting, 4.2 on performance evaluation measures, 4.3 on the results and finally Subsection 4.4 on their discussion. The whole paper is concluded and the perspective for future work is outlined in Section 5.

2. Related work

The primary goal of this section is to review the existing research on employing extraction features, machine learning, and deep learning models to identify and classify brain tumors. There are several works in this area that deal with the early prediction of brain disease, which is based on computer-aided diagnostic methods (CAD) without surgery or invasive methods. Sobhaninia et al. [44] proposed an architecture that is based on an encoder layer and uses post-max-pooling features for residual learning for brain tumor classification.

In this regard, there are several image-processing architectures interested in detecting and classifying tumors. Our current research focuses on the early prediction of brain tumors, which is similar to the work that will be cited. These studies belong to the same medical field and use the same techniques and methods of computer vision and infographics for this task. However our proposed method provided effective results, surpassing various state-of-the-art experiments on the topic of brain tumors in terms of accuracy.

Various deep convolution neural networks have already been trained are used to extract deep features from magnetic resonance (MR) brain images. Kang et al. [19] presented an architecture for classifying brain cancers using a collection of deep features and machine learning classifiers. In this area, Deepak et al. [13] proposed a method for classifying MRI images; this method is based on transfer learning by applying several models of machine learning to the MRI image dataset of the brain tumor, which is already pre-trained on a VGG-16 model of convolutional neural network. Despite the accurate results of these methods, they remain poor, mainly when dealing with large databases. Ari et al. [4] presented a method based on a pre-processing brain tumor dataset (resize, crop lesion, segment lesion, etc.). Kaplan et al. [20] used a feature extraction approach called Local Binary Patterns (LBP), which is a statistical image processing technique that allows us to extract useful and important characteristics from images. In the domain of computer vision, another approach that gives better results at the classification level is based on the combination of methods like concatenation and confusion of vectors of extracted features, there are several approaches use multiple techniques combined to obtain a model more efficient and effective than a model built with a single algorithm. Abbasi et al. [1] used techniques for segmentation and detection to distinguish between different brain regions based on feature extraction from MRI images or learning features like the local binary pattern (LBP) and the histogram of oriented gradients (HOG). Another type of method based on deep learning proves to be very effective while managing vast amounts of data. Mohsen et al. [29] presented a new method for classifying brain tumor images using a deep neural network (DNN) learning method that used fuzzy C-means to segment the images, discrete wavelet transforms (DWT) to

extract the features, principle component analysis (PCA) to reduce the dimensions, and DNN for classification.

The majority of existing medical MR imaging research focuses on the automatic classification and segmentation of tumor regions. Several researchers have recently presented various strategies for detecting and segmenting the tumor region in MR images. Convolutional neural networks are powerful architectures based on deep convolutional layers that automatically extract robust functionality from the input space related to traditional neural network layers. Rehman et al. [40] proposed CNN models such as AlexNet, GoogLeNet, and VGGNet to classify MRI images of brain tumors.

Timely, deep learning is generally applied in the medical industry. The fundamental CNNs that are applied for classification tasks have similar architectures. A CNN architecture is made up of a series of feed-forward layers that employ convolutional filters and pooling layers, following the last pooling layer, CNN uses many fully connected layers to turn the previous layers' 2D feature maps into 1D vectors for classification. In summary, CNNs rely on three characteristics. Firstly, each layer's units get input from the previous layer's units, which are all in the same tiny neighborhood. This technique allows for the extraction of basic features such as edges and corners. Secondly, in the subsequent layers, these features will be merged to detect higher-order features. The concept of shared weights, which is the second crucial attribute, means employing similar feature detectors throughout the image. Thirdly, CNNs frequently have many sub-sampling layers, which are either advantageous or harmful because this information varies for different situations according to the specific position of characteristics [41].

Although CNNs are beneficial in a variety of applications, they have several flaws, particularly in the sub-sampling layers, which provide only a tiny amount of translational invariance and lose the precise location of the most active feature detectors. A capsule neural network (CapsNet) is a sort of artificial neural network (ANN) that can be used to improve model hierarchical relationships in a machine learning system. To classify brain tumors, Afshar et al. [3] proposed a model based on the architecture of CapsNet that allows access to the tumor tissue without distracting it from the central target.

The majority of extant medical MR imaging research focuses on the automatic classification and segmentation of tumor regions. Several researchers have recently presented various strategies for detecting and segmenting the tumor region in MR images; Table 1 represents previous work carried out on different datasets.

Recent research has shown that deep learning techniques are widely used in expert and intelligent systems as well as in medical image analysis. The methodologies described previously presented limits at the level of data processing, more precisely in the feature extraction phase. These approaches took into account only the binary categorization (normal and abnormal) of the MRI image dataset, and they ignored extracting the crucial features and from the images. In addition, throughout the course of our investigation, it became evident that the referenced models exhibit a scarcity of data,

Tab. 1. Related work approaches to classification methods, feature extraction, and accuracy of brain tumor classification.

Authors	Feature Extraction and Classification Methods	Dataset	Accuracy
Díaz-Pernas et al. [14] 2021	Multi-pathway convolutional neural network (CNN)	3064 MRI	97.3%
Das et al. [12] 2019	Advanced Deep Learning-based Solutions (CNN)	3064 MRI	94.39%
Paul et al. [34] 2017	Fully connected and CNN	3064 MRI	91.43%
Kumar and Shree. [22] 2018	Probabilistic neural network (PNN)	650 MRI	95%
Khawaldeh et al. [21] 2017	CNN	587 MRI	91.16%
Hemanth et al. [16] 2019	CNN	220 MRI	94.5%

intricate computational processes, and suboptimal performance. For the mentioned reasons, it is recommended to search for a new approach that exceeds these constraints and gives us better prediction scores.

Real-time performance is a critical factor in medical diagnosis, particularly in emergency situations where timely and accurate decisions are essential for patient treatment and prognosis. Evaluating a model’s inference time and computational resource requirements ensures its suitability for real-world applications. Models designed for such scenarios must balance speed and accuracy to provide reliable diagnostics without compromising computational efficiency [23].

The main contribution of this study can be summarized as follows: during the pre-processing phase, we employed common computer vision and infographic techniques to facilitate subsequent tasks. Then, we utilized two descriptors, HOG and DWT, to extract relevant and significant features, accelerate model training, avoid overfitting, and thereby enhance the overall effectiveness and robustness of the classification framework. These extracted characteristics were combined into a vector, evaluated, and selected using a RandomForestRegressor, and then considered as inputs for classification machine learning algorithms. To validate the effectiveness of our approach, experiments were conducted on a well-known brain tumor dataset, and the results were compared with existing methodologies. Our approach has shown considerable value in the early prediction of brain tumors through advancements in computer vision and machine learning methods and their applications in the medical domain.

3. Material and methods

The general design of our suggested method is described in this section. We have used an approach that consists of two complementary main phases: feature extraction and classification. The features recovered from the first phase will be considered, such as the inputs from the machine learning classifiers of the second phase. The main objective of

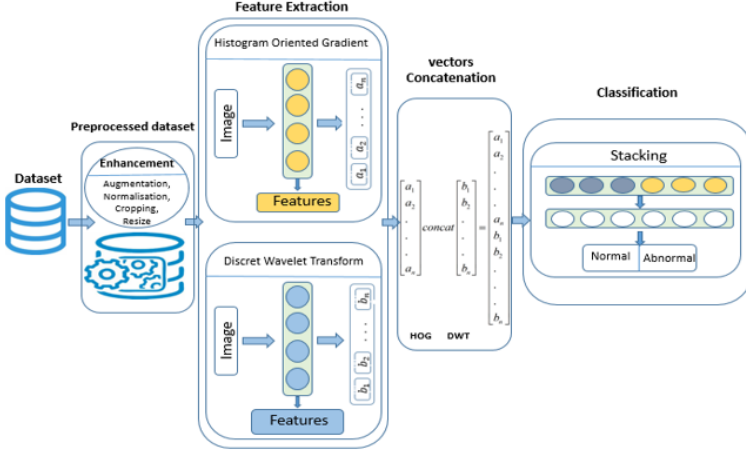


Fig. 1. Proposed architecture of histogram of oriented gradients, discrete wavelet transform, and ensemble learning (Stacking) for brain tumor classification.

this study is to find a higher classification score. Figure 1 illustrates the architecture of our suggested approach to classify brain tumors, which will be described in detail in Sec. 3.3. We have detailed each component of the proposed approach with greater clarity in the following Sections.

3.1. Dataset

The brain tumor dataset utilized in our research is crucial for classification, offering real-world data reflecting clinical complexities. It enables algorithm development and evaluation, facilitating supervised learning and serving as a benchmark for advancing medical image analysis. In this research, we used the free Kaggle brain tumor dataset [32].

3.1.1. Data description

The dataset we have used contains 253 brain MRI images split into two groups: ‘yes’ contains 155 tumorous brain MRI images, and ‘no’ contains 98 non-tumorous brain MRI images. We started to preprocess the dataset by applying imaging methods like normalization, resizing, cropping, and augmentation, to facilitate the employment of the following functions. These techniques applied in the data processing phase allowed us to have 2065 images. Distribution of the dataset across different categories is presented in Table 2.

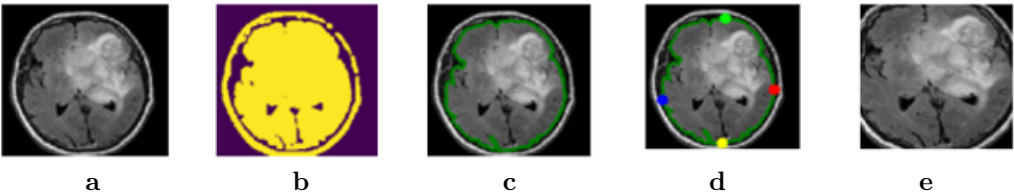


Fig. 2. Image representation of different stages of cropping images from the dataset: (a) original image; (b) thresholded; (c) outer contour (green); (d) edge points (R, G, B, Y); (e) cropped image.

3.1.2. Data pre-processing

The trend of processing datasets containing images for predictive purposes has gained prominence in the domains of computer graphics and computer vision [28]. In this manuscript, we will use image-processing functions. The primary techniques used in this part of the treatment will be discussed below.

Data crop: Nearly all of the images in our brain MRI datasets have undesirable spaces. Hence, it results in subpar classification performance. Therefore, it is vital to crop the pictures in order to eliminate unnecessary portions and use only the pertinent information on [49]. In this study, we employed the cropping approach that computes extreme points and returns a geographic subset of an object based on specifications provided by an extent object. Figure 2 illustrates how the MR images were cropped using an extreme point computation. This cropping method consists of five phases: 1^o we load the original MR images. 2^o We apply thresholding to the MR images in order to create binary images. 3^o We also undertake dilation and erosion processes to reduce image noise. 4^o We determine the four extreme points of the images (extreme top, bottom, right, and extreme left) using the largest contour of the threshold images. 5^o We crop the image based on the edge and extreme point data. Bicubic interpolation is used to enlarge cropped tumor images.

Tab. 2. Division of the dataset of images into training, validation, and testing, and the tumorous and non-tumorous classes.

Step	No. of images			Percentage of images [%]		
	Tumorous, 'yes'	Non-tumorous, 'no'	Total	Tumorous, 'yes'	Non-tumorous, 'no'	Total
Train	885	560	1445	61.3	38.7	70
Validation	190	120	310	61.3	38.7	15
Test	190	120	310	61.3	38.7	15
Total	1265	800	2065	61.3	38.7	100

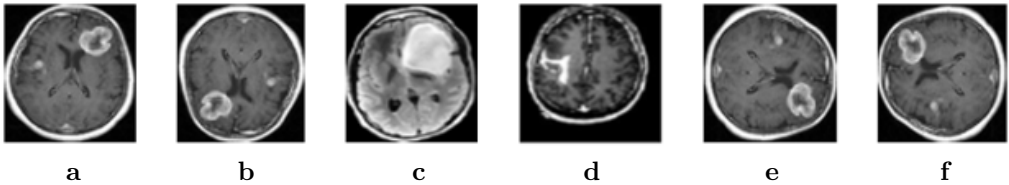


Fig. 3. Example of data augmentation: (a) vertical flip; (b) horizontal flip; (c) brightness increased; (d) vertical shift; (e) rotation $+90^\circ$; (f) rotation -90° .

The cropping function plays a crucial role in feature extraction for classification tasks. Cropping involves removing the outer parts of an image to focus on the most relevant region, which can enhance the performance of classification algorithms. By isolating the area of interest, cropping reduces noise and irrelevant information, leading to a more accurate representation of the essential features. This process not only helps in concentrating on the significant aspects of the image but also reduces the computational complexity by decreasing the amount of data that needs to be processed. Consequently, cropping contributes to improving the efficiency and accuracy of the classification model.

Data augmentation: Due to the relatively modest size of our MRI dataset, we performed image augmentation to increase the size of the dataset. Data augmentation is a technique that involves transforming the original dataset to produce a synthetic dataset, it is a procedure that generates additional training data by applying transformations to existing data to obtain new data [26]. This method involves creating numerous duplicates of the original image with various scales, orientations, locations, brightness, and other characteristics. Results from previous related work showed that augmenting existing data can increase accuracy model classification, rather than collecting new data. Figure 3 illustrates the augmentation techniques applied to the original dataset to generate the new dataset.

The function of image data augmentation is highly beneficial for feature extraction in classification tasks in machine learning. Data augmentation involves creating new training samples by applying random transformations such as rotation, scaling, translation, and flipping to the original images. This technique helps to increase the size and diversity of the training dataset, which is particularly valuable when dealing with limited data. By providing more varied examples, data augmentation allows the model to generalize better to new, unseen data, thereby enhancing its robustness and accuracy. Moreover, it helps to prevent overfitting by ensuring that the model does not memorize the training data but learns to identify the underlying features that are relevant for classification. Consequently, image data augmentation is a crucial step in improving the performance and reliability of machine learning models in image classification tasks.

3.2. Feature extraction

Feature extraction refers to the process of transforming raw data into digital features. These features will be processed while preserving the information from the original dataset. This method performs better than directly applying machine learning to raw data [47]. After the dataset had been pre-processed, we used a descriptor-based approach as a feature extractor to extract pertinent characteristics, speed up the models training; avoid overfitting, and thereby augmenting the overall effectiveness and robustness of the classification framework. Then, we concatenated these characteristics gleaned by HOG and DWT to create a unified input suitable for feeding into the final classifier. Lastly, we trained these extracted features using a RandomForestRegressor model to select the crucial features. In the following two sections, we describe the two descriptors used in our approach HOG and DWT.

3.2.1. Histogram of Oriented Gradients

Histogram of Oriented Gradients (HOG) is a feature descriptor used in computer vision and image processing for object detection; in other words, HOG is a technique for characterizing textures and shapes in an image, similar to Canny Edge Detector and Scale-Invariant Feature Transformation (SIFT) [33].

The HOG has the purpose of quantifying the distribution of local gradient orientations in an image. This approach counts occurrences of gradient orientation in localized parts of an image. This method is comparable to edge orientation histograms, scale-invariant feature transformation descriptors, and shape contexts, but it is more accurate because it is computed on a dense grid of equally spaced cells and employs overlapping local contrast normalization. Four types of normalization are explored. The unnormalized vector containing all the histograms of a single block is denoted by v , its k -norm by $\|v\|_k$, and e is a low-value constant. The normalization factor is then defined by:

- L2-norm:

$$f = \frac{v}{\sqrt{\|v\|_2^2 + e^2}} \quad (1)$$

- L2-hys: L2-norm followed by clipping (limiting the maximum values of v to 0.2) and renormalizing.
- L1-norm:

$$f = \frac{v}{\|v\|_1 + e^2} \quad (2)$$

- L1-sqrt:

$$f = \sqrt{\frac{v}{\|v\|_1 + e^2}} \quad (3)$$

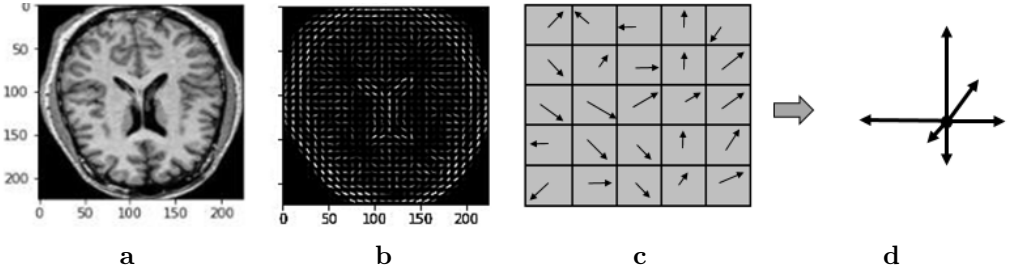


Fig. 4. Representation of features extracted from brain images by the HOG algorithm. (a) Original image; (b) representation of the image with the application of a HOG; (c) gradients in a cell; (d) Histogram of Oriented Gradients in six directions in 3D.

The L2-Hys, L2-norm, and L1-sqrt norms achieve similar performance, while L1-norm performs worse, but still significantly outperforms no normalization. In our approach we applied the first L2-norm normalization. Figure 4 illustrates the application of the HOG algorithm to our dataset.

The integration of the HOG function into our approach holds significant importance for several reasons. Firstly, HOG is widely acknowledged for its ability to capture texture and shape information within an image, making it a powerful tool for feature extraction. By leveraging HOG as an image descriptor in our methodology, we can extract relevant and discriminative features, which are crucial for the task of medical image classification. Furthermore, HOG provides a compact representation of the extracted features, aiding in reducing the dimensionality of the data and enhancing the efficiency of subsequent classification algorithms. By incorporating HOG into our approach, we can improve the quality of the extracted features and, consequently, the accuracy and robustness of our classification model. For our task, we aggregated the features extracted by HOG into a vector of dimensions (1, 1, 1000) to later concatenate it with another vector extracted by DWT, which has the same dimensions.

3.2.2. Discrete Wavelet Transform

Discrete wavelet transform is a data transformation technique that allows the signal to be represented in the form of wavelet coefficients, which can be useful for data compression, feature detection, noise reduction, frequency analysis, and other tasks [27].

Nowadays, DWT is widely used to extract the most relevant features at different orientations and scales from temporal signals or time series data, which can facilitate the modeling and analysis of these data, i.e., Gabor-wavelets capture the local structure of the image corresponding to spatial frequency (scales), space localization, and orientation selectivity [24]. The wavelet coefficients obtained through DWT can be used as features to train machine learning models for classification, regression, or other data analysis

tasks, as it provides localized time-frequency information of a signal using cascaded filter banks of high-pass and low-pass filters to extract features in a hierarchical manner [30].

The principle of the algorithm consists of dividing the image into four blocks at each iteration: three blocks concerning the details of the image, and the fourth corresponding to the most important information for the eye (low frequencies), which serves as a basis for the next iteration. In mathematics, a wavelet ψ is a summable square function of Hilbert space $L^2(R)$, the more often oscillating and with zero mean, chosen as a multi-scale analysis and reconstruction tool. Wavelets are generally found in families, made up of a mother wavelet and the set of its images by the elements of a subgroup of the group of affine transformations of R^n . We thus define a family $\psi_{s,\tau}$ where $(s, \tau) \in R^{+*} \times R$, of wavelets from the mother wavelet ψ :

$$\forall t \in R, \psi_{s,\tau}(t) = \frac{1}{\sqrt{s}} \psi \frac{t - \tau}{s} \quad (4)$$

By extension, families of functions on submanifolds of $\mathbb{R}$ invariant by a transformation group locally isomorphic to the affine group can also be qualified wavelet families.

We use wavelet coefficients for generating the initial features. The wavelet transform is traditionally used for feature extraction. The provision of localized frequency information about a function of a signal is the main advantage of wavelets and is particularly beneficial for classification. Earlier, wavelets were used as a feature extraction method for discrimination; they have advantages in fields like image processing, image watermarking, medical imaging, image compression, and many more. They are also used to denoise medical images. Orthogonal wavelets have always played a main role in biomedical image processing [48].

By applying DWT, we are able to decompose an image into the corresponding sub-bands with their relative DWT coefficients. The DWT is implemented using cascaded filter banks, in which the low pass and high pass filters satisfy certain specific constraints [8]. At each scale of feature extraction by this technique, there are four sub-band images (LL, LH, HH, and HL). The LH, HL, and HH sub-bands may be thought of as the detailed components of the image, while the LL sub-band can be thought of as the approximation component. For DWT decomposition at the next scale, only the sub-band LL is utilized for feature extraction. Additionally, the output feature vector uses the LL sub-band at the final level. Figure 5 illustrates DWT decomposition and its application to our dataset.

The use of DWT in feature extraction for classification tasks offers several key advantages. DWT is highly effective at capturing both spatial and frequency information from images, making it an excellent tool for identifying relevant features. This transform decomposes the image into different frequency components, allowing for the isolation of important details and patterns at various scales. By leveraging DWT, we can extract multi-resolution features that are essential for distinguishing between different classes in

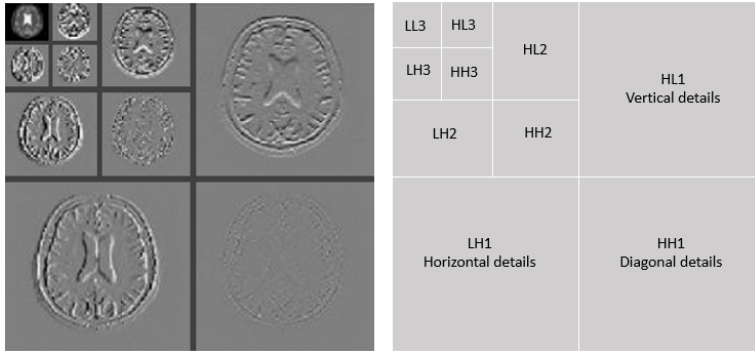


Fig. 5. Representation of the three steps of the features extracted from the brain image by the DWT algorithm.

medical image classification. Additionally, DWT helps to reduce the dimensionality of the data, which not only enhances computational efficiency but also mitigates the risk of overfitting. Consequently, incorporating DWT into our feature extraction process can significantly improve the accuracy and robustness of the classification model.

3.2.3. Concatenation

Following feature extraction, the subsequent procedural step is the concatenation of the extracted features. In the field of machine learning, the concatenation of input vectors in a model is a function that combines the inputs into a single input vector, puts them end to end, and then processes them according to the chosen method [37]. This technique allows for the combination of various sources of information, capturing diverse and complementary aspects of the data, thereby enhancing the overall data representation. Additionally, this approach can aid in reducing the dimensionality of the data by combining multiple features into a single vector, thus facilitating further processing and analysis. Furthermore, concatenating features can lead to more robust and effective classification models by integrating information from different modalities or sources, potentially resulting in improved prediction performance [39]. In our study, we concatenated a vector of (1.1.1000) of feature extracted by HOG with a vector of (1.1.1000) of feature extracted by DWT into a vector of (1.1.2000), then evaluated them by a RandomForestRegressor to select the most relevant ones, reduce the dimensionality of our dataset, and bolster the interpretability and dependability of our model's decisions. Finally, the resulting features were employed as inputs for the classifier. In this research, we employed a 10-fold cross-validation approach to ensure the reliability and statistical significance of the results, as well as to rigorously evaluate the model's performance. The dataset was divided into 10 equal subsets (folds). In each iteration, one fold was

designated as the test set, while the remaining nine folds were used for training the model. This process was repeated 10 times, with each fold serving as the test set exactly once. By averaging the performance metrics across all iterations, we achieved a comprehensive and unbiased assessment of the model's effectiveness.

3.3. Classification

Automatic classification, or supervised classification, is the algorithmic categorization of objects based on statistical data [6]. In our study, our goal is to classify brain tumors. We started by processing our dataset using usual image processing functions such as normalization, resizing, augmentation, and cropping.

Then, we applied two descriptor functions, HOG and DWT, to extract more information and to facilitate the task at the classification stage, to accelerate model training, to prevent overfitting, and thus to enhance the overall efficacy and resilience of the classification framework. The classification step consists of applying several machine learning classification models to our extracted features in order to find a model with high classification quality measures, that would give good results of identification of brain tumors.

Now we shall describe the elements of the classification methodology in detail. Its general structure has been already shown in Fig. 1, p. 9.

3.3.1. Machine learning models

Machine learning models can be conceptualized as algorithms trained to discern patterns in novel data and formulate predictions. These models are mathematically represented as functions designed to process input data, predict outcomes, and yield corresponding outputs [17]. Generally, these models undergo training on a designated dataset and are parameterized to extrapolate predictions for previously unseen data. Below, we list the different classification models used in our approach.

Support Vector Machine: It is one of the most efficient classification algorithms, having been proposed by Vapnik [9]. SVM converts the initial data space into a new space with a higher dimension using the kernel function $K(x_n, x_i)$. The following definition fits the hyperplane function used to separate the data:

$$f(x_i) = \sum_{n=1}^N \alpha_n y_n K(x_n, x_i) + b, \quad (5)$$

where x_n is support vector data (features extracted from brain MR image), α_n is Lagrange multiplier, and y_n represents a target class.

Gaussian Naïve Bayes: The machine learning classifier known as the Naive Bayes classifier operates under the assumption of conditional independence between the attributes and the class [31]. In this study, one of our ML classifiers for classifying brain

tumors is the Gaussian NB classifier. The conditional probability $P(Y|X)$ in the Gaussian NB classifier is determined as the sum of the individual conditional probabilities under the naive independence assumption as follows:

$$P(Y|X) = \frac{P(Y) \times P(X|Y)}{P(X)} = \frac{P(Y) \prod_{i=1}^n P(x_i|Y)}{P(X)} \quad (6)$$

where X is the presented data instance (an extracted deep feature from brain MR image) which is represented by its feature vector $(x_1, \dots, x_n)$, y is a class target (type of brain tumor) with two classes (normal and tumor) for two MRI datasets ensemble learning models.

K-Nearest Neighbors (k-NN): One of the simplest classification methods is k-NN. It makes predictions right away using the training set that is memorized. For example, to categorize a new data instance (a deep feature from a brain MR image), k-NN selects the set of k objects from the training instances that are closest to the new data instance by calculating the distance and assigns the label with two classes (normal or tumor). The selection is based on the majority vote of its k neighbors for the new data instance. The most popular methods for evaluating how close new data instances are to training data examples are Manhattan distance and Euclidean distance [38]. In this study, we applied the k-NN method using the Euclidean distance metric. Data points x and y 's Euclidean distance d is determined as follows:

$$d(x, y) = \sqrt{\sum_{i=1}^n (x_i - y_i)^2} \quad (7)$$

Random Forest: Breiman's ensemble learning technique RF [7] classifies new data instances (a deep feature of a brain MR image) into a class target (a type of brain tumor) with two classes (normal and tumor) for two MRI datasets. It does this by building multiple decision trees using the bagging approach. When building the decision trees, RF randomly chooses n features or attributes to determine the best split point using the Gini index as a cost function. This random selection of attributes results in less correlation between the trees and lower ensemble error rates. To predict the class target of a new data instance, the new observation is fed into all classification trees of the RF. When RF collects the predictions for each class, it chooses the class with the most votes as the new data instance's class label.

3.3.2. Ensemble learning models

Ensemble methods are machine learning techniques that combine several base models to produce one optimal predictive model. In other words, ensemble methods are techniques that combine the predictions of multiple machine learning models (called base learners

or learners) to obtain a more robust and accurate prediction. Ensemble methods are commonly used to improve the performance of machine learning models by reducing overfitting and increasing generalization [42]. One of the most well-liked ensemble machine learning strategies is stacking, which is used to forecast several nodes to create a new model and enhance model performance (in this research, it is the model that gave us effective results among the models used). By stacking, we can train many models to tackle related issues and then create a new model with higher performance based on the output of all the trained models. The three main classes of ensemble learning methods are bagging, stacking, and boosting.

Bagging (Bootstrap Aggregating): This method creates multiple data samples from the training set using bootstrapping (sampling with replacement) and trains a base model on each sample. The predictions from these models are then aggregated (usually using a majority vote) to form a final prediction.

Boosting methods assign weights to training examples based on the performance of previous models. The base models are trained iteratively, with an emphasis on misclassified examples. The predictions from each model are weighted to obtain a final prediction.

Stacking combines several base models using a meta-learning model that learns from the predictions of the base models. The meta-learning model takes the predictions from the base models as input and generates the final prediction.

3.4. Discussion on data and methods

Smaller datasets, despite their limitations, can enhance model generalization when combined with proper preprocessing, enhancement methods, and regularization. They encourage the model to focus on core features, reducing the risk of overfitting. Data augmentation techniques artificially increase variability, further improving generalization. Enhancement methods such as HOG and DWT can also compensate for the small dataset size by effectively extracting critical features. With careful management, small datasets can support a balanced and effective learning framework.

Several attempts have been made to classify brain tumors based on MRI using various machine and ensemble learning classifiers. In this research, we employed seven well-known and diverse classifiers, including Naive Bayes, k-NN, RF, SVM, Gradient Boosting, XGBoost, and Stacking, to determine which classifier works best for MRI-based brain tumor classification. A crucial factor in effectively building the model for MRI-based brain tumor classification is designing a method to generate a discriminative and informative feature from brain MR images. This is because the performance of machine and ensemble learning classifiers heavily depend on the input feature type, i.e., the features extracted from the image and the parameters used in the classifier.

Real-time performance is crucial in medical diagnosis, especially in emergencies requiring fast and accurate decisions. In our study, we evaluated the model's inference

time and achieved classification results within 4 seconds, ensuring a balance between speed and accuracy for practical applications.

4. Experiments and results

In this section, we present the experimental setup and results of our study on brain tumor classification using feature extraction and ensemble learning techniques. Initially, we provide an overview of our approach and the frameworks employed for implementing the code. Following this, we detail the evaluation metrics used to assess the performance of our classification models. Subsequently, we conduct a comprehensive comparison between our approach and related works in the field. Finally, we delve into a discussion of our findings and offer insights into future research directions.

4.1. Experimental setting

In this experiment, following the image pre-processing phase that included normalization, resizing, augmentation, and cropping, we employed two descriptor functions HOG and DWT as feature extractors to extract pertinent features speed up the models training, avoid overfitting, and thereby augmenting the overall effectiveness and robustness of the classification framework. Then, we concatenated these characteristics, evaluated, selected and used them as inputs for the machine learning classifiers. This novel strategy improved our classification prediction score. All trials were carried out on a computer with an NVIDIA GeForce GTX 1070 Ti GPU.

4.2. Performance evaluation

Evaluating the performance of a machine learning model involves employing a range of metrics and techniques to gauge its effectiveness, accuracy, and ability to generalize to new data. These metrics help assess how well the model might perform on unseen data and identify issues like overfitting or underfitting. Our experiment's effectiveness was determined using specific performance metrics tailored to the classification task, including precision, recall, accuracy, and F1-score.

Precision: It represents the percentage of relevant results and is defined as:

$$\text{Precision} = \frac{\text{TruePositive}}{\text{TruePositive} + \text{FalsePositive}} \quad (8)$$

Recall: It denotes the percentage of correctly classified total relevant results by the proposed algorithm and is defined as:

$$\text{Recall} = \frac{\text{TruePositive}}{\text{TruePositive} + \text{FalseNegative}} \quad (9)$$

Tab. 3. Comparison between the performance of machine learning and ensemble learning algorithms with the HOG descriptor as the only feature extraction algorithm.

Group	Model	Performances measures [%]			
		Accuracy	Precision	Recall	F1-Score
Machine Learning	Random Forest	82.3	82.3	82.3	82.3
	Support Vector Machine	88.1	86.4	93.0	89.6
	K-Nearest Neighbors	88.1	88.1	88.1	88.1
	Gaussian Naïve Bayes	61.5	63.4	61.5	60.7
Ensemble Learning	Gradient Boosting	84.2	84.2	84.2	84.2
	XGBoost	83.4	83.4	83.4	83.4
	Stacking	90.6	90.6	90.6	90.6

Accuracy: Formally, accuracy has the following definition:

$$\text{Accuracy} = \frac{\text{TruePositive} + \text{TrueNegative}}{\text{Total}}$$

(10)

F1-score: It is a machine learning measure commonly used in classification models and is defined as:

$$\text{F1-score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

(11)

4.3. Results

The empirical results were derived from the 'Brain MRI Images for Brain Tumor Detection' dataset, obtained from a Kaggle competition dedicated to brain tumor classification tasks. The primary aim of this experiment was to extract features using two distinct descriptors, namely HOG and Wavelet, with the intent to expedite machine learning model training, prevent overfitting, and enhance the overall effectiveness and robustness of the classification framework. These extracted features were concatenated into a vector, evaluated, and selected using the RandomForestRegressor to identify the most significant features, thereby reducing the dimensionality of our dataset, and enhancing the interpretability and reliability of our model's decisions, serving finally as inputs for machine learning classifiers. Subsequently, the outputs of these classifiers were aggregated to identify the most accurate predictions and improve the new model's performance through stacking, resulting in a high-performance classification. Tables 3 and 4 provide the prediction scores achieved by each descriptor across various machine and ensemble learning algorithms.

In summary, our empirical findings indicate that using a single descriptor for feature extraction, combined with the machine learning algorithms in our methodology, results in

Tab. 4. Comparison between the performance of machine learning and ensemble learning algorithms with DWT as the only feature extraction algorithm.

Group	Model	Performances measures [%]			
		Accuracy	Precision	Recall	F1-Score
Machine Learning	Random Forest	80.4	80.4	80.4	80.4
	Support Vector Machine	86.3	84.6	93	87.5
	K-Nearest Neighbors	85.8	85.8	85.8	85.8
	Gaussian Naïve Bayes	60.2	61.7	59.8	59.1
Ensemble Learning	Gradient Boosting	82.3	82.3	82.3	82.3
	XGBoost	81.7	81.7	81.7	81.7
	Stacking	88.5	88.5	88.5	88.5

Tab. 5. Comparison between the performance of machine learning and ensemble learning algorithms with the combination of features extracted from HOG and DWT.

Group	Model	Performances measures [%]			
		Accuracy	Precision	Recall	F1-Score
Machine Learning	Random Forest	82.6	82.6	82.6	82.6
	Support Vector Machine	89.7	88.7	93.2	89.7
	K-Nearest Neighbors	89.8	89.8	89.8	89.8
	Gaussian Naïve Bayes	62.3	63.8	61.5	61.4
Ensemble Learning	Gradient Boosting	85.3	85.3	85.3	85.3
	XGBoost	85.6	85.6	85.6	85.6
	Stacking	91.7	91.7	91.7	91.7

a notably larger classification score compared to similar studies using the same dataset and methodology. To further enhance this classification score, we propose combining both descriptors—HOG and wavelet—to extract additional features. This approach allows us to provide our classifier with enriched inputs, reducing the risk of overfitting.

We then use these extracted features to train a RandomForestRegressor model, which helps identify and select the most essential features, thereby reducing the dimensionality of our dataset and enhancing the interpretability and reliability of our model’s decisions. These selected features are subsequently used as inputs for our models. Table 5 shows the prediction scores for each algorithm using this vector concatenation approach.

Based on a comprehensive analysis of our results, we confidently assert that our approach has yielded robust outcomes, outperforming numerous state-of-the-art experiments, as shown in Tab 6. This success can be attributed to the meticulous phases of data preprocessing, extraction of image texture characteristics, and the synergistic

Tab. 6. Performance comparison between our proposed method and different CNN approaches on the same dataset.

Publication	Classification method	Feature extraction methods	Accuracy
[18]	ResNet-50	CNN	95%
[35]	VGG-16	Lu-Net	90%
[10]	AlexNet	CNN	96%
[19]	FC Layer	CNN	90%
Proposed	Stacking	HOG+Wavelet	92%

use of two descriptors—HOG and DWT—enhanced by a stacking algorithm that integrates multiple base models. Throughout these phases, we emphasize the crucial role of computer vision and infographic functions in enriching our dataset. Additionally, the effectiveness of HOG and DWT techniques in extracting essential features, expediting model training, and preventing overfitting has significantly boosted the overall efficacy and resilience of our classification framework.

The concatenation, assessment, and selection of feature vectors to identify critical features, reduce dataset dimensionality, and enhance the interpretability and reliability of our model’s decisions have further contributed to the success of our approach. Moreover, the incorporation of machine learning algorithms such as RF, SVM, and K-NN, along with the stacking technique, has streamlined the integration of their predictions, thereby improving classification accuracy. In summary, the fusion of these methods has resulted in a novel approach distinguished by superior accuracy.

4.4. Discussion of results

Prior research on early brain tumor prediction has shown varying levels of accuracy, all addressing the same fundamental challenge. The techniques used in each approach—from dataset preprocessing to feature extraction and classification—are crucial factors that highlight the value of our proposed method. In our study, we leverage common computer vision and graphics functions, including normalization, resizing, augmentation, and cropping, along with HOG and DWT feature extraction techniques, to ensure robust and informative input data for the classifier, expedite model training, and prevent overfitting.

Additionally, we combine multiple machine learning algorithms to construct a resilient model, resulting in improved classification scores. The following table presents the classification prediction precision of various methods applied to the same dataset, which originally contained 253 images before preprocessing. The significance of our

study lies in utilizing advanced imaging functions and descriptors to extract relevant information, reduce model training time, and prevent overfitting. These extracted features serve as inputs for our ML classifiers, along with the integration capability of machine learning algorithms such as SVM, K-NN, and RF, which consolidate their predictions into a single model for efficient classification through stacking.

Our research introduces a novel approach to brain tumor classification, combining the advantages of computer vision functions, feature extraction through descriptors, and classification using machine learning models. Unlike prior studies that often overlook preprocessing, we prioritize this stage by incorporating common computer vision and graphics techniques. Additionally, we use HOG and DWT as image descriptors to extract relevant and significant features, ensuring high-quality input data for our model, expediting training, and reducing overfitting—thereby creating a strong foundation for further processing. This integration of preprocessing and feature extraction allows us to capture crucial patterns and nuances essential for accurate disease classification.

Finally, we have employed a stacking model that combines the outputs of three machine learning classifiers to improve classification accuracy. Our approach combines advanced feature extraction techniques and accurate prediction models to surpass previous methods in brain tumor classification. Additionally, we evaluated the model's real-time performance, achieving classification results within a 4-second time frame and a classification accuracy of 92%, demonstrating its efficiency and suitability for practical applications.

5. Conclusion and perspective

In this paper, we have introduced an ensemble learning approach for brain tumor prediction and classification. Our study focuses on two primary phases. Firstly, after preprocessing the dataset using common computer vision and graphics functions including normalization, resizing, augmentation, and cropping, we proceed to the feature extraction phase. Here, we utilize HOG and DWT descriptors to extract salient and relevant features, thereby accelerating ML model training, preventing overfitting, and bolstering the overall effectiveness and robustness of the classification framework. Subsequently, we concatenate these extracted features into a single vector, evaluate, and select them using a RandomForestRegressor, then utilize them as input in the Stacking model to classify the results.

Our approach's effectiveness lies in combining features extracted by the HOG and DWT descriptors and integrating outputs from machine learning classifiers, resulting in a more efficient classification model through stacking. This technique maximizes the predictive power of individual classifiers, enhancing the overall accuracy and efficiency of our classification framework.

The experimental outcomes highlight the remarkable capability of the HOG and

DWT descriptors in extracting crucial features from MR images. Furthermore, the effectiveness of amalgamating ML models is evident in achieving efficient classification metrics. This study holds promise for advancing computer-assisted diagnosis in digital pathology, indicating potential breakthroughs in medical imaging analysis.

Several studies in disease classification often fall short of meeting medical experts' expectations due to issues such as poor performance, data dependency, or reliance on computationally complex deep learning models. In our future endeavors, we aim to investigate alternative large-scale datasets and devise methodologies to overcome these limitations, striving to advance the field of medical image analysis and provide more reliable tools for disease diagnosis and prognosis.

References

- [1] S. Abbasi and F. Tajeripour. Detection of brain tumor in 3D MRI images using local binary patterns and histogram orientation gradient. *Neurocomputing*, 219:526–535, 2016. doi:[10.1016/j.neucom.2016.09.051](https://doi.org/10.1016/j.neucom.2016.09.051).
- [2] P. Afshar, A. Mohammadi, and K. N. Plataniotis. Brain tumor type classification via capsule networks. In: *Proc. 2018 25th IEEE International Conference on Image Processing (ICIP)*, pp. 3129–3133, 2018. doi:[10.1109/ICIP.2018.8451379](https://doi.org/10.1109/ICIP.2018.8451379).
- [3] P. Afshar, K. N. Plataniotis, and A. Mohammadi. Capsule networks for brain tumor classification based on MRI images and coarse tumor boundaries. In: *Proc. ICASSP 2019 - 2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1368–1372, 2019. doi:[10.1109/ICASSP.2019.8683759](https://doi.org/10.1109/ICASSP.2019.8683759).
- [4] A. Ari and D. Hanbay. Deep learning based brain tumor classification and detection system. *Turkish Journal of Electrical Engineering and Computer Sciences*, 26(5):9, 2018. doi:[10.3906/elk-1801-8](https://doi.org/10.3906/elk-1801-8).
- [5] M. Arunachalam and S. Royappan Savarimuthu. An efficient and automatic glioblastoma brain tumor detection using shift-invariant shearlet transform and neural networks. *International Journal of Imaging Systems and Technology*, 27(3):216–226, 2017. doi:[10.1002/ima.22227](https://doi.org/10.1002/ima.22227).
- [6] A. Barragán-Montero, U. Javaid, G. Valdés, D. Nguyen, P. Desbordes, et al. Artificial intelligence and machine learning for medical imaging: A technology review. *Physica Medica: European Journal of Medical Physics*, 83:242–256, 2021. doi:[10.1016/j.ejmp.2021.04.016](https://doi.org/10.1016/j.ejmp.2021.04.016).
- [7] L. Breiman. Random forests. *Machine Learning*, 45:5–32, 2001. doi:[10.1023/A:1010933404324](https://doi.org/10.1023/A:1010933404324).
- [8] S. Charfi, R. Lahmyed, and L. Rangarajan. A novel approach for brain tumor detection using neural network. *International Journal of Research in Engineering & Technology*, 2:93–104, 2014. http://www.impactjournals.us/download/archives/2-77-1406198782-13_Eng-A novel Approach for brain tumor detection using Said Charfi.pdf.
- [9] C. Cortes and V. Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995. doi:[10.1007/BF00994018](https://doi.org/10.1007/BF00994018).
- [10] A. Çınar and M. Yildirim. Detection of tumors on brain MRI images using the hybrid convolutional neural network architecture. *Medical Hypotheses*, 139:109684, 2020. doi:[10.1016/j.mehy.2020.109684](https://doi.org/10.1016/j.mehy.2020.109684).
- [11] S. Das, O. F. M. R. R. Aranya, and N. N. Labiba. Brain tumor classification using convolutional neural network. In: *Proc. 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT)*, pp. 1–5, 2019. doi:[10.1109/ICASERT.2019.8934603](https://doi.org/10.1109/ICASERT.2019.8934603).

- [12] S. Das, S. Bose, G. K. Nayak, and S. Saxena. Deep learning-based ensemble model for brain tumor segmentation using multi-parametric MR scans. *Open Computer Science*, 12(1):211–226, 2022. doi:[10.1515/comp-2022-0242](https://doi.org/10.1515/comp-2022-0242).
- [13] S. Deepak and P. M. Ameer. Brain tumor classification using deep CNN features via transfer learning. *Computers in Biology and Medicine*, 111:103345, 2019. doi:[10.1016/j.combiomed.2019.103345](https://doi.org/10.1016/j.combiomed.2019.103345).
- [14] F. J. Díaz-Pernas, M. Martínez-Zarzuela, M. Antón-Rodríguez, and D. González-Ortega. A deep learning approach for brain tumor classification and segmentation using a multiscale convolutional neural network. *Healthcare*, 9(2):153, 2021. doi:[10.3390/healthcare9020153](https://doi.org/10.3390/healthcare9020153).
- [15] M. Gurbina, M. Lascu, and D. Lascu. Tumor detection and classification of MRI brain image using different wavelet transforms and Support Vector Machines. In: *Proc. 2019 42nd International Conference on Telecommunications and Signal Processing (TSP)*, pp. 505–508, 2019. doi:[10.1109/TSP.2019.8769040](https://doi.org/10.1109/TSP.2019.8769040).
- [16] G. Hemanth, M. Janardhan, and L. Sujihelen. Design and implementing brain tumor detection using machine learning approach. In: *Proc. 2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, pp. 1289–1294, 2019. doi:[10.1109/ICOEI.2019.8862553](https://doi.org/10.1109/ICOEI.2019.8862553).
- [17] L. Hussain, S. Saeed, I. A. Awan, A. Idris, M. S. A. Nadeem, et al. Detecting brain tumor using machines learning techniques based on different features extracting strategies. *Current Medical Imaging*, 15(6):595–606, 2019. doi:[10.2174/1573405614666180718123533](https://doi.org/10.2174/1573405614666180718123533).
- [18] M. Jha, R. Gupta, and R. Saxena. A framework for in-vivo human brain tumor detection using image augmentation and hybrid features. *Health Information Science and Systems*, 10(1):23, 2022. doi:[10.1007/s13755-022-00193-9](https://doi.org/10.1007/s13755-022-00193-9).
- [19] J. Kang, Z. Ullah, and J. Gwak. Mri-based brain tumor classification using ensemble of deep features and machine learning classifiers. *Sensors*, 21(6):2222, 2021. doi:[10.3390/s21062222](https://doi.org/10.3390/s21062222).
- [20] K. Kaplan, Y. Kaya, M. Kuncan, and H. M. Ertunç. Brain tumor classification using modified local binary patterns (LBP) feature extraction methods. *Medical Hypotheses*, 139:109696, 2020. doi:[10.1016/j.mehy.2020.109696](https://doi.org/10.1016/j.mehy.2020.109696).
- [21] S. Khawaldeh, U. Pervaiz, A. Rafiq, and R. S. Alkhawaldeh. Noninvasive grading of glioma tumor using magnetic resonance imaging with convolutional neural networks. *Applied Sciences*, 8(1), 2018. doi:[10.3390/app8010027](https://doi.org/10.3390/app8010027).
- [22] T. N. R. Kumar and N. V. Shree. Identification and classification of brain tumor MRI images with feature extraction using DWT and probabilistic neural network. *Brain Informatics*, 5(1):23–30, 2018. doi:[10.1007/s40708-017-0075-5](https://doi.org/10.1007/s40708-017-0075-5).
- [23] G. Li, J. Sun, Y. Song, J. Qu, Z. Zhu, et al. Real-time classification of brain tumors in MRI images with a convolutional operator-based hidden Markov model. *Journal of Real-Time Image Processing*, 18(4):1207–1219, 2021. doi:[10.1007/s11554-021-01072-4](https://doi.org/10.1007/s11554-021-01072-4).
- [24] Y. Liu, M. Muftah, T. Das, K. Robson, and D. Auer. Classification of MR tumor images based on Gabor-wavelet analysis. *Journal of Medical and Biological Engineering*, 32(1):22–28, 2011.
- [25] D. G. Lowe. Distinctive image features from scale-invariant keypoints. *International Journal of Computer Vision*, 60(2):91–110, 2004. doi:[10.1023/B:VISI.0000029664.99615.94](https://doi.org/10.1023/B:VISI.0000029664.99615.94).
- [26] K. Maharana, S. Mondal, and B. Nemade. A review: Data pre-processing and data augmentation techniques. *Global Transitions Proceedings*, 3(1):91–99, 2022. Proc. International Conference on Intelligent Engineering Approach (ICIEA-2022). doi:[10.1016/j.gltp.2022.04.020](https://doi.org/10.1016/j.gltp.2022.04.020).
- [27] S. Mallat and W. Hwang. Singularity detection and processing with wavelets. *IEEE Transactions on Information Theory*, 38(2):617–643, 1992. doi:[10.1109/18.119727](https://doi.org/10.1109/18.119727).

- [28] R. F. Mansour, J. Escorcía-Gutierrez, M. Gamarra, V. G. Díaz, D. Gupta, et al. Artificial intelligence with big data analytics-based brain intracranial hemorrhage e-diagnosis using CT images. *Neural Computing and Applications*, 35(22):16037–16049, 2023. doi:[10.1007/s00521-021-06240-y](https://doi.org/10.1007/s00521-021-06240-y).
- [29] H. Mohsen, E.-S. A. El-Dahshan, E.-S. M. El-Horbaty, and A.-B. M. Salem. Classification using deep learning neural networks for brain tumors. *Future Computing and Informatics Journal*, 3(1):68–71, 2017. doi:[10.1016/j.fcij.2017.12.001](https://doi.org/10.1016/j.fcij.2017.12.001).
- [30] N. Nabizadeh and M. Kubat. Brain tumors detection and segmentation in MR images: Gabor wavelet vs. statistical features. *Computers & Electrical Engineering*, 45:286–301, 2015. doi:[10.1016/j.compeleceng.2015.02.007](https://doi.org/10.1016/j.compeleceng.2015.02.007).
- [31] N. Naik, M. Matonkar, P. Shatandel, M. A. O. Verekar, and V. Salkar. Detection and classification of brain tumor using naïve Bayes and J48. *International Journal of Science Engineering and Information Technology Research*, 9(2):19–28, 2019.
- [32] M. Nickparvar. Brain tumor MRI dataset. Kaggle, 2021. doi:[10.34740/KAGGLE/DSV/2645886](https://doi.org/10.34740/KAGGLE/DSV/2645886).
- [33] T. Ojala, M. Pietikäinen, and D. Harwood. A comparative study of texture measures with classification based on featured distributions. *Pattern Recognition*, 29(1):51–59, 1996. doi:[10.1016/0031-3203\(95\)00067-4](https://doi.org/10.1016/0031-3203(95)00067-4).
- [34] J. S. Paul, A. J. Plassard, B. A. Landman, and D. Fabbri. Deep learning for brain tumor classification. In: *Proc. Medical Imaging 2017: Biomedical Applications in Molecular, Structural, and Functional Imaging*, vol. 10137, p. 1013710. SPIE, 2017. doi:[10.1117/12.2254195](https://doi.org/10.1117/12.2254195).
- [35] H. M. Rai and K. Chatterjee. Detection of brain abnormality by a novel Lu-Net deep neural CNN model from MR images. *Machine Learning with Applications*, 2:100004, 2020. doi:[10.1016/j.mlwa.2020.100004](https://doi.org/10.1016/j.mlwa.2020.100004).
- [36] P. Rajan and C. Sundar. Brain tumor detection and segmentation by intensity adjustment. *Journal of Medical Systems*, 43(8):282, 2019. doi:[10.1007/s10916-019-1368-4](https://doi.org/10.1007/s10916-019-1368-4).
- [37] V. Rajinikanth, A. N. Joseph Raj, K. P. Thanaraj, and G. R. Naik. A customized VGG19 network with concatenation of deep and handcrafted features for brain tumor detection. *Applied Sciences*, 10(10):3429, 2020. doi:[10.3390/app10103429](https://doi.org/10.3390/app10103429).
- [38] R. H. Ramdlon, E. Martiana Kusumaningtyas, and T. Karlita. Brain tumor classification using MRI images with K-Nearest Neighbor method. In: *Proc. 2019 International Electronics Symposium (IES)*, pp. 660–667, 2019. doi:[10.1109/ELECSYM.2019.8901560](https://doi.org/10.1109/ELECSYM.2019.8901560).
- [39] R. Ranjbarzadeh, A. Caputo, E. B. Tirkolaee, S. Jafarzadeh Ghouschi, and M. Ben-dechache. Brain tumor segmentation of MRI images: A comprehensive review on the application of artificial intelligence tools. *Computers in Biology and Medicine*, 152:106405, 2023. doi:[10.1016/j.combiomed.2022.106405](https://doi.org/10.1016/j.combiomed.2022.106405).
- [40] A. Rehman, S. Naz, M. I. Razzak, F. Akram, and M. Imran. A deep learning-based framework for automatic brain tumors classification using transfer learning. *Circuits, Systems, and Signal Processing*, 39(2):757–775, 2020. doi:[10.1007/s00034-019-01246-3](https://doi.org/10.1007/s00034-019-01246-3).
- [41] J. Seetha and S. S. Raja. Brain tumor classification using convolutional neural networks. *Biomedical & Pharmacology Journal*, 11(3):1457–1461, 2018. doi:[10.13005/bpj/1511](https://doi.org/10.13005/bpj/1511).
- [42] A. Shafi, M. B. Rahman, T. Anwar, R. S. Halder, and H. E. Kays. Classification of brain tumors and auto-immune disease using ensemble learning. *Informatics in Medicine Unlocked*, 24:100608, 2021. doi:[10.1016/j.imu.2021.100608](https://doi.org/10.1016/j.imu.2021.100608).
- [43] D. Singh and K. Kaur. Classification of abnormalities in brain MRI images using GLCM, PCA and SVM. *International Journal of Engineering and Advanced Technology*, 1(6):243–248, 2012. <https://www.ijeat.org/portfolio-item/F0676081612/>.

- [44] Z. Sobhaninia, N. Karimi, P. Khadivi, R. Roshandel, and S. Samavi. Brain tumor classification using medial residual encoder layers. *arXiv*, 2019. ArXiv:2011.00628. doi:[10.48550/arXiv.2011.00628](https://doi.org/10.48550/arXiv.2011.00628).
- [45] Z. N. K. Swati, Q. Zhao, M. Kabir, F. Ali, Z. Ali, et al. Brain tumor classification for MR images using transfer learning and fine-tuning. *Computerized Medical Imaging and Graphics*, 75:34–46, 2019. doi:[10.1016/j.compmedimag.2019.05.001](https://doi.org/10.1016/j.compmedimag.2019.05.001).
- [46] F. Tajeripour, E. Kabir, and A. Sheikhi. Fabric defect detection using modified local binary patterns. *EURASIP Journal on Advances in Signal Processing*, 2008(1):783898, 2008. doi:[10.1155/2008/783898](https://doi.org/10.1155/2008/783898).
- [47] D. P. Tian. A review on image feature extraction and representation techniques. *International Journal of Multimedia and Ubiquitous Engineering*, 8(4):385–395, 2013.
- [48] S. Wang, M. Chen, Y. Li, Y. Zhang, L. Han, et al. Detection of dendritic spines using wavelet-based conditional symmetric analysis and regularized morphological shared-weight neural networks. *Computational and Mathematical Methods in Medicine*, 2015(1):454076, 2015. doi:[10.1155/2015/454076](https://doi.org/10.1155/2015/454076).
- [49] J. Yan, S. Lin, S. B. Kang, and X. Tang. Learning the change for automatic image cropping. In: *Proc. 2013 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 971–978, 2013. doi:[10.1109/CVPR.2013.130](https://doi.org/10.1109/CVPR.2013.130).
- [50] I. Zine-dine, J. Riffi, K. El Fazazi, M. A. Mahraz, and H. Tairi. Brain tumor classification using machine and transfer learning. In: *Proc. 2nd International Conference on Big Data, Modelling and Machine Learning (BML)*, pp. 566–571. INSTICC, 2022. doi:[10.5220/0010762800003101](https://doi.org/10.5220/0010762800003101).



Iliass Zine-dine: Ph.D. in Computer Science at the Laboratory of Computer Science, Signals, Automation, and Cognitivism (LISAC), Faculty of Science, Dhar El Mahraz.



Anass Fahfouh: Doctor of Computer Science at the Laboratory of Computer Science, Signals, Automation, and Cognitivism (LISAC), Faculty of Science, Dhar El Mahraz.



Jamal Riffi: Doctor of Computer Science at the Laboratory of Computer Science, Signals, Automation, and Cognitivism (LISAC), Faculty of Science, Dhar El Mahraz.



Khalid El Fazazy: Doctor of Computer Science at the Laboratory of Computer Science, Signals, Automation, and Cognitivism (LISAC), Faculty of Science, Dhar El Mahraz.

Ismail El Batteoui: Doctor of Computer Science at the Laboratory of Computer Science, Signals, Automation, and Cognitivism (LISAC), Faculty of Science, Dhar El Mahraz.



Mohamed Adnane Mahraz: Doctor of Computer Science at the Laboratory of Computer Science, Signals, Automation, and Cognitivism (LISAC), Faculty of Science, Dhar El Mahraz.



Hamid Tairi: Doctor of Computer Science at the Laboratory of Computer Science, Signals, Automation, and Cognitivism (LISAC), Faculty of Science, Dhar El Mahraz.

CLASSIFICATION OF MAIZE GROWTH STAGES USING DEEP NEURAL NETWORKS WITH VOTING CLASSIFIER

Justyna S. Stypułkowska^{1,2,*} , Przemysław Rokita² 

¹Lukasiewicz Research Network – Institute of Aviation, Warsaw, Poland

²Warsaw University of Technology, Warsaw, Poland

*Corresponding author: Justyna S. Stypułkowska (justyna.stypulkowska@ilot.lukasiewicz.gov.pl)

Abstract Deep learning significantly supports key tasks in science, engineering, and precision agriculture. In this study, we propose a method for automatically determining maize developmental stages on the BBCH scale (phases 10-19) using RGB and multispectral images, deep neural networks, and a voting classifier. The method was evaluated using RGB images and multispectral data from the MicaSense RedEdge MX-Dual camera, with training conducted on HTC_r50, HTC_r101, HTC_x101, and Mask2Former architectures. The models were trained on RGB images and separately on individual spectral channels from the multispectral camera, and their effectiveness was evaluated based on classification performance. For multispectral images, a voting classifier was employed because the varying perspectives of individual spectral channels made it impossible to align and merge them into a single coherent image. Results indicate that HTC_r50, HTC_r101, and HTC_x101 trained on spectral channels with a voting classifier outperformed their RGB-trained counterparts in precision, recall, and F1-score, while Mask2Former demonstrated higher precision with a voting classifier but achieved better accuracy, recall, and F1-score when trained on RGB images. Mask2Former trained on RGB images yielded the highest accuracy, whereas HTC_r50 trained on spectral channels with a voting classifier achieved superior precision, recall, and F1-score. This approach facilitates automated monitoring of maize growth stages and supports result aggregation for precision agriculture applications. It offers a scalable framework that can be adapted for other crops with appropriate labeled datasets, highlighting the potential of deep learning for crop condition assessment in precision agriculture and beyond.

Keywords: AI, deep learning, image recognition, RGB imaging, multispectral imaging, voting classifier, precision farming, determining growth stages of maize, BBCH scale.

1. Introduction

Knowledge of the developmental phases of plants and their precise determination for individual locations are crucial for calculating plant condition parameters within larger areas of semi-cultivated fields, in line with the concept of precision agriculture [12,21,22]. A measure commonly used by scientists to quantify the developmental phase of a plant is in is the international plant development scale BBCH [17,21]. The BBCH scale (Biologische Bundesanstalt, Bundessortenamt und Chemische Industrie) is a standardized system for identifying the phenological development stages of plants. It uses a two-digit coding system where the first digit represents the principal growth stage (e.g., germination, leaf development, flowering), and the second digit provides a more detailed subdivision of each stage (e.g., the number of leaves developed) [17]. This scale allows for consistent documentation and comparison of growth stages across different plant species and has

been widely adopted by researchers, agronomists, and farmers for crop monitoring and management [17, 21].

The BBCH scale is widely utilized by agronomists, researchers, and farmers to monitor and document the growth stages of crops. It aids in standardizing the timing for agricultural practices such as fertilizing, and pesticide application, ensuring optimal crop management and productivity [16, 21, 22]. Until now, the determination of developmental phases of specific plant species has only been done manually by visually analyzing the plants [17]. Automating this process allows faster analysis, which will have a directly impacts on timely human intervention and help provide plants with the right conditions for development. These facts underline the need to develop a robust and rapid method of assessing developmental phases, which can be carried out in an automated manner.

Artificial intelligence comes to the aid of this process [16, 21, 24]. The use of deep learning techniques and the appropriate preparation of new training datasets make it possible to develop trained models capable of detecting the indicated plants and determining their developmental phases based on image analysis [12]. The automatic detection and classification of the BBCH phases of plant development using artificial intelligence is still something of a novelty at present, but it is certainly the direction of the future in precision agriculture [22]. The automation of this process with a defined accuracy and speed, using deep learning algorithms, is therefore a very valuable and desirable advancement compared to the current method of manually determining these parameters. The development of this issue is heavily dependent on the creation of a dedicated dataset [24]. We were the first to create datasets composed of images representing maize plants growing in a real crop field. One dataset consists of RGB images, and the other consists of images acquired in 10 spectral channels. In these datasets, we assigned each plant a corresponding BBCH scale value (from stage 10 to stage 19). This enabled us to develop a method to train AI algorithms to create a model capable of analyzing the images and automatically determining the developmental phases of the plants under study, in this case maize, from the images. Without the use of a similar solution, assessing the quality of plant development parameters on a large scale in a controlled environment is unattainable, given the enormous time and effort required from participants. Our solution supports crop management from its initial stage and can support yield early enough so that the amount of food produced can be easily and efficiently increased. Our method is the start of research into accurately determining the early stages of plant development (in this case maize) from close range and is far superior to existing manual methods in terms of efficiency.

In this paper, we focus on the replication of results using the following deep learning architectures: HTC_r101, HTC_r50, HTC_x101, and Mask2Former [20, 30, 32, 33]. These architectures demonstrate optimal performance in terms of the training process and do not require excessive computational resources for training. We investigate their efficiency and effectiveness when trained on a set of RGB and multispectral images.

In addition, we conducted tests by dividing the multispectral dataset into individual single spectral channels, which provided an answer as to which of the tested algorithms performs best on the indicated datasets and which dataset to use for the detection and classification of developmental phases to achieve the best results. We have also introduced an additional method using a voting classifier, in which models trained on individual spectral channels vote on the final result of selecting a class denoting a specific developmental phase on the BBCH scale. This novel research expands our scope of data analysis to other spectra beyond the previously popular RGB imaging. Our work opens up a new avenue to explore new questions and inspires us to continue our research with a new dataset combining spectral channels and to use another multispectral camera for this purpose as well.

2. Related works

Assessing plant growth stages is crucial for determining their condition parameters [21]. In precision agriculture, an additional requirement is the automation of this process and its accuracy, even at the level of individual plants or small areas within large fields [21]. This ensures proper control over plant development conditions and helps maintain high food quality.

Automatic determination of plant growth stages and conditions has significantly advanced thanks to deep learning and image analysis [22]. Traditional methods rely on manually inspecting plants to document their growth stages, while modern approaches use automated representation learning from images to predict outcomes and assign growth stage values to plants, typically using the BBCH scale [12].

Several studies have explored the use of deep learning models to determine plant condition parameters, including the classification of maize growth stages. This addresses the increasing demand for such solutions in agriculture, particularly in precision farming. These solutions are being developed to meet the need for effective crop management and the monitoring of condition parameters.

For example, Xu et al. [30] proposed a deep learning approach for determining maize growth stages by counting leaves. They developed a two-step method combining instance segmentation and object detection, employing Mask R-CNN and YOLOv5 architectures. This method effectively detected and counted leaves, overcoming challenges related to background and weeds. Using RGB images captured by UAVs, their approach represents a significant advancement in precision agriculture.

Liu et al. [20] developed a system to measure maize seedling emergence by evaluating count, size, uniformity, and distribution. Using deep learning with UAV-captured RGB images, they overcame challenges like shadows and planting density. The system, based on the YOLO architecture and TOPSIS method, accurately assessed seedling quality and identified areas with poor emergence in experimental fields.

Yu et al. [32] used deep convolutional neural networks (DCNNs) to estimate maize aboveground biomass (AGB) from multisource UAV imagery. They showed that AGB estimation, essential for crop growth assessment, can be effectively modeled using regression between AGB and agronomic traits from UAV data. DCNNs provided superior results, especially during the vegetative phase.

Zhang et al. [33] developed a method for detecting maize tassels in UAV-captured RGB images. They highlighted that tassel developmental stage and branch number are key phenotypic traits for assessing growth, pollen quantity, and planning tassel pruning in seed fields. Using a Random Forest classifier and the VGG16 network, their algorithm effectively detected tassels in complex field conditions, improving crop management and yield quality assessment.

Yao et al. [31] proposed a method for classifying maize growth stages using phenotypic traits and UAV-captured data. They combined vegetation indices (VI), textural features (TF), and phenotypic parameters like leaf chlorophyll content (LCC), leaf area index (LAI), fractional vegetation cover (FVC), and canopy height (CH). The highest accuracy (95.1%) was achieved with a Random Forest classifier using LCC, LAI, FVC, and CH. The study showed phenotypic features outperform vegetation indices, and integrating UAV data with machine learning enables accurate maize growth stage monitoring.

Bera et al. [3] proposed PND-Net, a system combining graph convolutional networks (GCN) with traditional CNNs to classify plant nutrient deficiencies and diseases. The model integrates local leaf image features (Xception, ResNet-50, Inception-V3, MobileNet-V2) with spatial relationships captured by GCN, using spatial pyramid pooling (SPP) for multi-scale feature aggregation. Tests on datasets (banana, coffee, potato, PlantDoc) showed high performance: 90.00%, 90.54%, 96.18%, and 84.30%, respectively. PND-Net also achieved state-of-the-art results in medical image classification (BreakHis, SIPaKMeD), making it valuable for precision agriculture and medicine.

Bera et al. [4] proposed RAFA-Net, a method combining CNNs with a regional attention mechanism for food classification and plant stress recognition. The model captures contextual information and long-range dependencies using spatial pyramid pooling (SPP) and average pooling. Tested on food datasets (UECFood-100, UECFood-256, MAFood-121) and plant stress datasets (IP-102, PlantDoc-27), RAFA-Net achieved top accuracies of 91.69%, 91.56%, 96.97%, 92.36%, and 85.54%. The results highlight RAFA-Net's effectiveness in precision agriculture and food processing.

Wu et al. [28] proposed an innovative approach for identifying strawberry diseases using a deep learning model based on the Squeeze-and-Excitation (SE) mechanism. The system integrates sensor data acquisition and plant imaging, transmitting images to the cloud via a dedicated gateway for analysis. This solution enables efficient monitoring of strawberry health, improving crop management and yields.

Bompani et al. [5] explore the implementation of computer vision algorithms on a

heterogeneous multicore microcontroller to accelerate pest detection, specifically targeting the codling moth in apple orchards. Sensor nodes with cameras capture and process images locally, thereby reducing transmission delays. This approach improves real-time pest monitoring, which is crucial for protecting crops and minimizing losses.

Bansal et al. [1] proposed PA-RDFKNet, a deep learning model integrating RGB and hyperspectral imaging for plant age estimation. By combining features from both modalities, PA-RDFKNet significantly improves accuracy over single-modality methods. This approach supports precision agriculture by enhancing plant growth monitoring and optimizing agronomic practices.

Bera et al. [2] introduced APDC (Attention-based Plant Disease Classification), a method using CNNs with an attention mechanism to identify plant diseases from leaf images. The model extracts features, highlights key regions, and classifies them with a softmax layer. Tested on PlantPathology, PaddyCrop, PaddyDoctor, and PlantVillage datasets, APDC achieved accuracies of 97.74%, 99.16%, 99.62%, and 99.97%. This end-to-end trainable model, using lightweight CNNs like MobileNet-v2 and DenseNet-169, is efficient for precision agriculture.

Based on the reviewed articles, most researchers successfully use deep learning models to determine growth stages and other plant condition parameters. They primarily employ RGB imaging techniques captured by UAVs for this purpose. As we have observed, some of the most powerful deep learning models, such as HTC and Mask2Former, have not yet been explored for this specific task. The literature review also revealed that RGB imaging is typically used for determining plant growth stages, including maize, rather than multispectral imaging. This has led to gaps in knowledge regarding whether HTC and Mask2Former can be effectively applied and implemented for maize growth stage classification. Another research gap is the limited use of multispectral imaging for determining plant growth stages, particularly for maize. This article aims to address these research gaps. We intend to investigate which type of imaging (RGB or multispectral) is more suitable for this task and which of the examined algorithms proves to be the most effective. The goal of the research is to select the most effective configuration in the form of: imaging type plus the chosen algorithm from the following options: HTC_r50, HTC_r101, HTC_x101, and Mask2Former.

3. Data and methods

3.1. Datasets

In order to carry out the research, the results of which we present in this article, we used specially prepared datasets of tagged images of maize at different stages of development, divided into RGB and multispectral sets. Data were collected from the same test plots, during the 2021–2022 growing seasons. We collected the data using an RGB camera

Tab. 1. Parameters of RGB and Multispectral Datasets

Dataset Type	Camera	Image Count	Resolution	Total Labeled Objects	Objects per Class (BBCH 10–19)
RGB	RGB Camera	396	12 MP	884	83, 83, 137, 83, 83 83, 83, 83, 83, 83
Multispectral	Micasense RedEdge-MX DUAL (10 spectral channels)	556 per channel	5 MP	9070 (907 per channel)	82, 86, 162, 82, 82 83, 84, 83, 83, 80 (per channel)

and a MicaSense RedEdge-MX DUAL multispectral camera, which captures 10 spectral channels covering the 400 to 900 nm range.

The RGB camera used automatic white balance and exposure settings, eliminating the need for additional manual calibration. For the multispectral camera, we used standard calibration procedures involving reference panels with known reflectance values. This ensured consistency and accuracy of the spectral data captured across different channels.

The datasets include images captured under varying weather conditions, different levels of sunlight, and at various times of the day to maximize diversity. The images were labeled according to the international BBCH plant development scale, adapted for maize, as shown in Figure 1. This scale reflects the number of leaves developed by a plant, with values ranging from 10 to 19, where the tens digit represents the leaf stage, and the ones digit indicates the number of leaves.

Table 1 summarizes the detailed parameters of both datasets, including image count, resolution, total labeled objects, and distribution of objects across BBCH phases.

After labeling the images using the Label Studio environment with the polygon method, the datasets were divided into training and validation sets. The training sets comprised 70% of the data, while the validation sets comprised 30%, with stratification ensuring an even distribution of BBCH phases across the sets.

Examples from each dataset are shown in Figure 2, which displays RGB images of maize at different stages of development, captured under various weather conditions and at different times of the day. Figure 3 presents images taken by the individual lenses of the MicaSense RedEdge-MX DUAL multispectral camera, alongside a comparative image captured by the RGB camera.

3.2. Methods

In this section, we provide a detailed description of the methods used in our study. Figure 4 illustrates the general scheme of the proposed maize growth stage classification system.

Our system employs three primary classification approaches: classification on RGB

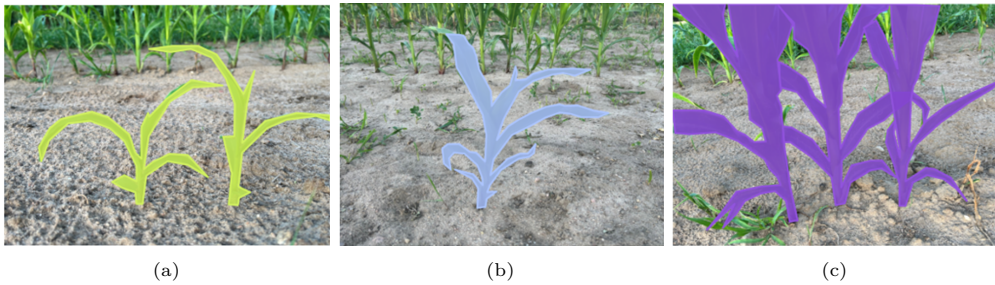


Fig. 1. Examples of determining BBCH scale values for maize and labelling them on images using 'the polygon method', (a) maize at stage 14 of the BBCH scale, (b) maize at stage 18 of the BBCH scale, (c) maize at stage 19 of the BBCH scale.

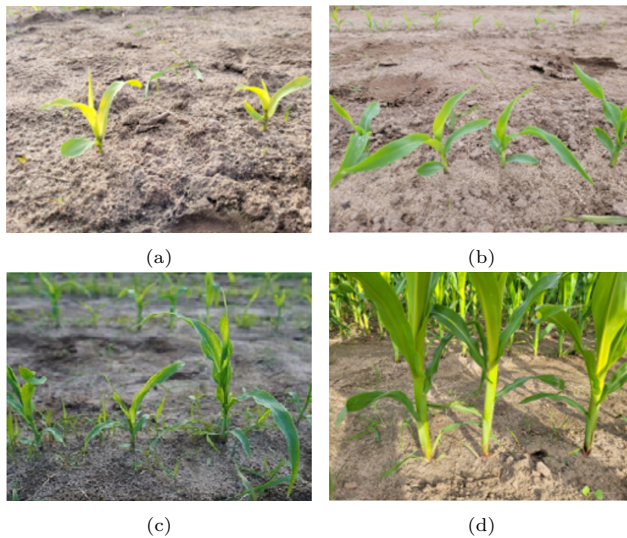


Fig. 2. Examples of RGB images showing maize at various stages of development, captured under different weather conditions and times of day.

images, classification on individual spectral channels, and classification using a voting classifier. Each approach uses deep learning models, including HTC_x101, HTC_r101, HTC_r50, and Mask2Former, trained on corresponding datasets consisting of RGB or individual spectral channels. The best-performing algorithm was selected based on the highest accuracy achieved during the training process.

The first approach involves the classification of RGB images using the best algorithm identified through model evaluation. In the second approach, classification is performed

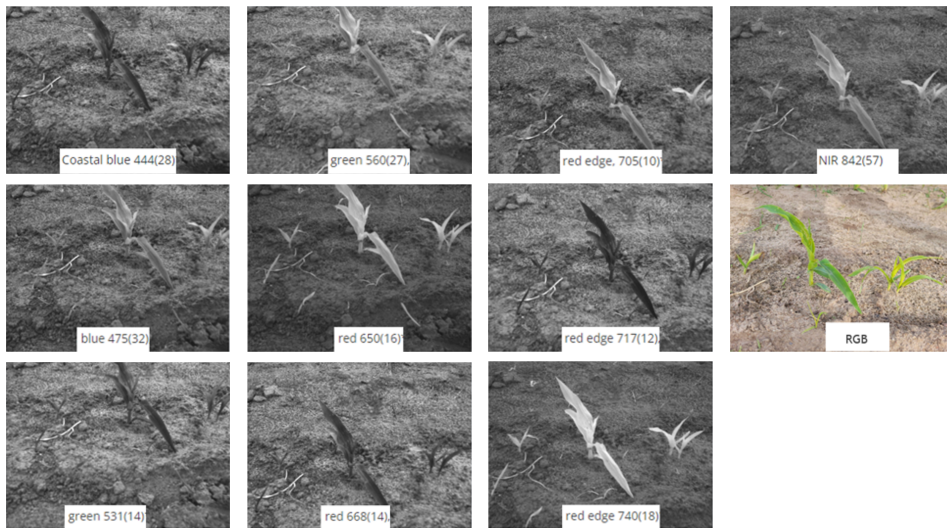


Fig. 3. Examples of images captured from individual spectral channels and an RGB camera, demonstrating various bands and perspectives.

using 10 different models, each trained on a separate spectral channel, with the best-performing algorithm applied to each channel. The third approach utilizes a voting classifier, which combines the results from the 10 models trained on individual spectral channels. The final maize growth stage is determined based on the consensus of these models. A more detailed explanation of each method is provided in the subsequent sections of the article.

3.2.1. Models architectures

During our research, we used the following deep artificial neural network architectures for image analysis: HTC_x101, HTC_r101, HTC_r50, Mask2Former [8,10,15,29]. The highlights of these architectures we describe below.

HTC_x101, HTC_r101 and HTC_r50

HTC, or Hybrid Task Cascade, is an advanced model architecture used for simultaneous object detection and instance segmentation tasks. Proposed by SenseTime Research, HTC is renowned for its high accuracy in benchmarks like COCO [7]. The HTC architecture includes configurations like HTC_x101, HTC_r101, and HTC_r50, varying by the backbone used. HTC_x101 employs the ResNeXt-101 backbone, featuring 101 layers and grouped convolutions to enhance efficiency and accuracy [29]. HTC_r101 utilizes the ResNet-101 backbone, which includes a residual mechanism to improve gradient

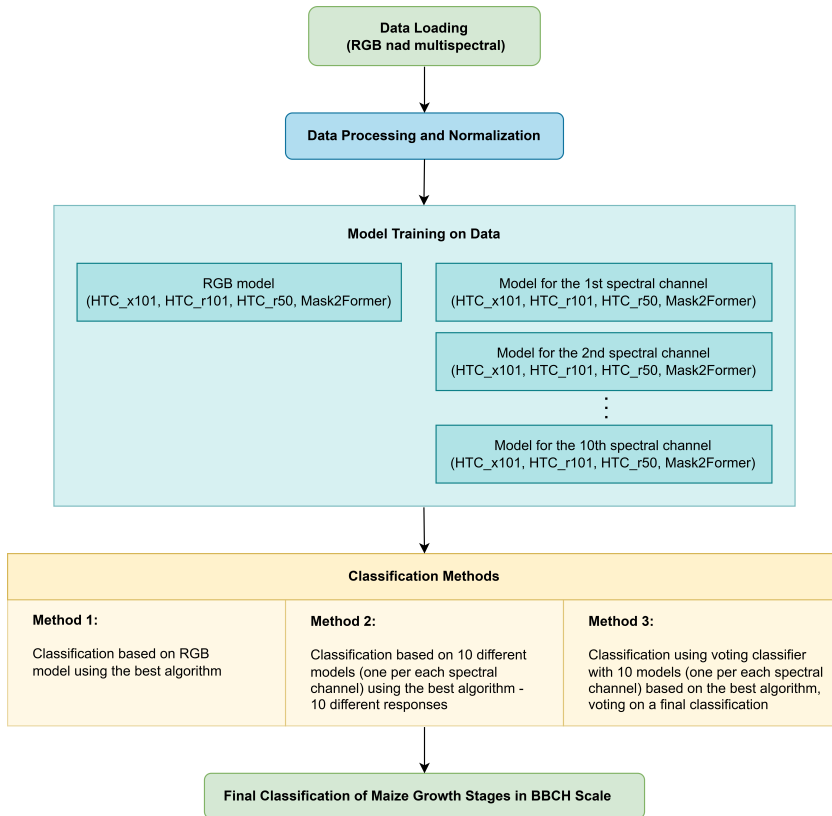


Fig. 4. Scheme of the proposed maize development stage classification system.

propagation and training [15]. HTC_r50, on the other hand, uses the ResNet-50 backbone, incorporating 50 layers with a residual mechanism that strikes a balance between performance and computational efficiency [18]. All three configurations use the Feature Pyramid Network (FPN) as the ‘neck’, which generates feature maps at different scales and enables efficient detection of objects across various sizes [18].

Regarding the ‘heads’, each model features similar components. The RPN (Region Proposal Network) head generates Region of Interest (ROI) proposals for further processing [23]. The ROI head conducts multi-stage bounding box regression and object classification to enhance detection accuracy [13]. The Mask head is responsible for instance segmentation, accurately delineating object contours within an image [14]. Finally, the Semantic head incorporates contextual information to improve performance in semantic segmentation tasks [35].

The HTC_x101, HTC_r101, and HTC_r50 models are trained using Stochastic Gradient Descent (SGD) optimization, with key hyperparameters including learning rate, momentum, and weight decay [6]. They incorporate advanced techniques such as RoIAlign (Region of Interest Align), which enhances the accuracy of object detection and segmentation. The use of multi-stage bounding box regression and semantic segmentation contributes to their high performance in benchmarks like COCO [14].

HTC's various configurations offer the flexibility to choose the right architecture depending on computational and precision requirements, making them versatile tools for advanced computer vision applications.

Mask2Former

Mask2Former is an advanced model architecture designed for various image segmentation tasks, including instance, semantic, and panoptic segmentation. It incorporates several innovations that enhance its performance over previous models.

The architecture employs a transformer-based backbone, utilizing a self-attention mechanism to efficiently process visual data and capture global dependencies within images, which is crucial for accurate segmentation [27]. Unlike traditional methods that generate masks for predefined regions, Mask2Former features dynamic mask prediction. This approach generates masks based on the context of each image, significantly increasing the model's flexibility and precision in mask generation [37].

Additionally, Mask2Former integrates instance, semantic, and panoptic segmentation tasks into a unified framework, allowing it to perform multiple types of segmentation without requiring structural modifications. The model also utilizes a query-based learning mechanism, where queries are dynamically updated during training to adapt to various scenarios, enhancing the quality of the generated masks [37].

Furthermore, Mask2Former employs advanced loss functions, such as focal loss, to effectively address issues with class imbalance in the training data, thereby improving overall performance and training efficiency [19].

Mask2Former is trained with finely tuned hyperparameters such as learning rate, weight decay, and the use of regularization techniques such as dropout and data augmentation [25]. The model also uses self-attention and query mechanisms for dynamic mask learning, which enhances its ability to accurately segment [37].

The architecture achieves high performance in benchmarks such as COCO, ADE20K, and Cityscapes, demonstrating superiority over previous segmentation methods [11, 36]. Its versatility and innovative approach to mask prediction make it a powerful tool in the field of image segmentation, for both research and practical applications [9, 37]. Mask2Former represents a significant step forward in segmentation model architectures, combining advanced visual data processing techniques with an efficient approach to dynamic mask prediction [34, 37].

Tab. 2. Hyperparameters and Configuration for HTC_r50, HTC_r101, HTC_x101, and Mask2Former Models

Model	Backbone	LR	Momentum	Weight Decay	Batch Size	Loss Functions
HTC_r50	ResNet-50	0.0003	0.9	0.0001	1	CrossEntropyLoss, SmoothL1Loss
HTC_r101	ResNet-101	0.0003	0.9	0.0001	1	CrossEntropyLoss, SmoothL1Loss
HTC_x101	ResNeXt-101	0.0003	0.9	0.0001	1	CrossEntropyLoss, SmoothL1Loss
Mask2Former	ResNet-50	0.0003	0.9	0.0001	1	CrossEntropyLoss, DiceLoss

3.2.2. Implementation, training and evaluation procedures

In our research, we used the PyTorch library to implement various models. We defined model architectures, specifying backbones, the Feature Pyramid Network (FPN) as the neck, and heads such as RPN, ROI, Mask, and Semantic Heads. The detailed configuration, including hyperparameters, backbones, and loss functions, is presented in Table 2.

To initialize these models, we employed weights pre-trained on the ImageNet dataset, leveraging knowledge embedded in large-scale datasets to enhance performance on our smaller labeled datasets.

To further improve robustness and generalization, we applied data augmentation techniques during training. These included resizing images to 1333x1000 pixels while maintaining their aspect ratio, random horizontal flipping with a probability of 50%, normalization using mean and standard deviation values for RGB channels, and padding to ensure image dimensions were divisible by 32. For segmentation tasks, masks were downsampled by a factor of 0.125. During testing, multi-scale augmentation was applied with resizing to 1333x1000 pixels, and flipping was disabled to maintain consistency in evaluation.

Models were trained using Stochastic Gradient Descent (SGD) to minimize the loss function. We used CrossEntropyLoss and SmoothL1Loss for most models, while for Mask2Former, we employed CrossEntropyLoss and DiceLoss.

Model performance was evaluated using the following measures: mAP (Mean Average Precision), accuracy, precision, sensitivity (recall), and IoU (Intersection over Union).

3.2.3. Description of the algorithm voting process

For the multispectral images, we noted that objects appeared at varying distances from the image edges due to different lens angles. This issue was particularly pronounced for maize at higher developmental stages (BBCH 14–19), where variations in angles altered object shapes, making it difficult to create composite images from different spectral

channels. While one lens might capture a leaf as relatively straight, another could record it as curved. These discrepancies complicated the superimposition of images from different spectral channels into a single composite image.

- **image_1:** *class_1* – [polygon_1: area, polygon_2: area], *class_2* – [polygon_1: area]
- **image_2:** *class_3* – [polygon_1: area]

The final results obtained take the following form:

- image_1: class_6
- image_2: class_5

To address this issue and fully utilize all spectral channels of the RedEdge-MX camera, we employed a voting classifier composed of 10 individual models. Each model was trained on images from a specific spectral channel using the HTC algorithm with a ResNeXt101 backbone, as this configuration consistently provided the best performance. After training, the predictions from these models were aggregated by plant identifier (file_id). For each plant, the final class was determined by majority voting. In cases where there was a tie, the class with the highest average polygon score was selected.

The aggregated predictions yield results in the following format:

- **image_1:** *class_1* – [polygon_1: score, polygon_2: score], *class_2* – [polygon_1: score]
- **image_2:** *class_3* – [polygon_1: score]

The voting classifier aggregates these results and assigns the class with the highest score to an image. The final results are presented as:

- image_1: class_6
- image_2: class_5

During validation, the markings are aggregated into classes using defined polygons for each spectral channel. An image may contain multiple polygons for various classes. To determine a single class per image, we selected the polygon with the largest area. The initial data structure for each spectral channel is as follows:

The voting classifier predicts the class for each plant based on images taken from different angles by separate cameras. By aggregating predictions from all spectral channels, we can achieve more robust and accurate classifications, even when individual models produce inconsistent results due to variations in object appearance.

Figure 5 illustrates the workflow of the voting classifier, from training individual models on spectral channels to aggregating predictions and selecting the final class.

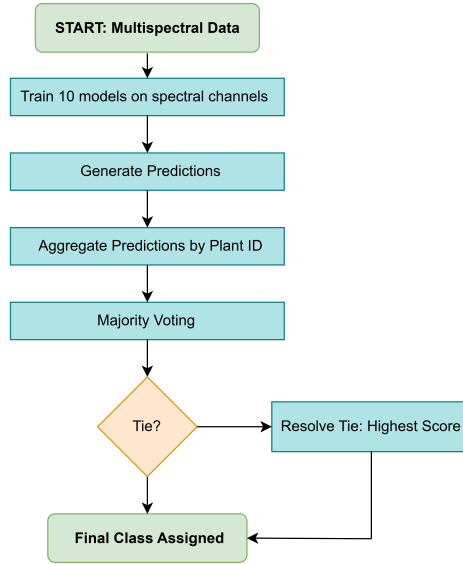


Fig. 5. Workflow of the voting classifier: training models on spectral channels, making predictions, and aggregating results through majority voting.

4. Experimental results

In this chapter, we present the experimental results for BBCH scale classification of developmental stages. We compare results obtained from RGB data, multispectral data, and the voting method developed using multispectral data.

4.1. Algorithms results on multispectral data

During the experimental study, we noticed differences in classification performance between the different algorithms used for the training process of deep neural networks and between the different spectral channels on which the algorithms were trained.

In Table 3 we present the classification results for the spectral channels recorded for each of the algorithms tested. The model trained on the data from channel 10 clearly differs in classification efficiency from the models trained on the other spectral channels. The table presented illustrates the effectiveness of each classification method on multispectral data.

From the analysis, we conclude that for the HTC algorithm with ResNet101, spectral channel 07 (red edge 705 (10)) yielded the best results (in the notation of channels in the optical specifications of cameras and multispectral sensors, in the description,

Tab. 3. Classification results for each of the spectral channels analysed using individual deep learning algorithms.

channel	accuracy				precision			
	HTC_r101	HTC_r50	HTC_x101	Mask2Former	HTC_r101	HTC_r50	HTC_x101	Mask2Former
01	0.486957	0.552174	0.547826	0.539130	0.486508	0.592528	0.582530	0.550819
02	0.568282	0.559471	0.559471	0.555066	0.594116	0.563848	0.586652	0.611719
03	0.548246	0.550661	0.530702	0.508772	0.587350	0.573304	0.570517	0.567900
04	0.567100	0.549784	0.541126	0.510823	0.603094	0.587433	0.563036	0.502541
05	0.575893	0.580357	0.558036	0.531250	0.596516	0.605068	0.578664	0.545954
06	0.547085	0.551570	0.581081	0.520179	0.570063	0.577274	0.591761	0.545100
07	0.582609	0.565217	0.569565	0.495652	0.606866	0.581716	0.593181	0.566532
08	0.538117	0.522321	0.540179	0.486607	0.575040	0.556559	0.572584	0.504101
09	0.551570	0.569507	0.549550	0.524664	0.586433	0.584708	0.561682	0.540334
10	0.219298	0.223684	0.214912	0.232456	0.161993	0.175143	0.162701	0.174677

channel	recall				F1-score			
	HTC_r101	HTC_r50	HTC_x101	Mask2Former	HTC_r101	HTC_r50	HTC_x101	Mask2Former
01	0.546268	0.608676	0.582942	0.547557	0.482994	0.558260	0.543616	0.506817
02	0.611719	0.619491	0.589024	0.599922	0.563584	0.554428	0.564948	0.543282
03	0.593651	0.591402	0.566799	0.549762	0.552418	0.557723	0.521853	0.509900
04	0.626295	0.600151	0.566362	0.557919	0.566393	0.555305	0.539768	0.494669
05	0.621161	0.630494	0.587758	0.571906	0.585859	0.589184	0.552485	0.523784
06	0.571536	0.585762	0.610729	0.558193	0.546741	0.556209	0.591596	0.521969
07	0.627316	0.587195	0.602076	0.541441	0.587952	0.560104	0.570215	0.474963
08	0.561341	0.553343	0.555533	0.498973	0.542855	0.529349	0.552871	0.476158
09	0.611539	0.596285	0.565288	0.541253	0.557820	0.570818	0.544688	0.527224
10	0.215904	0.228801	0.217906	0.257787	0.175971	0.187777	0.176843	0.185934

Tab. 4. Best classification results for each spectral channel.

Spectral channel	Wavelength [nm]	Best algorithm	Metrics
01 (coastal blue)	444 (28)	HTC (ResNet50)	accuracy, precision, recall, F1-score
02 (blue)	475 (32)	HTC (ResNet101)	accuracy, recall
		Mask2Former	precision
		HTC (ResNeXt101)	F1-score
03 (green)	531 (14)	HTC (ResNet50)	accuracy, F1-score
		HTC (ResNet101)	precision, recall
04 (green)	560 (27)	HTC (ResNet101)	accuracy, precision, recall, F1-score
05 (red)	650 (16)	HTC (ResNet50)	accuracy, precision, recall, F1-score
06 (red)	668 (14)	HTC (ResNeXt101)	accuracy, precision, recall, F1-score
07 (red edge)	705 (10)	HTC (ResNet101)	accuracy, precision, recall, F1-score
08 (red edge)	717 (12)	HTC (ResNet101)	precision, recall
		HTC (ResNeXt101)	accuracy, F1-score
		HTC (ResNet50)	accuracy, F1-score
09 (red edge)	740 (18)	HTC (ResNet101)	precision, recall
		Mask2Former	accuracy, recall
10 (NIR)	842 (57)	Mask2Former	accuracy, recall
		HTC (ResNet50)	precision, F1-score

e.g. 444 (28), the first value (444 nm) refers to the central wavelength, and the value in parentheses (28 nm) represents the bandwidth; so, in this example, the spectral range is $430 \pm (28/2)$ nm). With ResNet50, channel 05 (red 650 (16)) performed best, and with ResNeXt101, channel 06 (red 668 (14)) was optimal. For Mask2Former, channel 02 (blue 475 (32)) provided the best results. Different algorithms thus achieve optimal performance with different spectral channels. Table 4 summarizes the best-performing algorithm for each spectral channel.

The classification results for spectral channel 10 (NIR 842 (57)) show a significant

Tab. 5. Comparison of the classification results across different algorithms on RGB images with those obtained using multispectral images and voting classifier approach. The analysis uses accuracy, precision, recall, and F1-score measures.

Model	Approach	accuracy	precision	recall	F1-score
HTC_r101	voting classifier	0.651466	0.684678	0.686132	0.643477
	RGB	0.706667	0.460286	0.496550	0.465737
HTC_r50	voting classifier	0.661238	0.690290	0.699036	0.659793
	RGB	0.680000	0.424074	0.404996	0.406152
HTC_x101	voting classifier	0.657980	0.663664	0.676181	0.652508
	RGB	0.760000	0.525599	0.580694	0.524339
Mask2Former	voting classifier	0.625407	0.615539	0.630647	0.592380
	RGB	0.800000	0.596212	0.660516	0.600132

deviation in accuracy compared to other channels. This indicates that classification using only this spectral channel has the lowest object classification efficiency among the analyzed bands.

4.2. Algorithms results on RGB data

In addition to the multispectral studies, we conducted experiments with RGB images. We observed variations in classification performance among different algorithms when trained on RGB images.

The Mask2Former algorithm achieved the best RGB image classification results in terms of accuracy, precision, recall, and F1-score, making it the most effective model for RGB among those studied. However, the HTC algorithm with a ResNeXt101 backbone ranked second, followed by HTC with a ResNet101 backbone in third place and HTC with a ResNet50 backbone in fourth place.

4.3. Results obtained in the voting process

Table 5 summarizes the classification accuracy results for all tested algorithms, comparing RGB models with those using our voting method based on individual spectral channels. The evaluation measures used are accuracy, precision, recall, and F1-score.

The results show that the HTC algorithm with a ResNet101 backbone, trained on RGB data, achieves a higher accuracy measure than the voting classifier based on all spectral channels. However, for the precision, recall, and F1-score measures, the voting classifier achieves better results.

For the HTC algorithm with a ResNet50 backbone, the model trained on RGB data outperforms our voting method based on single spectral channel models in terms of accuracy. However, the voting method performs better in precision, recall and F1-score measures.

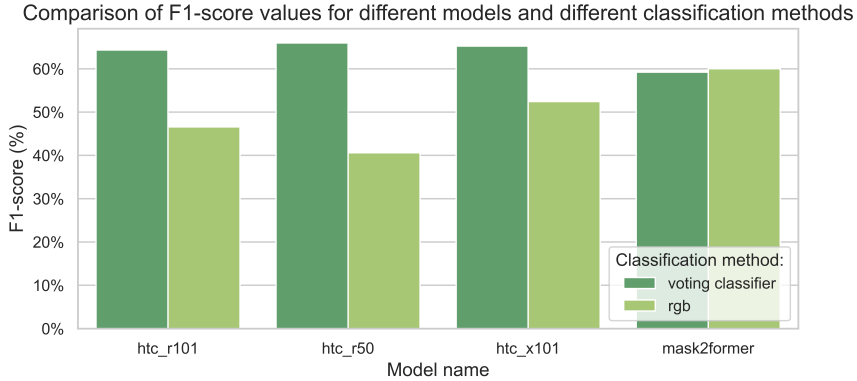


Fig. 6. Visualisation comparing the classification results between algorithms trained on RGB images and those using multispectral images with a voting classifier approach.

The HTC algorithm with a ResNeXt101 backbone, trained on RGB data, achieves a higher accuracy measure than the voting classifier based on all spectral channels. However, for the precision, recall, and F1-score measures, the voting classifier achieves better results.

For the Mask2Former algorithm, the model trained on RGB data outperforms our voting method based on single spectral channel models in terms of accuracy, recall, and F1-score. However, the voting method performs better in the precision measure.

In turn, in Figure 6, we present a graphical summary of the comparative data for the model trained on RGB images and the method using a voting classifier. We used the F1-score measure for comparison.

For HTC with ResNet101, ResNet50 and ResNeXt101, the voting classifier outperforms the RGB-trained model. In contrast, for the Mask2Former algorithm, the RGB-trained model outperforms the voting classifier.

Key conclusions include that the voting classifier based on single spectral channels performed better than the RGB classifier. Single-channel models generally show lower quality compared to the RGB classifier trained on three-channel images; however, the potential of voting techniques to enhance predictions improved the overall performance.

4.4. Learning curves

In this chapter, we present the learning curves recorded during the training of each model (the complete set of curves is available in the repository [26]). Below are the curves for each algorithm, illustrating training performance across different spectral channels.

Figure 7 compares the performance of the HTC_r101 model trained on individual

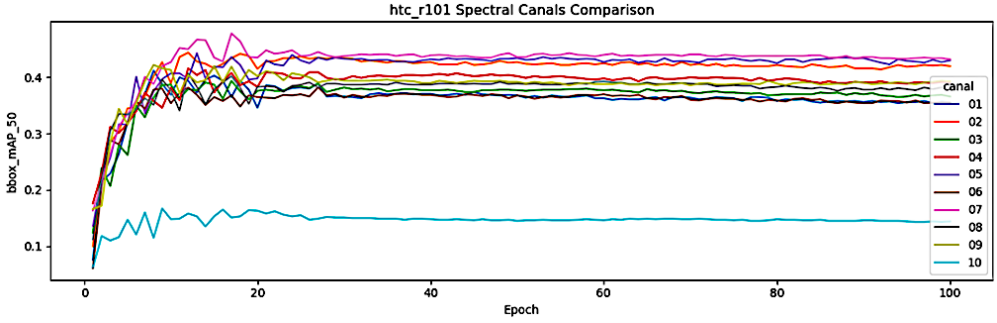


Fig. 7. Comparison of the training curves of the HTC_r101 model trained on individual spectral channels.

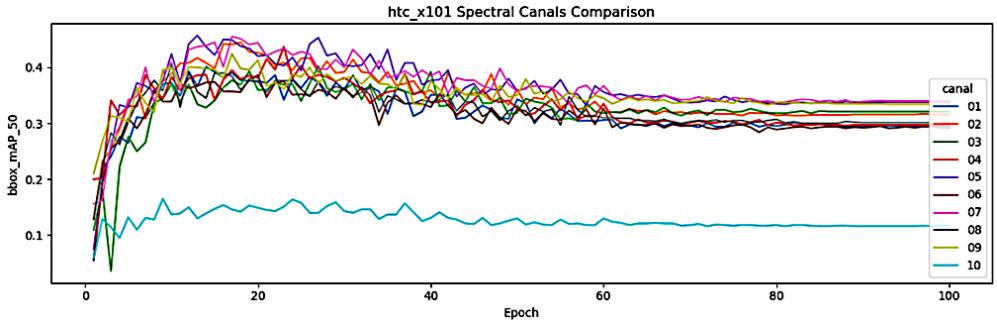


Fig. 8. Comparison of the training curves of the HTC_x101 model trained on individual spectral channels.

spectral channels over 100 epochs. The horizontal axis shows epochs, and the vertical axis displays object detection performance expressed by the measure `bbox_mAP_50`. This measure denotes the Bounding Box Mean Average Precision at IoU 50%. It considers detections as correct if the Intersection over Union (IoU) between the predicted and ground truth bounding boxes is at least 50%. Channel 10 exhibits the lowest performance, with `bbox_mAP_50` values around 0.1 after 20 epochs. In contrast, other channels show better results, with `bbox_mAP_50` values around 0.4 and minor fluctuations. Channel 7 achieves the highest efficiency, with a maximum `bbox_mAP_50` of about 0.47, maintaining the best performance among all individual spectral channels.

Figure 8 presents the learning curves for the HTC_x101 model across spectral channels. Channel 10 shows the lowest performance, with `bbox_mAP_50` around 0.4 initially, dropping to 0.35 between 40-60 epochs, and stabilizing above 0.3 afterward. Channels 05 and 07 perform best, with very similar results.

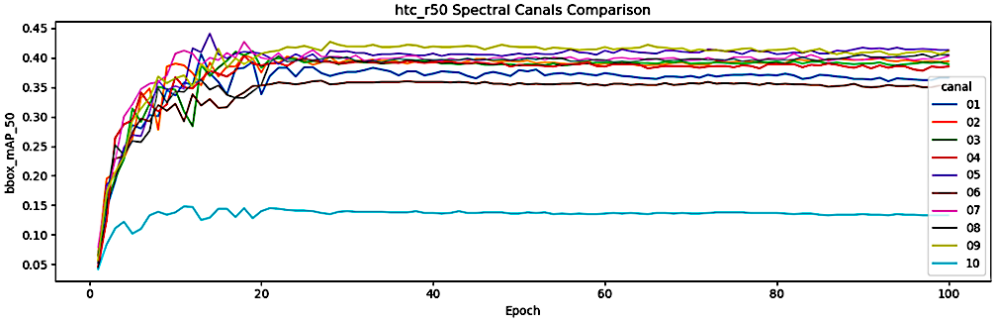


Fig. 9. Comparison of the training curves of the HTC_r50 model trained on individual spectral channels.

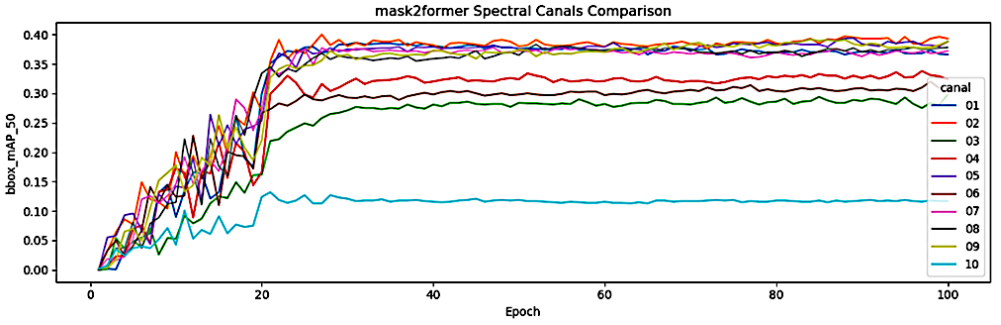


Fig. 10. Comparison of the training curves of the Mask2Former model trained on individual spectral channels.

For the HTC_r50 algorithm (Figure 9), the learning curves are similar to those of HTC_r101. Channel 10 shows the lowest performance, with `bbox_mAP_50` values around 0.4 after 10 epochs, remaining stable with slight fluctuations up to 100 epochs. The values for channels 01-09 are more stable after reaching a maximum, indicating their better performance in detecting objects with the HTC_r50 model.

For the Mask2Former algorithm (Figure 10), the lowest results can also be observed for channel 10, where `bbox_mAP_50` oscillates around 0.125 after the first 20 epochs and remains at this level until the end of the observation, i.e. the end of 100 epochs. The remaining channels reach higher `bbox_mAP_50` values, but are no longer such a compact group as in the previous algorithms. Of all the channels, the highest `bbox_mAP` values are reached by channel 02.

The results show that different algorithms reach peak `bbox_mAP_50` values for various spectral channels, with most channels stabilizing after 20 epochs (except HTC_x10,

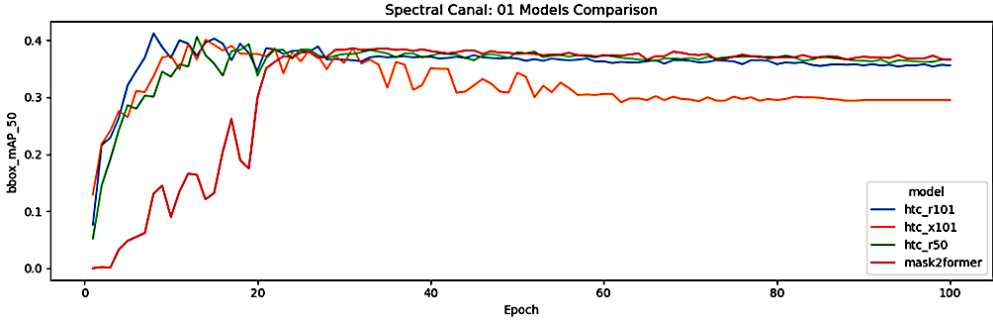


Fig. 11. Comparison of the training curves of different models on the dataset from the first spectral channel.

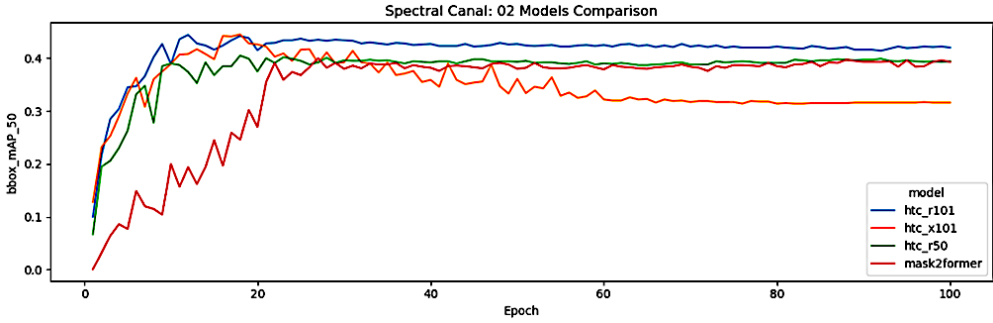


Fig. 12. Comparison of the training curves of different models on the dataset from the second spectral channel.

stabilizing after 60 epochs). Channel no. 10 consistently yields the poorest learning results.

We now examine models trained on individual spectral channels. Figures 11 and 12 compare the performance of four models on datasets from two spectral channels over 100 epochs. Additional graphs for other channels are available in the repository [26]. The horizontal axis represents epochs, and the vertical axis shows bbox_mAP_50, indicating detection accuracy.

Figure 11 shows results for spectral channel 01. The Mask2Former model achieves the highest and most stable bbox_mAP_50 of around 0.4 after 20 epochs. HTC_r101 and HTC_r50 models also stabilize around 0.4 but perform slightly worse. HTC_x101 starts strong but declines after 20 epochs, stabilizing around 0.3, indicating lower performance.

Analyzing spectral channel 02 (Figure 12), HTC_r101 and HTC_r50 rapidly increase

bbox_mAP_50 to around 0.4 and stabilize. Mask2Former also rises to about 0.4 within 20 epochs. HTC_x101 initially increases to 0.4 but then drops to around 0.3. HTC_r101 achieves the highest performance, while HTC_x101 shows the lowest one.

For spectral channel 03 (see the repository [26]), HTC_r101 and HTC_r50 rapidly increase in bbox_mAP_50 during the first 10 epochs, stabilizing around 0.4. Mask2Former rises swiftly in the first 20 epochs, then stabilizes at about 0.3. HTC_x101 initially increases to 0.4, but then declines and stabilizes around 0.3. The highest performance is achieved by the HTC_r50 model, while the Mask2Former model performs the worst.

For spectral channel 04 (see the repository [26]), HTC_r101 and HTC_r50 rapidly increase bbox_mAP_50 to around 0.4 and stabilize. Mask2Former rises to 0.3 within 20 epochs. HTC_x101 also reaches 0.4 initially but declines to around 0.3. HTC_r50 shows the highest performance, while Mask2Former performs the worst.

By analysing the results for spectral channel 05 (see the repository [26]) it can be discovered that HTC_r101 and HTC_r50 rapidly increase bbox_mAP_50 to around 0.4 and stabilize. Mask2Former also stabilizes at about 0.4 after 20 epochs. HTC_x101 rises to 0.4 initially but declines to around 0.3. HTC_r101 achieves the highest results, followed by HTC_r50, while HTC_x101 shows the lowest performance.

For spectral channel 06 (see the repository [26]), the HTC_r101 and HTC_r50 algorithms quickly increase bbox_mAP_50 to around 0.4 and then stabilize. Mask2Former rises swiftly in the first 20 epochs and stabilizes at about 0.3. HTC_x101 reaches 0.38 initially but declines to around 0.3. HTC_r101 achieves the highest results, followed by HTC_r50, with HTC_x101 performing the worst.

For spectral channel 07 (see the repository [26]), the HTC_r101 and HTC_r50 algorithms quickly increase bbox_mAP_50 to 0.45 and 0.4, respectively, and stabilize. Mask2Former rises rapidly in the first 20 epochs and stabilizes at around 0.4. HTC_x101 reaches 0.45 initially but declines to 0.3. HTC_r101 performs the best, followed by HTC_r50, with HTC_x101 showing the lowest results.

For spectral channel 08 (see the repository [26]), the HTC_r101 and HTC_r50 algorithms quickly increase bbox_mAP_50 to around 0.4 and stabilize. Mask2Former also rises rapidly and stabilizes at about 0.4. HTC_x101 reaches 0.4 initially but declines to 0.3. HTC_r50 performs the best, while HTC_x101 shows the lowest results.

For spectral channel 09 (see the repository [26]), the HTC_r101 and HTC_r50 algorithms show a rapid rise in bbox_mAP_50 to around 0.4, stabilizing at this level. Mask2Former also reaches about 0.4 after 20 epochs. The HTC_x101 model initially increases to 0.4 but declines to 0.3. HTC_r50 achieves the highest performance, followed by HTC_r101 and Mask2Former, with HTC_x101 showing the lowest results.

For spectral channel 10 (see the repository [26]), the HTC_r101 and HTC_r50 algorithms quickly rise in bbox_mAP_50 to about 0.15, stabilizing there. The Mask2Former

also increases to around 0.125. The HTC_x101 model initially reaches 0.4 but declines to 0.125. HTC_r101 achieves the highest performance, followed by HTC_r50 and Mask2Former, with HTC_x101 showing the lowest results.

5. Conclusions

We have presented an innovative approach for automating maize growth monitoring using image analysis and artificial intelligence techniques. Our method employs deep neural networks to analyze RGB and multispectral images, along with an additional voting classifier. The goal was to efficiently detect and classify maize developmental stages based on the BBCH scale, enabling automatic monitoring of plant development phases and presenting the results on large scale, e.g., in the form of a map.

Our results demonstrate a highly automated method for detecting and classifying maize developmental stages with plant-level accuracy. Compared to manual methods, our solution significantly accelerates the classification process through real-time image analysis on a field robot, allowing for efficient maize growth stage monitoring. While UAV-based approaches cover larger field areas per image, our method offers greater precision at the individual plant level. By integrating advanced image analysis and deep learning algorithms, our solution achieves high automation and accuracy. A literature review confirms the novelty of our method.

The models in our study were trained on proprietary datasets of labeled maize images at various BBCH developmental stages (10–19), captured in both RGB and multispectral spectra. This allowed comprehensive training separately on RGB and each spectral channel. Furthermore, we evaluated various deep learning architectures to assess detection and classification performance across different training datasets and algorithms.

To improve the performance of the model we employed pre-trained backbones such as ResNeXt-101, ResNet-101, and ResNet-50, initialized with ImageNet weights. Fine-tuning these models on our labeled datasets leveraged the rich feature representations learned from large-scale datasets, improving accuracy and robustness. Additionally, we applied data augmentation techniques such as resizing, random horizontal flipping, normalization, padding, and mask downscaling, further enhancing model performance.

Single-channel models generally performed worse than RGB models due to their limited spectral information. However, the voting classifier improved prediction quality.

Comparing the results of different algorithms and training sets, we observed that HTC_r50, HTC_r101, and HTC_x101 achieved higher precision, recall, and F1-score when trained on single spectral channels with a voting classifier than on RGB data. For Mask2Former, precision was slightly higher with the voting classifier, while accuracy, recall, and F1-score were better for RGB data.

For RGB images, the best overall performance across all measures was achieved by Mask2Former, followed by HTC_x101, HTC_r101, and HTC_r50. For single spectral

channels with the voting classifier, HTC_r50 achieved the highest accuracy and F1-score, followed by HTC_x101, HTC_r101, and Mask2Former. Regarding precision and recall, HTC_r50 performed best, followed by HTC_r101, HTC_x101, and Mask2Former.

According to the accuracy measure, the best performance was achieved by the model Mask2Former trained on RGB data, while in terms of precision, recall, and F1-score, HTC_r50 trained on individual spectral channels with a voting classifier performed best.

Another key finding was the identification of optimal spectral channels for maize growth stage classification. For HTC_r101, channel 07 yielded the best results. For HTC_x101, channels 05 and 07 were optimal. HTC_r50 performed best on channel 05, while Mask2Former achieved the best results on channel 02.

Our solution enables precise plant condition tracking, supporting decision-making in precision agriculture. Moreover, our method can be adapted to other crops by developing appropriate datasets and retraining deep neural networks.

6. Discussion of limitations

A multispectral camera with multiple lenses captures spectral channels from different angles, causing variability in plant shapes across the channels. To mitigate this, a single-lens system should be used. We explored this approach in our other research.

The models were trained on proprietary datasets with labeled maize images at various growth stages. The limited diversity of the dataset may affect the models' ability to generalize to real-world field conditions. Expanding the dataset with images from different locations and conditions can improve model performance and robustness.

Our solution was developed specifically for monitoring the developmental stages of maize. Adapting the method to other crops requires creating new datasets and retraining the models. However, validating the effectiveness of the developed solution for maize offers promising prospects for successful application to other crops as well.

Acknowledgement

This research was supported by the FITOEXPORT project (Contract No. GOSPOSTRATEG 1/385957/5/NCBR/2018), funded by the National Centre for Research and Development (NCBR). The authors express their gratitude for the support provided.

References

- [1] S. Bansal, M. Singh, S. Barda, N. Goel, and M. Saini. PA-RDFKNet: Unifying Plant Age Estimation through RGB-Depth Fusion and Knowledge Distillation. *IEEE Transactions on AgriFood Electronics*, 2(2):226–235, 2024. doi:[10.1109/TAFE.2024.3418818](https://doi.org/10.1109/TAFE.2024.3418818).
- [2] A. Bera, D. Bhattacharjee, and O. Krejcar. An attention-based deep network for plant disease classification. *Machine Graphics and Vision*, 33(1):47–67, 2024. doi:[10.22630/MGV.2024.33.1.3](https://doi.org/10.22630/MGV.2024.33.1.3).

- [3] A. Bera, D. Bhattacharjee, and O. Krejcar. PND-Net: Plant Nutrition Deficiency and Disease Classification Using Graph Convolutional Network. *Scientific Reports*, 14(1):15537, 2024. doi:[10.1038/s41598-024-66543-7](https://doi.org/10.1038/s41598-024-66543-7).
- [4] A. Bera, O. Krejcar, and D. Bhattacharjee. RAFA-Net: Region Attention Network for Food Items and Agricultural Stress Recognition. *IEEE Transactions on AgriFood Electronics*, pp. 1–13, 2024. doi:[10.1109/TAFE.2024.3466561](https://doi.org/10.1109/TAFE.2024.3466561).
- [5] L. Bompani et al. Accelerating Image-based Pest Detection on a Heterogeneous Multicore Microcontroller. *IEEE Transactions on AgriFood Electronics*, 2(2):170–180, 2024. doi:[10.1109/TAFE.2024.3451888](https://doi.org/10.1109/TAFE.2024.3451888).
- [6] L. Bottou. Stochastic Gradient Descent Tricks. In: G. Montavon, G. B. Orr, and K.-R. Müller, eds., *Neural Networks: Tricks of the Trade*, vol. 7700 of *Lecture Notes in Computer Science*, pp. 421–436. Springer, second edition edn., 2012. doi:[10.1007/978-3-642-35289-8_25](https://doi.org/10.1007/978-3-642-35289-8_25).
- [7] K. Chen, J. Pang, J. Wang, Y. Xiong, X. Li, et al. Hybrid Task Cascade for Instance Segmentation. In: *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 4974–4983. Long Beach, CA, USA, June 16-20 2019. doi:[10.1109/CVPR.2019.00511](https://doi.org/10.1109/CVPR.2019.00511).
- [8] K. Chen, J. Wang, J. Pang, Y. Cao, Y. Xiong, et al. MMDetection: Open MMLab Detection Toolbox and Benchmark. *arXiv*, 2019. ArXiv:1906.07155. doi:[10.48550/arXiv.1906.07155](https://doi.org/10.48550/arXiv.1906.07155).
- [9] L.-C. Chen, Y. Zhu, G. Papandreou, F. Schroff, and H. Adam. Encoder-Decoder with Atrous Separable Convolution for Semantic Image Segmentation. In: *Proc. European Conference on Computer Vision (ECCV)*, pp. 801–818. Springer, Munich, Germany, September 8-14 2018. doi:[10.1007/978-3-030-01234-2_49](https://doi.org/10.1007/978-3-030-01234-2_49).
- [10] B. Cheng, A. Schwing, and A. Kirillov. Masked-attention Mask Transformer for Universal Image Segmentation. In: *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1280–1289. New Orleans, LA, USA, 2022. doi:[10.1109/CVPR52688.2022.01280](https://doi.org/10.1109/CVPR52688.2022.01280).
- [11] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, et al. The Cityscapes Dataset for Semantic Urban Scene Understanding. In: *Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3213–3223. IEEE, Las Vegas, NV, USA, June 27-30 2016. doi:[10.1109/CVPR.2016.350](https://doi.org/10.1109/CVPR.2016.350).
- [12] S. Figiel. Development of Artificial Intelligence and Potential Impact of Its Applications in Agriculture on Labor Use and Productivity. *Zagadnienia Ekonomiki Rolnej / Problems of Agricultural Economics*, 373(4):5–21, 2022. doi:[10.30858/zer/153583](https://doi.org/10.30858/zer/153583).
- [13] R. Girshick. Fast R-CNN. In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 1440–1448. Santiago, Chile, December 7-13 2015. doi:[10.1109/ICCV.2015.169](https://doi.org/10.1109/ICCV.2015.169).
- [14] K. He, G. Gkioxari, P. Dollár, and R. Girshick. Mask R-CNN. In: *IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988. Venice, Italy, October 22-29 2017. doi:[10.1109/ICCV.2017.322](https://doi.org/10.1109/ICCV.2017.322).
- [15] K. He, X. Zhang, S. Ren, and J. Sun. Deep Residual Learning for Image Recognition. In: *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 770–778. IEEE, Las Vegas, NV, USA, 2016. doi:[10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90).
- [16] A. Kamilaris and F. X. Prenafeta-Boldú. Deep Learning in Agriculture: A Survey. *Computers and Electronics in Agriculture*, 147:70–90, 2018. doi:[10.1016/j.compag.2018.02.016](https://doi.org/10.1016/j.compag.2018.02.016).
- [17] P. D. Lancashire, H. Bleiholder, T. van den Boom, P. Langelüddecke, R. Stauss, et al. A Uniform Decimal Code for Growth Stages of Crops and Weeds: BBCH Monograph. *Annals of Applied Biology*, 119(3):561–601, 1991. doi:[10.1111/j.1744-7348.1991.tb04895.x](https://doi.org/10.1111/j.1744-7348.1991.tb04895.x).

- [18] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, et al. Feature Pyramid Networks for Object Detection. In: *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 936–944. Honolulu, HI, USA, July 21–26 2017. doi:[10.1109/CVPR.2017.106](https://doi.org/10.1109/CVPR.2017.106).
- [19] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár. Focal Loss for Dense Object Detection. In: *Proc. 2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 2980–2988. IEEE, Venice, Italy, October 22–29 2017. doi:[10.1109/ICCV.2017.324](https://doi.org/10.1109/ICCV.2017.324).
- [20] M. Liu, W.-H. Su, and X.-Q. Wang. Quantitative Evaluation of Maize Emergence Using UAV Imagery and Deep Learning. *Remote Sensing*, 15(8):1979, 2023. doi:[10.3390/rs15081979](https://doi.org/10.3390/rs15081979).
- [21] U. Meier, H. Bleiholder, L. Buhr, C. Feller, H. Hack, et al. The BBCH system for coding the phenological growth stages of plants – history and publications. *Journal für Kulturpflanzen*, 61(2):41–52, 2009. doi:[10.5073/JfK.2009.02.01](https://doi.org/10.5073/JfK.2009.02.01).
- [22] E. F. I. Raj, M. Appadurai, and K. Athiappan. Precision farming in modern agriculture. In: *Smart Agriculture Automation Using Advanced Technologies: Data Analytics and Machine Learning, Cloud Architecture, Automation and IoT*, pp. 61–87. Springer Singapore, Singapore, 2022. doi:[10.1007/978-981-16-6124-2_4](https://doi.org/10.1007/978-981-16-6124-2_4).
- [23] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. In: *Advances in Neural Information Processing Systems (NeurIPS)*, pp. 91–99. Montreal, Canada, December 7–12 2015. doi:[10.1109/TPAMI.2016.2577031](https://doi.org/10.1109/TPAMI.2016.2577031).
- [24] T. A. Shaikh, T. Rasool, and F. R. Lone. Towards Leveraging the Role of Machine Learning and Artificial Intelligence in Precision Agriculture and Smart Farming. *Computers and Electronics in Agriculture*, 198:107119, 2022. doi:[10.1016/j.compag.2022.107119](https://doi.org/10.1016/j.compag.2022.107119).
- [25] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research*, 15(56):1929–1958, June 2014. doi:[10.5555/2627435.2670313](https://doi.org/10.5555/2627435.2670313).
- [26] J. Stypułkowska. bbch-maize-learning-curves. <https://github.com/JustynaStypulkowska/bbch-maize-learning-curves>, 2024. GitHub repository.
- [27] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, et al. Attention Is All You Need. In: *Advances in Neural Information Processing Systems 30 (NIPS 2017)*, pp. 6000–6010. Curran Associates, Inc., Long Beach, CA, USA, December 4–9 2017. doi:[10.5555/3295222.3295349](https://doi.org/10.5555/3295222.3295349).
- [28] J. Wu, V. Abolghasemi, M. H. Anisi, U. Dar, A. Ivanov, et al. Strawberry Disease Detection Through an Advanced Squeeze-and-Excitation Deep Learning Model. *IEEE Transactions on Agri-Food Electronics*, 2(2):259–267, 2024. doi:[10.1109/TAFE.2024.3412285](https://doi.org/10.1109/TAFE.2024.3412285).
- [29] S. Xie, R. Girshick, P. Dollár, Z. Tu, and K. He. Aggregated Residual Transformations for Deep Neural Networks. In: *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5987–5995. IEEE, Honolulu, HI, USA, July 21–26 2017. doi:[10.1109/CVPR.2017.634](https://doi.org/10.1109/CVPR.2017.634).
- [30] X. Xu, L. Wang, M. Shu, X. Liang, A. Ghafoor, et al. Detection and Counting of Maize Leaves Based on Two-Stage Deep Learning with UAV-Based RGB Image. *Remote Sensing*, 14(21):5388, 2022. doi:[10.3390/rs14215388](https://doi.org/10.3390/rs14215388).
- [31] Y. Yao, J. Yue, Y. Liu, H. Yang, H. Feng, et al. Classification of Maize Growth Stages Based on Phenotypic Traits and UAV Remote Sensing. *Agriculture*, 14(7):1175, 2024. doi:[10.3390/agriculture14071175](https://doi.org/10.3390/agriculture14071175).
- [32] D. Yu, Y. Zha, Z. Sun, et al. Deep Convolutional Neural Networks for Estimating Maize Above-Ground Biomass Using Multi-Source UAV Images: A Comparison with Traditional Machine Learning Algorithms. *Precision Agriculture*, 24(1):92–113, 2023. doi:[10.1007/s11119-022-09932-0](https://doi.org/10.1007/s11119-022-09932-0).

- [33] X. Zan, X. Zhang, Z. Xing, W. Liu, X. Zhang, et al. Automatic Detection of Maize Tassels from UAV Images by Combining Random Forest Classifier and VGG16. *Remote Sensing*, 12(18):3049, 2020. doi:[10.3390/rs12183049](https://doi.org/10.3390/rs12183049).
- [34] H. Zhang, Y. Liu, K. Ma, H. Su, S. Li, et al. Transformers for Image Segmentation. In: *Proc. 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1251–1260. IEEE, Long Beach, CA, USA, June 16-20 2019. doi:[10.1109/CVPR.2019.00130](https://doi.org/10.1109/CVPR.2019.00130).
- [35] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid Scene Parsing Network. In: *Proc. IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2881–2890. Honolulu, HI, USA, July 21-26 2017. doi:[10.1109/CVPR.2017.660](https://doi.org/10.1109/CVPR.2017.660).
- [36] B. Zhou, H. Zhao, X. Puig, S. Fidler, A. Barriuso, et al. Scene Parsing through ADE20K Dataset. In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5122–5130. IEEE, Honolulu, HI, USA, July 21-26 2017. doi:[10.1109/CVPR.2017.544](https://doi.org/10.1109/CVPR.2017.544).
- [37] X. Zhu, Z. Zhang, Z. Li, X. Wang, J. Sun, et al. Mask2Former: A Transformer Architecture for Universal Image Segmentation. In: *Proc. IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7823–7833. IEEE, New Orleans, LA, USA, June 19-24 2022. doi:[10.1109/CVPR52688.2022.00767](https://doi.org/10.1109/CVPR52688.2022.00767).

SELECTING UPDATE BLOCKS OF CONVOLUTIONAL NEURAL NETWORKS USING GENETIC ALGORITHM IN TRANSFER LEARNING

Md. Mehedi Hasan¹, Muhammad Ibrahim^{2*} , Md. Sawkat Ali¹

¹Department of Computer Science and Engineering, East West University, Dhaka, Bangladesh

²Department of Computer Science and Engineering, University of Dhaka, Dhaka, Bangladesh

*Corresponding author: Muhammad Ibrahim ibrahim313@du.ac.bd

Abstract The performance of convolutional neural networks (CNN) for computer vision problems depends heavily on their architectures. Transfer learning performance of a CNN strongly relies on selection of its trainable layers. Selecting the most effective update layers for a certain target dataset often requires expert knowledge on CNN architecture which many practitioners do not possess. General users prefer to use an available architecture (e.g. GoogleNet, ResNet, EfficientNet etc.) that is developed by domain experts. With the ever-growing number of layers, it is increasingly becoming difficult and cumbersome to handpick the update layers. Therefore, in this paper we explore the application of a genetic algorithm to mitigate this problem. The convolutional layers of popular pre-trained networks are often grouped into modules that constitute their building blocks. We devise a genetic algorithm to select blocks of layers for updating the parameters. By experimenting with EfficientNetB0 pre-trained on ImageNet and using three popular image datasets – namely Food-101, CIFAR-100 and MangoLeafBD – as target datasets, we show that our algorithm yields similar or better results than the baseline in terms of accuracy, and requires lower training and evaluation time due to learning a smaller number of parameters. We also devise a measure called block importance to measure each block's efficacy as an update block and analyze the importance of the blocks selected by our algorithm.

Keywords: computer vision; transfer learning; convolutional neural network; EfficientNet; genetic algorithm.

1. Introduction

High-performance image recognition models are often developed using Convolutional Neural Networks (CNN). As the prevalent approach of deep learning in image classification, CNNs have shown exceptional supremacy over many approaches in various real-world machine-learning applications, especially in the broad area of computer vision [18]. The performance of a CNN depend heavily on its architecture [28], [27], and hence, all of the state-of-the-art CNNs, such as GoogleNet [30], ResNet [13], DenseNet [14] etc., are handcrafted by experts who have rich domain knowledge. As it is not always feasible for general practitioners of CNN to acquire such expertise, these users often opt to use a pre-designed architecture that suits their need. CNNs are usually designed with a fixed computational resource budget, and then scaled up at a later time for better accuracy as more computational resources become available.

The training process of CNNs requires large sized datasets because these models need to learn a huge number of parameters. Since the parameter space is colossal, sufficient

quantity of training data are warranted to learn complex patterns. This requirement of having large datasets, however, can be relaxed using the transfer learning setting [25] where practitioners reuse existing pre-trained networks, thereby reducing the training time significantly. In transfer learning, during training phase, the parameters of some layers of the pre-trained network are kept fixed while updating the rest with the (target) dataset at hand. Generally, early layers, i.e., the layers near the input, of a CNN detect low-dimensional information like the color and edges of an image, and the later layers, i.e., the layers near the output, extract high-dimensional features that help to identify the ground truth labels [37]. Therefore, for transfer learning, usually the early layers of the pre-trained model are kept frozen while the parameters of the later layers are updated.

Some recent empirical studies, however, report good results by applying the opposite practice, i.e., keeping the parameters of the later layers fixed instead of that of earlier ones. To mention a few such works: Zunair et al [38] use a VGG16 network [28] pre-trained with ImageNet for the prediction task of Bangla characters and report that the best accuracy is found when the first input layer and early fully-connected layers are selected as update layers. Gafoorian et al [10] apply transfer learning on MRI images and report that the best performance is achieved when parameters of only six input layers are updated. Therefore, it is intriguing to investigate into the appropriate layers to be updated for transfer learning, thereby triggering this research.

Nowadays, common CNN architectures contain a lot of layers. For example, VGG16, InceptionV3, and GoogleNet have 16, 94, and 22 layers, respectively. EfficientNetB0 and EfficientNetB7 [32] have 237 and 813 layers, respectively. When the general users want to use a pre-trained model, such a large number of layers makes it hard for them to manually select the appropriate layers to be updated. Manually selecting the layers to be updated involves a trial-and-error approach which is time consuming and tiresome.

To mitigate this difficulty, there are studies that apply the so-called metaheuristic optimization algorithms, such as genetic algorithm, to automatically select the effective layers to be updated (we discuss these works in detail in Section 2). However, we have not found any work that investigates the problem of selecting the appropriate blocks – not individual layers – to be updated in a transfer learning setting using genetic algorithms. Our investigation is dedicated to this endeavor.

In this paper, we develop a genetic algorithm-based [11,22] method to automatically select blocks of layers – instead of individual layers – to update the parameters. This technique is expected to significantly minimize the training time of CNNs while maintaining similar accuracy. We also adapt a recently proposed metric named OTDD [2] to calculate the importance of the blocks of layers. Using this metric we calculate the contribution of each block in identifying the features of the data. In all these investigations we use three target datasets, namely, Food-101, CIFAR-100, and MangoLeafBD. As for the CNN, we use EfficientNet [32] models pre-trained on ImageNet [8].

The rest of the paper is organized as follows. Section 2 discusses the relevant existing works. Section 3 presents the background knowledge required to understand the paper. Section 4 presents the framework and methodology of the investigation. Section 5 demonstrates the experimental results. Finally, Section 6 concludes the paper.

2. Related Work

Deep learning and transfer learning are well-known for their effectiveness in image classification task [23]. It is well-known that the structure and performance of a CNN rely heavily on its hyper-parameters. Hyper-parameters are often manually chosen by experts to obtain a model with expected performance. However, different datasets may require different model structure, and hence, choosing them by trial and error can be tedious. AszemiDomic [4] discuss the usefulness of different search and optimization methods to find a suitable set of hyper-parameters. The authors also develop a hybrid optimization method combining a genetic algorithm with local search that can be used to optimize CNN structure.

To improve CNN performance, some studies have explored the possibility of training multiple CNNs at the same time using different means to promote cooperation and specialization among them. Such sets of CNNs are called committees. Bochinski et al. [5] propose a way of defining the CNN structure in terms of its hyper-parameters and a framework to automatically find out the best set of hyper-parameters using an evolutionary optimization algorithm. Additionally, they extend their framework to optimize a CNN committee. The goal of this study is to establish a framework to optimize CNN committees for better performance.

Yanan and Bing et al. [29] are inspired by the success of ResNet and DenseNet and propose a genetic algorithm-based evolutionary approach of automatically designing CNNs using blocks from ResNet and DenseNet. The proposed algorithm is self-sufficiently automatic and does not expect any specific domain expertise from the user.

XieYuille et al. [35] venture to find a way to automatically build effective CNN structures for a given dataset. As the number of possible network structures increase exponentially with the number of layers in the network, the authors employ a genetic algorithm to navigate through the expanding search space. They propose a fixed length encoding strategy which represents a network architecture in the population and a genetic algorithm that operates on this population to produce better generations. Experiments with the CIFAR10 and ILSVRC2012 datasets show that their method produces CNNs with competitive or better recognition accuracy.

Lee et al. [19] propose a genetic algorithm that considers CNN structure and its hyper-parameters both in the optimization space and produces a CNN with optimal architecture and hyper-parameter values for the given dataset. The performance of the

algorithm is evaluated with 18F-Florbetaben Amyloid PET/CT images for classification of Alzheimer's disease.

Loussaie et al. [20] observe that hyper-parameters of a network such as network depth, number of filters, and their sizes dramatically affect the performance of a CNN. They propose a genetic algorithm that finds the optimal values of those parameters for a given dataset, and thus produces an optimal CNN architecture.

Tian and Chen [33] develop a new genetic algorithm to find out the best suited pre-trained model for different datasets. They come up with a new genetic encoding model that represents different pre-trained CNN models in the population and an evolutionary approach to promote the best performing models in each generation. Experimental results have evidently shown that their approach outperforms some of the existing classification methods.

Cai and Luo [7] propose neural architecture search which is a resource-heavy search mechanism that automatically searches for CNN architectures. The authors devise an evolutionary framework that employs a multi-task, multi-objective search approach to find optimally balanced CNN architecture for a given task.

Finally, as discussed earlier, Nagae et al. [24] propose a method to automatically select effective layers using a genetic algorithm for InceptionV3 network. The authors also coin a term called Optimal Transport Dataset Distance (OTDD) to quantitatively evaluate a particular layer's efficacy as an update layer. They use OTDD to estimate a layer's importance as an update layer and compare a layer's contribution to accuracy with its OTDD score to reveal that layer OTDD score is indicative of a layer's capability of detecting features from the target dataset.

From the above discussion we see that although there are several works that utilize genetic algorithm to select the best setting of hyper-parameters and network architectures, to the best of our knowledge, there is no existing work that employs a genetic algorithm to select the best blocks to be updated in a transfer learning setting. Our investigation presented in this paper has filled this gap in the literature.

3. Background Study

As mentioned earlier, in this investigation we use EfficientNet [32] models pre-trained on ImageNet [8], though our developed framework is equally applicable to other types of CNNs. In this section we briefly discuss the architecture of EfficientNet and PathNet. We also discuss a metric called Optimal Transport Dataset Distance (OTDD).

3.1. EfficientNet

CNNs are often scaled up to achieve better performance. For example, ResNet [13] can be scaled up from ResNet-18 to ResNet-200 by incorporating more layers. GPipe's

ImageNet [15] top-1 accuracy is improved to 84.3% by scaling up the baseline model by 4 times. Scaling CNNs up by their depth [13] or width [36] are the most commonly performed by the practitioners, but scaling up the models by image resolution [15] is also a popular method. Only one of the three dimensions – depth, width, and image size – is usually scaled. Although two or three dimensions can be arbitrarily scaled, it involves tedious manual tuning, and yet often fails to provide better accuracy and efficiency. To resolve this, EfficientNet [32] proposes a simple yet effective compound scaling method that uses a constant scaling ratio to scale all three dimensions of network in a controlled and balanced way. The intuition is: as the input image size increases, it makes sense that the model will need more layers and more channels to extract even finer details from larger images. In fact, previous studies [21, 26, 36] have shown that there is a certain correlation between network width and depth. The compound scaling method uniformly scales network width, depth, and resolution with a set of constant scaling coefficients. In particular, if 2^N times more computational resources become available, then this method simply scales up the network depth by α^N , width by β^N , and image size by γ^N , where α, β, γ are constant coefficients chosen through a small grid search on the baseline network.

3.2. PathNet and StepwisePathNet

When performing transfer learning to the target datasets, each layer of a pre-trained CNN detects features that are common among the source and target datasets [37]. Therefore, it is imperative to efficiently select layers that are effective feature detectors for the target datasets and then update their parameters while keeping the other layers frozen. Additionally, as the network architectures have become more complex due to the increased availability of computational resources, an efficient way of selecting effective layers without manual labor is required. The StepwisePathNet method [16] tries to address this specific need. This algorithm expands DeepMind’s PathNet [9] to select the update layers in a straight-chain network. StepwisePathNet algorithm labels each layer as either fixed or updated, and the selection is optimized by a genetic algorithm that employs a tournament selection mechanism.

3.3. Improvement on StepwisePathNet

Citing the limitations of StepwisePathNet, Nagae et al [24] improves the algorithm by applying another genetic algorithm. The authors work with InceptionV3 architecture [31] and apply a genetic algorithm to automatically select the effective layers of the network to be updated during learning for the target dataset.

3.4. Optimal Transport Dataset Distance (OTDD)

Evaluation of the distance between two labeled datasets has been explored in studies utilizing the optimal transport distance [2], which provides a way to quantify the difference between two datasets and correlate dataset distance with transfer learning efficacy. Optimal transport deals with the issue of transferring material from one place to another at minimum cost, and can also be used to mathematically compare two different probability distributions [34].

Optimal Transport Dataset Distance (OTDD) [2] is a measure to compute the distance between two different labeled datasets. Using this metric, in [24], the importance of a layer is calculated by estimating its effectiveness as an update layer for transfer learning and its potential for detecting common traits in both source and target datasets. A subset of feature maps generated at layer l for both source and target datasets are taken and denoted respectively as A_l^{source} and A_l^{target} . Then, the layer importance LI is expressed as:

$$\text{LI}(l) = \frac{\text{OTDD}(A_l^{\text{source}}, A_l^{\text{target}})}{\text{OTDD}(A_l^{\text{source}}, A_l^{\text{source}'} + \epsilon)}, \quad (1)$$

where OTDD is calculated using Eq. (16) of [24] and ϵ is a small nonzero positive number. The denominator in Eq. (1) denotes the optimal transport distance between two different subsets, source and source', of the source dataset, where the difference is created due to the difference in sampling. This ratio basically captures the difference between the features maps generated for the source and target datasets at a particular layer. Datasets with similar feature sets should result in similar feature maps at an effective layer, yielding a lower LI. A lower value of LI indicates higher adaptability of the model for the target dataset.

4. Proposed Framework and Methodology

It is well-known to the research community that the performance of CNNs is highly dependent on their architecture, and hence all high performing CNNs like GoogleNet, ResNet, DenseNet, EfficientNet, InceptionV3 etc. have been manually designed by experts who possess profound knowledge on CNNs. Unfortunately, such deeper understanding of CNNs and expertise in machine learning cannot be expected from all general practitioners of these models. Hence, general users often opt to use a pre-designed architecture that suits their need, thereby giving rise to the notion of transfer learning. In this setting, the user of a CNN does not need to train the model on a large dataset, and instead takes advantage of a pre-trained model which has already been trained on a large source dataset. The intuition behind this strategy is as follows. Some layers of CNNs extract low level information such as edge, color, shape etc. from input images, while other layers detect high level features like ground truth labels of the instances.

Since in image classification task, the edges and low-level shapes of images are needed to be extracted irrespective of the domain at hand, there is no benefit to re-extract these features, and so the parameters that contribute to extracting these low level features are no longer needed to be re-learned across different domains/datasets. Therefore, in a transfer learning setting, it is imperative to decide which layers should be kept fixed (i.e., no new learning of parameters are needed) and which layers should be learnt/updated anew for the target dataset. The challenge, however, is, as the depth and complexity of CNNs are rapidly growing with the increasing availability of computational resources, it is increasingly becoming infeasible for general practitioners to handpick effective layers for update.

Many popular CNN architectures such as ResNet, MnasNet, GoogleNet, EfficientNet etc. are constituted of groups of convolutional layers which are called blocks. Each group of layer or block helps the model identify a low or high level characterizing feature. From this intuition, we pose the research question: *Can we automatically select appropriate blocks for update using a genetic algorithm that would yield at least similar accuracy to the baseline model?* The effect will be lesser blocks in the network (i.e., reduced number of parameters) to be learnt for transfer learning, thereby reducing the execution time.

4.1. Automatic Block Selection for Update by Genetic Algorithm

In our proposed scheme, a genetic algorithm is devised to select the blocks to be updated so as to reduce the training time, and, at the same time, to yield good accuracy in prediction for the target datasets. A genetic algorithm is a metaheuristic search algorithm that selects a good-enough solution from the vast search space of potential solutions. It trades off between exploration and exploitation, which means, optimizing a potential solution while escaping the local minima. This algorithm, broadly, works as follows [11, 22]. It begins its journey in the solution space with some potential solutions whose set is called *population*. It then selects two *parent* solutions from the solution pool based on some fitness function, and then applies two operations, namely crossover and mutation, to generate *children* solutions. This process is repeated until a good-enough solution is found. Use of the fitness function ensures exploitation of the search space, and use of randomization allows exploration of the search space.

In our algorithm, we maintain a binary array to denote the blocks of the network where each block is denoted by 0 or 1. A 1 means the block is selected for an update. Thus, for each solution or genotype g , if the i th gene g_i is equal to 1, then the corresponding i th block of the network, i.e., the feature extracted model, is selected for an update, which means, all of the layers of this block are selected as update layers. If the block is not selected, then its layers are frozen, i.e., the parameters of its layers are not updated. Initially, all the genes of the genotypes are set either 0 or 1 uniformly at random. Then, each genotype is evaluated against an evaluation function, which, in our case, is the classification accuracy. Then, the best two genotypes are chosen, and the

crossover and mutation operations are performed in order to get offspring genotypes. This crossover and mutation operations enforce exploration in the search space. We iterate this process for 100 rounds (known as epochs) to obtain the final (best) model.

4.2. Block Accuracy and Block Importance

In order to understand the impact of each block on accuracy of the target dataset, we evaluate the obtained test accuracy of the fine-tuned model on target datasets when only a particular block is selected to be updated. For each block, only that particular block in the feature extracted model is selected to be updated and the rest are frozen. The model is then fine-tuned on the target dataset for up to 20 epochs, and the evaluation accuracy of the final model is recorded as the block accuracy denoted by BA.

Activation feature map datasets for all the source and target datasets are generated for each block of the pre-trained model, which are then used to calculate the Optimal Transport Dataset Distance (OTDD) between the pairs of source and target datasets.

The OTDD is calculated according to the method of [2] using their implementation [3].

Following the definition of layer importance (defined in Eq. (1)), we define block importance, BI for b th block as follows:

$$BI(b) = \frac{OTDD(A_b^{\text{source}}, A_b^{\text{target}})}{OTDD(A_b^{\text{source}}, A_b^{\text{source}'}) + \epsilon}, \quad (2)$$

where source and source' denote two different subsets of the source dataset (as explained below Eq. (1)). This ratio adeptly captures the difference between the feature maps generated for the source and target datasets at a particular block. Datasets with similar feature sets should result in similar feature maps at an effective block, yielding a lower BI. Lower value of BI indicates higher adaptability of the model for target dataset.

4.3. Model

We apply our proposed scheme on a popular pre-trained network for transfer learning called EfficientNetB0 which has 237 layers grouped in 8 blocks. For larger models of the EfficientNet family, similar results are expected since they are just scaled up versions of the base model and the basic building blocks remain the same. Moreover, since our method is generic, we expect similar result on other pre-trained networks besides EfficientNet.

As mentioned earlier, EfficientNetB0 is pre-trained on ImageNet dataset. As target datasets, we employ three datasets, namely CIFAR-100, Food-101, and MangoLeafBD (details of these datasets are discussed in the next section). In contrast to the existing practice of selecting individual layers (as done in [24]), our genetic algorithm select blocks

of layers for update blocks from the feature-extracted model. The fully connected layers of the model are always selected as update layers.

Now we discuss the technical details of our algorithm. Regarding the values of hyper-parameters, we set up some empirical values as follows. We set the population size to 7. Elite plus roulette method is used as the selection method. The mutation probability is set to 1%.

The training phase is performed for 100 epochs with the Adam optimizer (with 0.0001 as the learning rate). We note here that while it is natural to try different combinations of values of these hyper-parameters, in this research our primary goal is to investigate the efficacy of a generic genetic algorithm. The best model obtained from the genetic algorithm is then trained on the target dataset. The training accuracy, validation accuracy, training time and the number of parameters are recorded as evaluation metrics.

5. Experimental Results

In this section we analyze the experimental results and discuss the findings. We measure the performance against five parameters: update layer/block selection time, training time, evaluation time, accuracy, and the number of parameters to be learnt.

5.1. Datasets and Experimental Settings

We use three target datasets: (1) CIFAR-100 [17], (2) Food-101 [6], (3) MangoLeafBD [1]. CIFAR-100 is an object recognition dataset with 100 classes each having 500 images for training and 100 images for testing. Food-101 is a food recognition dataset with 101 classes each of which has 750 training and 250 testing images. MangoLeafBD is a recently released dataset containing 4000 images of mango leaves with 8 classes of diseases. Figures 1, 2, and 3 show some sample images from the three datasets. All input images are shuffled, resized to 224×224 pixels, batched and pre-fetched for optimal data loading and training performance. All experiments are performed in a single NVIDIA GeForce RTX 2060 with a batch size of 32.

5.2. Baseline: Automatic Layer Selection by Genetic Algorithm

As the baseline method, we compare our method with the work of Nagae et al. [24]. The authors of this work use a genetic algorithm to select the appropriate update layers for the target datasets. So we think this work is an appropriate baseline for our proposed method.



Fig. 1. Sample images from CIFAR10 dataset.

Tab. 1. Performance comparison between the existing benchmark algorithm and our proposed algorithm on three datasets. LayerSelect – layer selection algorithm by [24]; BlockSelect – our proposed block selection scheme. Parameters: Runtime – time taken by the layer/block selection algorithm; Training time – neural network’s training time after selecting layers/blocks; Evaluation time – testing time; Accuracy – percentage of correct predictions on test set; # parameters – number of parameters to be learnt for learning the target datasets.

Dataset →	Food-101		CIFAR-100		MangoLeafBD	
Parameter ↓	LayerSelect	BlockSelect	LayerSelect	BlockSelect	LayerSelect	BlockSelect
Runtime	3 h	43 min	1 h 34 min	39 min	12 min 19 s	2 min 42 s
Training time	31 min	21 min	28 min	19 min	19 min	18 min
Evaluation time	42 ms	32 ms	31 ms	31 ms	58 ms	99 ms
Accuracy	0.77	0.79	0.81	0.82	1.0	0.997
# parameters	5, 79, 813	31, 13, 413	2, 56, 500	8, 30, 758	3, 61, 896	4, 24, 784

5.3. Performance Comparison

Table 1 presents the experimental results of Food-101, CIFAR-100, and MangoLeafBD target datasets. Here, LayerSelect indicates the layer selection algorithm of [24], and BlockSelect indicates our proposed algorithm. In the 1st column, runtime means the time taken by the layer/block selection algorithm, training time is the neural network’s training time after selecting layers/blocks, evaluation time means the testing time, accuracy is the percentage of correct prediction on test set, and finally # parameters is the number of parameters to be learnt for learning the target datasets.

From the experimental results we see that our proposed block selection algorithm works much faster than the existing baseline, i.e., the layer selection algorithm [24]. Still we achieve slightly better results in two datasets and slightly worse result in the



Fig. 2. Sample images from Food-101 dataset.

other dataset. Overall, the block selection algorithm is found to maintain similar level of accuracy while reducing the training and evaluation time.

From the experimental results it is evident that the block selection algorithm is faster than the layer selection method and yields a model that has similar or better accuracy and inference time.

5.4. Block Importance and Block Accuracy

Table 2 presents the values of Block Importance (BI) and Block Accuracy (BA) metrics for the target datasets, and Fig. 4 illustrates the results graphically. BI is calculated



Fig. 3. Sample images from MangoLeafBD dataset.

Tab. 2. Block Importance (BI) and Block Accuracy (BA) of various blocks for all three datasets.

Block	Food-101			CIFAR-100			MangoLeafBD		
	BI	Train BA	Test BA	BI	Train BA	Test BA	BI	Train BA	Test BA
1	1.420	0.84	0.72	1.952	0.88	0.72	1.491	1	0.993
2	1.484	0.83	0.72	1.471	0.89	0.72	1.459	1	0.995
3	1.073	0.83	0.72	1.152	0.89	0.72	1.259	1	0.995
4	1.072	0.82	0.72	0.964	0.90	0.71	1.078	1	0.995
5	1.011	0.84	0.72	1.021	0.89	0.72	0.029	1	0.995
6	0.959	0.84	0.72	1.000	0.90	0.71	0.815	1	0.995
7	1.132	0.82	0.72	0.965	0.89	0.72	0.022	1	0.995

using Eq. (2). BA is the obtained from the training and test accuracy when transfer learning is performed considering only the update layers of that block.

Conventionally, in transfer learning, it is believed that updating the layers close to the output side of the network is more effective. From our experimental data we cannot decisively claim this conjecture to be true, but it is evident that updating output side blocks works pretty well. We also see that updating blocks with lower BI values results in high training and testing accuracy. Though we cannot conclusively claim that updating the blocks with higher BI does not work, the trends shown by the experimental data evidently suggests that updating blocks with lower BI is effective for transfer learning.

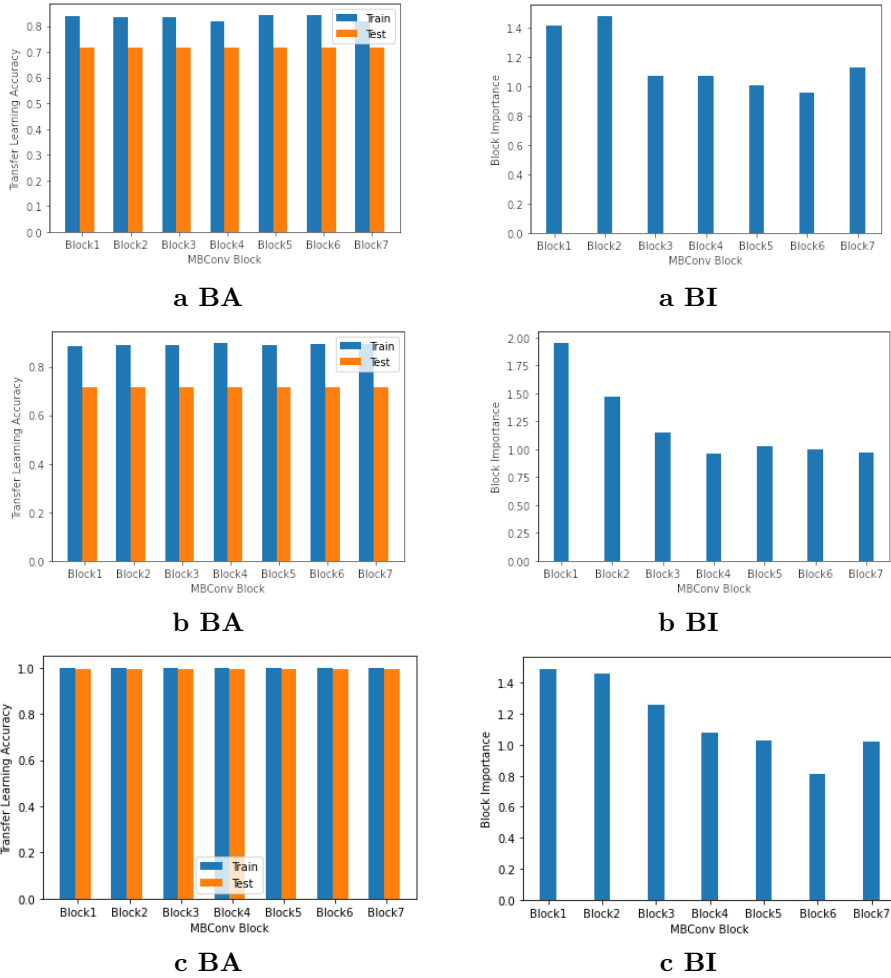


Fig. 4. Block Accuracy (BA) and Block Importance (BI) of various blocks of different datasets. (a) FOOD-100 dataset; (b) CIFAR-100 dataset; (c) MangoLeafBD dataset.

6. Conclusion

In order to assist practitioners of transfer learning select appropriate parameters of a convolutional neural network, in this research we have proposed a genetic algorithm-based solution that automatically selects CNN blocks, i.e., groups of layers, for the parameters to be updated only in these layers, for effective transfer learning. The proposed block

selection algorithm selects blocks of the network instead of layers, which results in less computational time requirement and yet yields similar or slightly better accuracy over the baseline method. Currently the proposed block selection algorithm is only implemented for a popular CNN called EfficientNet. This algorithm may easily be extended to other types of CNNs. The genetic algorithm devised here can also be honed using sophisticated methods and hyper-parameter tuning. In addition, other metaheuristic algorithms (like simulated annealing [12]) can be investigated to better select the layers and blocks for update.

References

- [1] S. I. Ahmed, M. Ibrahim, M. Nadim, M. M. Rahman, M. M. Shejunti, et al. MangoLeafBD: A comprehensive image dataset to classify diseased and healthy mango leaves. *Data in Brief*, 47:108941, 2023. doi:10.1016/j.dib.2023.108941.
- [2] D. Alvarez-Melis and N. Fusi. Geometric dataset distances via optimal transport. In: *Proc. 33th Int. Conf. Neural Information Processing Systems (NeurIPS)*, pp. 21428–21439. Curran Associates, 2020. https://proceedings.neurips.cc/paper_files/paper/2020/hash/f52a7b2610fb4d3f74b4106fb80b233d-Abstract.html.
- [3] D. Alvarez-Melis, Microsoft Open Source, and C. Yang. microsoft / otdd. GitHub. <https://github.com/microsoft/otdd>.
- [4] N. M. Aszemi and P. D. D. Dominic. Hyperparameter optimization in convolutional neural network using genetic algorithms. *International Journal of Advanced Computer Science and Applications*, 10(6):269–278, 2019. doi:10.14569/IJACSA.2019.0100638.
- [5] E. Bochinski, T. Senst, and T. Sikora. Hyper-parameter optimization for convolutional neural network committees based on evolutionary algorithms. In: *Proc. 2017 IEEE International Conference on Image Processing (ICIP)*, pp. 3924–3928, 2017. doi:10.1109/ICIP.2017.8297018.
- [6] L. Bossard, M. Guillaumin, and L. V. Gool. Food-101 – Mining discriminative components with random forests. In: *Proc. European Conference on Computer Vision (ECCV 2014)*, pp. 446–461, 2014. doi:https://doi.org/10.1007/978-3-319-10599-4_29.
- [7] R. Cai and J. Luo. Multi-task learning for multi-objective evolutionary neural architecture search. In: *Proc. 2021 IEEE Congress on Evolutionary Computation (CEC)*, pp. 1680–1687, 2021. doi:10.1109/CEC45853.2021.9504721.
- [8] D. J. Dong, W. Socher, R. Li, Li-Jia, K. Li, et al. ImageNet: A large-scale hierarchical image database. In: *Proc. 2009 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 248–255, 2009. doi:10.1109/CVPR.2009.5206848.
- [9] C. Fernando, D. Banarse, C. Blundell, Y. Zwols, D. Ha, et al. Pathnet: Evolution channels gradient descent in super neural networks. *arXiv*, 2017. ArXiv.1701.08734. doi:10.48550/ARXIV.1701.08734.
- [10] M. Ghafoorian, A. Mehertash, T. Kapur, N. Karssemeijer, E. Marchiori, et al. Transfer learning for domain adaptation in mri: Application in brain lesion segmentation. In: *Proc. International Conference on Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, p. 516–524, 2017. doi:https://doi.org/10.1007/978-3-319-66179-7_59.
- [11] D. E. Goldberg. *Genetic Algorithms*. Pearson Education India, 2013.

- [12] M. S. Haque, M. Fahim, and M. Ibrahim. An exploratory study on simulated annealing for feature selection in learning-to-rank. *International Journal of Intelligent Systems and Applications (IJISA)*, 16:86–103, 2024. doi:[10.5815/ijisa.2024.04.06](https://doi.org/10.5815/ijisa.2024.04.06).
- [13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. *Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, p. 770–778, 2016. doi:[10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90).
- [14] G. Huang, Z. Liu, K. Q. Weinberger, and L. van der Maaten. Densely connected convolutional networks. *Proc. 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, p. 2261–2269, 2017. doi:[10.1109/CVPR.2017.243](https://doi.org/10.1109/CVPR.2017.243).
- [15] Y. Huang, Y. Cheng, D. Chen, H. Lee, J. Ngiam, et al. GPipe: Efficient training of giant neural networks using pipeline parallelism. *arXiv*, 2019. ArXiv:1811.06965v5. doi:[10.48550/arXiv.1811.06965](https://doi.org/10.48550/arXiv.1811.06965).
- [16] S. Imai, S. Kawai, and H. Nobuhara. Stepwise pathnet: A layer-by-layer knowledge selection-based transfer learning algorithm. *Scientific Reports*, 10:8132, 2020. doi:[10.1038/s41598-020-64165-3](https://doi.org/10.1038/s41598-020-64165-3).
- [17] A. Krizhevsky. *Learning multiple layers of features from tiny images*. M.sc. thesis, University of Toronto, 2009. <https://www.cs.toronto.edu/~kriz/learning-features-2009-TR.pdf>.
- [18] A. Krizhevsky, I. Sutskever, and G. E. Hinton. ImageNet classification with deep convolutional neural networks. In: *Proc. Advances in Neural Information Processing Systems 25 (NIPS)*, p. 1097–1105. Curran Associates, 2012. <https://proceedings.neurips.cc/paper/2012/hash/c399862d3b9d6b76c8436e924a68c45b-Abstract.html>.
- [19] S. Lee, J. Kim, H. Kang, D.-Y. Kang, and J. Park. Genetic algorithm based deep learning neural network structure and hyperparameter optimization. *Applied Sciences*, 11(2):744, 2021. doi:[10.3390/app11020744](https://doi.org/10.3390/app11020744).
- [20] S. Loussaief and A. Abdelkrim. Convolutional neural network hyper-parameters optimization based on genetic algorithms. *International Journal of Advanced Computer Science and Applications*, 9(10), 2018. doi:[10.14569/IJACSA.2018.091031](https://doi.org/10.14569/IJACSA.2018.091031).
- [21] Z. Lu, H. Pu, F. Wang, Z. Hu, and L. Wang. The expressive power of neural networks: A view from the width. In: *Proc. Advances in Neural Information Processing Systems 30 (NIPS 2017)*, 2017. <https://proceedings.neurips.cc/paper/2017/hash/32cbf687880eb1674a07bf717761dd3a-Abstract.html>.
- [22] Z. Michalewicz. *Genetic Algorithms + Data Structures = Evolution Programs*. Springer, Berlin, Heidelberg, 2013. doi:[10.1007/978-3-662-07418-3](https://doi.org/10.1007/978-3-662-07418-3).
- [23] A. Mimi, S. F. T. Zohura, M. Ibrahim, R. R. Haque, O. Farrok, et al. Identifying selected diseases of leaves using deep learning and transfer learning models. *Machine Graphics and Vision*, 32(1):55–71, 2023. doi:[10.22630/MGV.2023.32.1.3](https://doi.org/10.22630/MGV.2023.32.1.3).
- [24] S. Nagae, D. Kanda, S. Kawa, and H. Nobuhara. Automatic layer selection for transfer learning and quantitative evaluation of layer effectiveness. *Neurocomputing*, 469:151–162, 2022. doi:[10.1016/j.neucom.2021.10.051](https://doi.org/10.1016/j.neucom.2021.10.051).
- [25] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010. doi:[10.1109/TKDE.2009.191](https://doi.org/10.1109/TKDE.2009.191).
- [26] M. Raghu, B. Poole, J. Kleinberg, S. Ganguli, and J. Sohl-Dickstein. On the expressive power of deep neural networks. In: *Proc. Advances in Neural Information Processing Systems 30 (NIPS 2017)*, 2017. <https://proceedings.neurips.cc/paper/2017/hash/32cbf687880eb1674a07bf717761dd3a-Abstract.html>.
- [27] H. Rehana, M. Ibrahim, and M. H. Ali. Plant disease detection using region-based convolutional neural network. *arXiv*, 2023. ArXiv:10.48550. doi:[10.48550/arXiv.2303.09063](https://doi.org/10.48550/arXiv.2303.09063).

- [28] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In: *Proc. 3rd International Conference on Learning Representations (ICLR)*, 2015. Accessible in arXiv. doi:[10.48550/arXiv.1409.1556](https://doi.org/10.48550/arXiv.1409.1556).
- [29] Y. Sun, B. Xue, M. Zhang, and G. G. Yen. Automatically evolving CNN architectures based on blocks. *arXiv*, 2019. ArXiv:1810.11875v2. doi:[10.48550/arXiv.1810.11875v2](https://doi.org/10.48550/arXiv.1810.11875v2).
- [30] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, et al. Going deeper with convolutions. In: *Proc. 2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 1–9, 2015. doi:[10.1109/CVPR.2015.7298594](https://doi.org/10.1109/CVPR.2015.7298594).
- [31] C. Szegedy, V. Vanhoucke, S. Ioffe, J. Shlens, and Z. Wojna. Rethinking the inception architecture for computer vision. In: *Proc. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2818–2826, 2016. doi:[10.1109/CVPR.2016.308](https://doi.org/10.1109/CVPR.2016.308).
- [32] M. Tan and Q. V. Le. EfficientNet: Rethinking model scaling for convolutional neural networks. In: *Proc. International Conference on Machine Learning (ICML)*, vol. 97 of *Proceedings of Machine Learning Research*, p. 6105–6114, 2019. <https://proceedings.mlr.press/v97/tan19a.html>.
- [33] H. Tian, S. Pouyanfar, J. Chen, S. Chen, and S. S. Iyengar. Automatic convolutional neural network selection for image classification using genetic algorithms. *Proc. 2018 IEEE International Conference on Information Reuse and Integration (IRI)*, pp. 444–451, 2018. doi:[10.1109/IRI.2018.00071](https://doi.org/10.1109/IRI.2018.00071).
- [34] C. Villani. *Optimal transport, Old and New*. Springer, 2008. doi:[10.1007/978-3-540-71050-9](https://doi.org/10.1007/978-3-540-71050-9).
- [35] L. Xie and A. Yuille. Genetic CNN. In: *Proc. 2017 IEEE International Conference on Computer Vision (ICCV)*, pp. 1388–1397, 2017. doi:[10.1109/ICCV.2017.154](https://doi.org/10.1109/ICCV.2017.154).
- [36] S. Zagoruyko and N. Komodakis. Wide residual networks. In: *Proc. British Machine Vision Conference (BMVC)*, pp. 87.1–87.12, 2016. <https://bmva-archive.org.uk/bmvc/2016/papers/paper087/index.html>.
- [37] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In: *Proc. European Conference on Computer Vision (ECCV 2014)*, vol. 8689 of *Lecture Notes in Computer Science*, p. 818–833, 2014. doi:[10.1007/978-3-319-10590-1_53](https://doi.org/10.1007/978-3-319-10590-1_53).
- [38] H. Zunair, N. Mohammed, and S. Momen. Unconventional wisdom: A new transfer learning approach applied to Bengali numeral classification. *Proc. 2018 International Conference on Bangla Speech and Language Processing (ICBSLP)*, 21:1–6, 2018. doi:[10.1109/ICBSLP.2018.8554435](https://doi.org/10.1109/ICBSLP.2018.8554435).

ENHANCED U-NET MODEL FOR ACCURATE AERIAL ROAD SEGMENTATION

Rayene Doghmane^{1,*}  and Karima Boukari^{2,**} 

¹Laboratory of Automatic and Signals of Annaba (LASA)

²Laboratory of Study and Research in Instrumentation and Communication of Annaba (LERICA)

Faculty of Technology, Badji Mokhtar-Annaba University, Annaba, Algeria

*Corresponding author: Rayene Doghmane (rayene.doghmane@univ-annaba.dz)

**Corresponding author: Karima Boukari (karima.boukari@univ-annaba.dz)

Abstract In computer vision, Convolutional Neural Networks (CNNs) have become a foundation for image analysis. They excel in tasks such as object recognition, classification, and more, semantic segmentation. In order to achieve better accuracy, it is crucial to apply normalization techniques to the network for enhancing overall performance. This paper introduces an innovative approach that incorporates Batch Group Normalization (BGN) into the popular U-Net for binary semantic segmentation, with a particular focus on aerial road detection. Our research primarily focuses on evaluating the BGN-UNet's performance compared to traditional normalization techniques, such as Batch Normalization (BN) and Group Normalization (GN). With a batch size of 2, the U-Net model enhanced with Batch Group Normalization (BGN-UNet) achieves a remarkable Mean IoU of 98.4% in aerial road segmentation, demonstrating its superior accuracy in this task.

Keywords: image analysis, image recognition, normalization techniques, batch group normalization, semantic segmentation, BGN-UNet, aerial road detection.

1. Introduction

Road extraction proves to be a crucial task in the analysis of remote sensing imagery [4]. It plays a significant role in various aspects of society and the economy. Despite its importance, accurately extracting roads faces challenges due to the presence of non-road objects, and the complexity of the background. These factors contribute to the difficulty of achieving precise road extraction. Addressing these challenges frequently requires the utilization of pixel-wise semantic segmentation to extract road areas accurately (Fig. 1).

Semantic segmentation, a fundamental task in computer vision, involves the classification of individual pixels in an image into distinct object categories, thus aiding in a good comprehension of visual content [12]. In this field, the UNet architecture has proven to be reliable and effective across various applications [27, 28].

Our research focuses on the more recent innovation, Batch Group Normalization (BGN). When integrated into the UNet architecture, BGN exhibits the potential to enhance the accuracy and efficiency of convolutional neural networks for tasks like binary semantic segmentation, with a specific emphasis on road detection.

Figure 2 displays different normalization techniques: Batch Normalization (BN), Layer Normalization (LN), Group Normalization (GN), and Batch-Group Normalization



Fig. 1. Binary Semantic Segmentation for Aerial Road Image. (a) Aerial Road Image. (b) Segmentation for the Aerial Road Image.

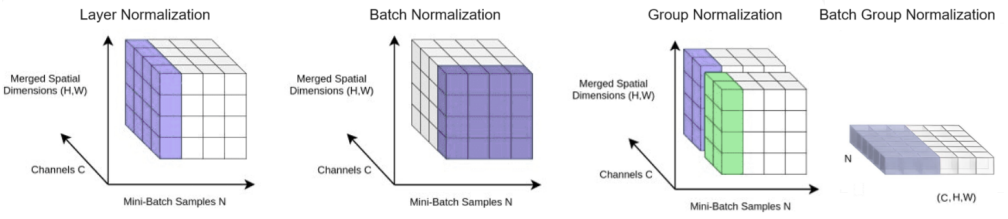


Fig. 2. Normalization Techniques.

(BGN). In each subfigure, you can see a feature map tensor, with axes representing the batch size (N), the number of channels (C), and the spatial dimensions (H, W). The pixels in purple are used to compute the statistics. BGN offers a unique perspective by combining the dimensions of channels, height, and width into a unified dimension and subsequently partitions this new dimension into distinct feature groups. This paper looks at how normalization methods in deep learning have changed over time. It also talks about how using BGN can make UNet better for tasks like semantic segmentation. The upcoming sections will offer comprehensive information on our research methods, the results of our experiments, and the discussions that follow. This will illuminate the power of combining U-Net and BGN in the constantly evolving fields of computer vision and deep learning.

The remainder of this paper is organized as follows. Section 2 reviews related works relevant to our study. Section 3 details the methodology, including the normalization techniques (Subsection 3.1), data preprocessing (Subsection 3.2), and data augmentation (Subsection 3.3) employed in the study. In Section 4, we describe the application of BGN-UNet to aerial road segmentation, with Subsection 4.1 in which the model is described, and Subsection 4.2 focusing on the contributions and novelty of the proposed

model. Section 5 presents the results and discussion, along with an ablation study (Subsection 5.1), practical applications of BGN-UNet in road detection (Subsection 5.2) and the challenges and considerations for practical implementation of BGN-UNet (Subsection 5.3). Finally, Section 6 concludes the paper by summarizing the key findings and implications of this work.

2. Related works

A novel end-to-end generative adversarial network was introduced by Zhang et al. [38] to carry out the road extraction task in aerial images. The combination of DCGAN and CGAN was utilized in their model for extracting roads from aerial images, followed by the replacement of deconvolutional layers with FCN. Therefore, the performance of the model is significantly impacted by data from different sources. A deep learning model, called the Recurrent Convolutional Neural Network U-Net (RCNN-UNet), was proposed by Yang et al. [34] for road detection and centerline extraction. It is an end-to-end deep learning model that exploits the spatial context and rich low-level visual features through the design of the RCNN unit. However, this can be computationally intensive and may require significant resources, including computational power and memory.

In addition, a novel deep learning-based convolutional network called VNet model with 2D convolutional kernel to extract road networks from high-resolution remote sensing imagery was introduced by ABOLFAZL et al. [2] a new objective loss function based on cross-entropy and dice loss (CEDL) was used to combine local information and global information, diminish the influence of class imbalance, and improve road segmentation results. However, the use of 2D convolutional kernels and the fully convolutional architecture can lead to significant computational overhead, particularly when processing large datasets. This may result in longer training times and increased resource requirements.

Furthermore, an improved road detection algorithm [10] that integrates Deep Convolutional Neural Networks (CNNs) with a Random Forest classifier has been proposed to enhance the accuracy of analyzing Very High Resolution (VHR) remotely sensed images. However, the performance of the algorithm is highly dependent on the quality and quantity of training data, which can pose a limitation in areas with insufficient labeled datasets.

Recent advancements in road extraction from high-resolution remote sensing images have been marked by the RADANet model [9], which employs a deformable attention mechanism to enhance feature extraction in complex environments, demonstrating superior performance compared to traditional techniques. However, a major limitation is its difficulty in effectively utilizing the spatial relationships and structure of roads, making it challenging to improve extraction accuracy in complex road settings.

According to Shaofu et al. [19] the proposed method MS-AGAN offers an efficient,

cost-effective, and reliable approach for dynamically updating road networks using high-resolution remote sensing images. However, despite its improved performance in road extraction, it may still encounter difficulties in complex scenes with high vegetation coverage and occlusions, leading to fragmentation and discontinuities in the extracted road networks.

Moreover, the authors of [18] propose an improved UNet++ network suitable for road extraction from high-resolution remote sensing images. By integrating the CBAM module and incorporating weight information in both channel and spatial dimensions of the feature map, this approach effectively suppresses the network's learning of non-road information, resulting in a more efficient and targeted model.

In summary, the proposed improvement of the UNet network for road extraction from remote sensing images offers notable strengths and weaknesses [30]. It enhances feature extraction through a CNN-Transformer architecture, improving segmentation accuracy with a double upsampling module and a combination of cross-entropy (CE) and Dice loss functions. This results in good training stability, robustness, and generalization across various datasets compared to established models like UNet, PSPNet, DeepLabV3, and TransUNet. However, the algorithm also exhibits high computational complexity and long training times, making it less suitable for mobile or embedded devices. Its resource-intensive nature may limit real-time applications, and the model's complexity could lead to overfitting on smaller datasets. Thus, while the approach shows promise in enhancing road extraction accuracy, its practical applicability is constrained by these limitations, indicating a need for future research into more efficient methods for image semantic segmentation.

Researchers in deep learning are always looking for ways, like normalization methods, to improve the training efficiency, generalization, and overall performance of neural networks. Throughout the years, a series of normalization techniques have been explored by researchers to enhance the training and generalization of deep learning models. This chronological survey encompasses the inception of Batch Normalization in 2015 by [14], extending to the more recent introduction of Batch Group Normalization (BGN) by Zhou in 2020 [42]. The timeline of normalization methods began with Batch Normalization (BN), which revolutionized deep learning by stabilizing training dynamics and accelerating convergence. Subsequently, Group Normalization (GN), proposed by Wu and He in 2018 [33], addressed certain limitations of BN, particularly when dealing with small batch sizes. Furthermore, Layer Normalization [7], Weight Normalization [25], Instance Normalization [29], and Positional Normalization [17] were introduced, each catering to specific requirements and contributing to the maturation of deep learning models.

3. Methodology

3.1. Normalization techniques

Normalization plays a crucial role in the training of Convolutional Neural Networks (CNNs) to ensure stable and effective learning. A standard normalization layer involves four steps: (1) grouping the feature map into distinct feature groups; (2) computing mean and variance statistics for each feature group; (3) normalizing each feature group using the calculated statistics; and (4) adjusting the normalized feature map to preserve the representation ability of the Convolutional Neural Network (CNN).

Batch Norm is a normalization technique done between the layers of a Neural Network instead of in the raw data [14]. In a neural network, batch normalization (BN) is achieved through a normalization step that fixes the means and variances of each layer's inputs [35] as schematically shown in Figure 2. Normalization is applied separately to each group of data, called a mini-batch, during the training process. This is a general formulation of feature normalization expressed as:

$$x_i := \frac{1}{\sigma_i}(x_i - \mu_i) . \quad (1)$$

Here, x represents the feature computed by a layer, and i is an index. In the context of 2D images, $i = (i_N, i_C, i_H, i_W)$ is a 4D vector indexing the features in the order (N, C, H, W) , where:

- N is the batch axis,
- C is the channel axis,
- H is the spatial height axis, and
- W is the spatial width axis.

In Batch Normalization, the transformation applied to the input feature to compute the normalized output is given by the following formula:

$$S_i = \{k \mid k_C = i_C\} , \quad (2)$$

where i_C (and k_C) denotes the sub-index of i (and k) along the C axis. This implies that pixels sharing the same channel index are normalized together. In other words, for each channel, Batch Normalization (BN) computes μ and σ along the (N, H, W) axes. In other words, the mean and variance are calculated along the batch dimension. Thus, the transformation helps to normalize the input and make the optimization process more stable during training.

According to [36], Batch Normalization does not work effectively for tasks requiring training with small batches, such as image segmentation, often due to memory limitations. Efforts have been made to explore alternative normalization techniques, including Group Normalization, where normalization is applied across partitions of features or

channels, with different pre-defined groups [33]. In Group Normalization (GN), the normalization is performed across partitions of features or channels. The formula for Group Normalization is as follows:

$$S_i = \left\{ k \mid k_N = i_N, \left\lfloor \frac{k_C}{C/G} \right\rfloor = \left\lfloor \frac{i_C}{C/G} \right\rfloor \right\}, \quad (3)$$

where:

- G is the number of groups (pre-defined hyper-parameter, with $G = 32$ by default),
- C/G is the number of channels per group,
- $\lfloor \cdot \rfloor$ denotes the floor operation,
- $\left\lfloor \frac{k_C}{C/G} \right\rfloor = \left\lfloor \frac{i_C}{C/G} \right\rfloor$ means that the indexes i and k are in the same group of channels, assuming each group of channels is stored in a sequential order along the C axis.

In Batch Group Normalization (BGN) technique, the channel, height, and width dimensions are initially concatenated into a new dimension, resulting in a flattened representation denoted as $F_{N \times D}$, where $D = C \times H \times W$. The mean μ_g in (4) and variance σ_g^2 in (5) are then computed along both the batch and the new dimension.

$$\mu_g = \frac{1}{N \times S} \sum_{n=1}^N \sum_{d=(g-1).S+1}^{g.S} f_{n,d}, \quad (4)$$

$$\sigma_g^2 = \frac{1}{N \times S} \sum_{n=1}^N \sum_{d=(g-1).S+1}^{g.S} (f_{n,d} - \mu_g)^2, \quad (5)$$

where $g = 1, \dots, G$ is a group index used in the group normalization technique and G is the number of groups that the new dimension is divided into, and is a hyper-parameter; $f_{n,d}$ is a member of $F(1)_{N \times D}$, representing a feature instance after merging the channel, height, and width dimensions into a new dimension; $D = C \times H \times W$, where C , H , and W are the channel, height, and width dimensions, respectively. Further, $S = M/G$ is the number of instances inside each divided feature group. The notation $g.S$ represents the range of feature instances included in group g for the calculation of the mean μ_g and variance σ_g^2 . Specifically, the summation over d goes from $(g-1).S+1$ to $g.S$, indicating that each group g contains S feature instances along the new dimension D .

BGN dynamically adjusts the number of feature instances used for statistical calculation, employing the group technique from Group Normalization (GN). When dealing with a small batch size, a smaller value for G is chosen to combine the entire new dimension, preventing noisy statistics. Conversely, with a larger batch size, a larger G is selected to partition the new dimension into smaller segments, facilitating the calculation of more accurate and less confused statistics. That is why, In our training process, $G=2$ was used for a batch size of 2 to combine the entire new dimension, and $G=32$ was used for batch sizes 8 and 16.

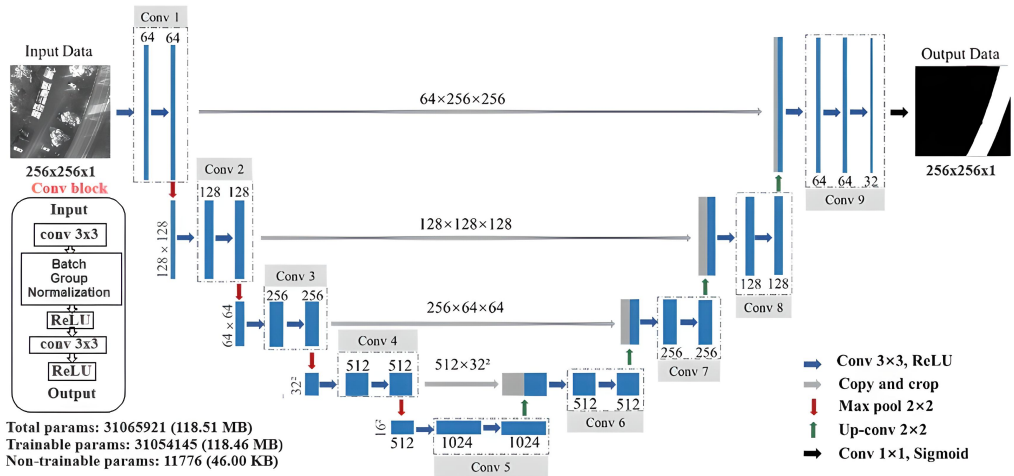


Fig. 3. BGN-UNet Architecture.

This multi-step approach allows BGN to capitalize on the generalization capabilities of GN, while also addressing the limitations of BN. Batch Group Normalization (BGN) has been implemented as a custom layer within the Keras deep learning framework.

Figure 3 provides a visual representation of the BGN-UNet model's architecture based on the original UNet architecture [23]. It illustrates how Batch Group Normalization (BGN) is incorporated into the UNet structure. Figure 4 compares the convolutional blocks in the standard UNet model with those in the BGN-UNet, emphasizing the differences in their structures and how BGN is employed.

BGN-UNet is defined for image segmentation tasks. Within this network, convolutional blocks, encoder blocks, and decoder blocks are included. The basic building block of the network is defined by the conv block function. It applies convolutional layers with BatchGroupNormalization and ReLU activation functions. This block is used to extract features from the input data. The encoder block function combines the convolutional block with max-pooling, allowing the network to progressively downsample the input image and capture high-level features. The decoder block function handles the upsampling and feature concatenation process. It uses transpose convolutional layers to increase the spatial resolution and combines the features from the encoder to refine the segmentation. The build unet function is responsible for constructing the entire UNet architecture, which consists of encoder and decoder blocks.

The choice of the final activation function is determined by the number of classes in the output. In the specific context of binary semantic segmentation, where the objective is to differentiate between two distinct classes, the choice of the activation function at the

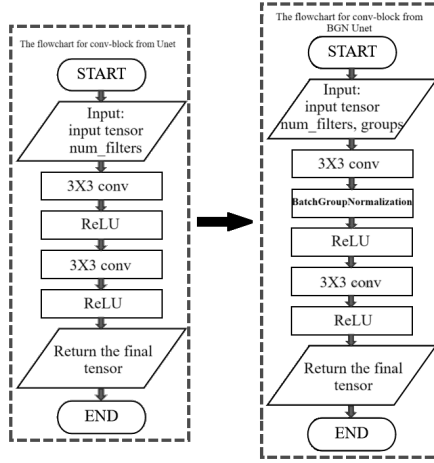


Fig. 4. Convolutional Blocks in UNet and BGN-UNet.

output layer is the sigmoid function which is preferable as output activation function [5]. This function assigns a continuous value to each pixel within the segmented image [31], ranging from 0 to 1. Pixels closer to a value of 1 are indicative of belonging to the road class, while those closer to 0 are representative of the non-road class. In essence, the BGN-UNet is designed for image segmentation tasks, where it takes an input image and produces a segmented image as the output. It effectively combines convolutional layers, normalization techniques, and upsampling to capture detailed features and produce accurate segmentations.

3.2. Data preprocessing

To make it easier to understand and analyze the data, it is imperative to systematically structure and format the dataset. The way data is prepared can vary significantly based on what we want to achieve with the data and the methods we plan to use for analysis [1]. In our research, a road dataset comprised of two distinct sequences is employed. The first sequence consists of 224 images, each possessing dimensions of 848×480 pixels, while the second sequence encompasses 109 images, each characterized by dimensions of 1280×720 pixels, all of these come with corresponding ground truth.

The dataset used for this study can be downloaded from internet and was previously utilized by [39]. However, we would like to note that the original source for downloading the dataset [40] appears to be unavailable at this time. In the meantime, the dataset can be accessed via the link in [41].

Data preprocessing is an indispensable phase in the pipeline of image analysis, particularly in the context of road segmentation. The following steps encapsulate the procedures employed to transform the raw data into a structured and manageable dataset, each step is guided by a specific rationale:

1. Color space conversion to grayscale

In the first step of our process, the original RGB images are transformed into grayscale versions. This is done to simplify the data and make it more efficient for our model. Grayscale images reduce complexity and memory usage since they remove color information while preserving the critical road segmentation details [15]. This approach improves computational efficiency and helps us optimize resource usage.

2. Padding for dimension alignment

Our goal here is to apply reflection padding to resize images in a way that their dimensions become divisible by 256. Reflection padding helps maintain the continuity and information within the image. The rationale behind using padding lies in its importance for achieving uniformity in image dimensions allowing the images to be divided into smaller images (patches) each measuring 256×256 pixels. After applying padding, the images in the first sequence have a size of 1024×512 pixels, and in the second sequence, the images are resized at 1280×768 pixels.

3. Patch creation for training

Our objective is to create image patches as a solution for segmenting larger images. This process involves dividing the larger images into smaller without overlapping. By partitioning the images into patches, a diverse training dataset is generated, consisting of 1792 patches from the first image sequence and 970 patches from the second sequence. The expression (6) below illustrates how to calculate the number of created patches as it is shown in Figure 5. The choice of a patch size of 256×256 has been made for the sake of computational efficiency, striking a balance between capturing adequate spatial information and maintaining a manageable computational load. It is observed in the literature and established practices for similar tasks that a patch size of 256×256 is commonly chosen, reflecting a widely adopted approach in the field [16, 21].

$$n = \left(\frac{\text{SIZE_X}}{\text{patch_size}} \right) \left(\frac{\text{SIZE_Y}}{\text{patch_size}} \right) \quad (6)$$

4. Elimination of non-informative patches

Pruning non-informative patches is very important for data quality and model efficiency. By retaining only patches with pertinent road-related content as shown in Figure 6, it is ensured that the training dataset is composed exclusively of relevant information, enhancing the model's accuracy and mitigating the inclusion of noise or irrelevant details. The selection of patches was automated based on the presence of information in the corresponding masks, avoiding manual elimination [8].



Fig. 5. Creating patches. (a) Patches from padded grayscale image; (b) patches from padded ground truth.

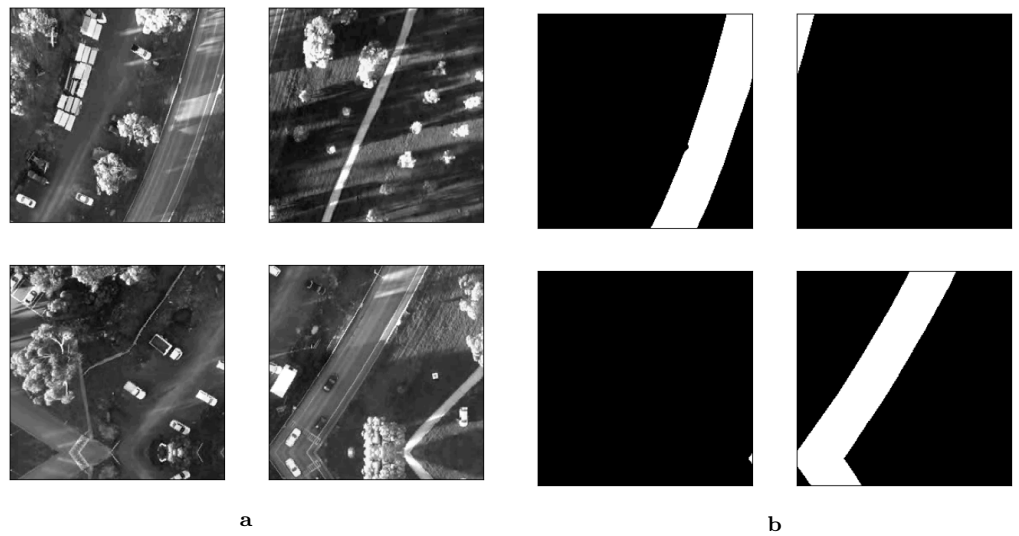


Fig. 6. Elimination of non-informative patches. (a) Informative images patches; (b) informative ground truth patches.

5. Fusion of image sequences

The fusion of image sequences yields a unified dataset of 906 images. This fusion enhances the dataset’s richness, incorporating diverse scenarios from both sequences, and increases the model’s capacity to generalize across different road environments and improving the robustness of the road segmentation model.

6. Data normalization

Data normalization is the process of transforming raw data values to another form

with properties [3]. In this case, our data must be more suitable for neural network algorithms which require data that are on a 0-1 scale.

Each of these steps contributes to the overall data preprocessing strategy, ensuring that the dataset is meticulously prepared and optimized for the road segmentation task. Given that our dataset is relatively small, with images of 1024×512 and 1280×768 pixels, we have the opportunity to train our UNet model from scratch, which is particularly advantageous for adapting to the specific characteristics of our dataset.

3.3. Data augmentation

Data augmentation is a technique used to increase the size of a dataset by applying various transformations to the original images. This technique is particularly useful in deep learning tasks, where a large amount of data is required to train the model effectively [26]. In this study, data augmentation was employed to augment the size of the dataset, which consisted of 906 grayscale images with dimensions of 256×256 , for aerial road segmentation using the UNet model and batch group normalization.

While it's true that a small dataset might limit how well the model generalizes to larger datasets, image augmentation techniques can help improve performance by generating additional training samples through techniques like rotation and flipping. This effectively increases the diversity of the training data without needing more labelled samples. In fact, augmentation can help prevent overfitting and improve model robustness, leading to better performance even when applied to larger datasets [37]. Although a small dataset is a limitation, these techniques can mitigate its impact and enhance generalizability.

Even though the study uses a small dataset, the method, along with data augmentation, can also work well with larger datasets. We plan to test its effectiveness on larger datasets in future research.

Also, according to [24] the size of the dataset required may depend on various factors such as the complexity of the task and the number of parameters. This statement implies that for simpler tasks or those with fewer parameters, smaller datasets may suffice for effective training.

The data augmentation techniques presented in [6], including horizontal and vertical flipping to help the model learn to recognize road patterns in different orientations, as well as rotation, were implemented. These techniques were applied to the original images to generate new training examples. The augmented dataset was then used to train the UNet model with batch group normalization. In our approach to aerial road segmentation, data augmentation played a crucial role, enabling an increase in the size of our dataset and an improvement in the accuracy of our model. The utilization of a highly accurate and efficient model for aerial road segmentation was achieved through the combination of UNet and batch group normalization with data augmentation.

4. Aerial road segmentation using BGN-UNet

4.1. The model

The UNet model is constructed with a series of convolutional and decoder blocks, each meticulously designed to capture and refine features for binary segmentation tasks. The convolutional block (`conv_block`) is a crucial building block, comprising two consecutive 3×3 convolutional layers. Notably, Batch Group Normalization is incorporated into the (`conv_block`), enhancing the stability and efficiency of the network during training. Following the convolutional layers, Batch Group Normalization is applied, followed by Rectified Linear Unit (ReLU) activation, synergistically contributing to feature extraction. On the other hand, the decoder block (`decoder_block`) leverages a transposed convolutional layer with a 2×2 kernel for effective upsampling. The upsampled features undergo concatenation with corresponding features from the encoder block and are subsequently processed through the (`conv_block`) to extract informative features. Throughout the UNet architecture, the encoder systematically downsamples the input image through convolution and max-pooling operations. Conversely, the decoder adeptly upsamples the features to generate a segmentation map. In the context of this binary segmentation task, the final layer of the model employs a 1×1 convolution with a sigmoid activation function, facilitating the precise prediction of pixel-wise binary masks. This thoughtful architectural choice, integrating 3×3 convolutions, Batch Group Normalization, and the appropriate activation function, underscores the model's efficacy in capturing spatial information and thereby enhancing its performance in accurately delineating objects of interest in the input images.

To evaluate the performance of the proposed model, experiments were conducted using an aerial road dataset described in the research paper titled Efficient Road Detection and Tracking for Unmanned Aerial Vehicles [39]. Specifically, 1792 aerial images from this dataset were utilized. As previously explained in the data preprocessing section 3.2, before training the model, the data is prepared. To enhance manageability, the large images are partitioned into smaller 256 by 256 pixel images, commonly referred to as patches, and are represented in grayscale. This approach simplifies data processing and model training. The dataset used in this study comprises a total of 906 informative patches extracted from road aerial images. To facilitate the training and evaluation of our model, the dataset is randomly partitioned into three distinct sets. These sets are designated for various purposes: one is allocated for training the model, another is reserved for validation during the training process, and the final set serves as the test set for the evaluation of model performance. The division is conducted using the `train_test_split` function, which separates the image and mask datasets into 724 images for training (80%), 91 images for validation (10%), and 91 images for testing (10%), in order to ensure the consistency of results across experiments.

Figure 7 highlights the flowchart of methodologies used in this paper. The model

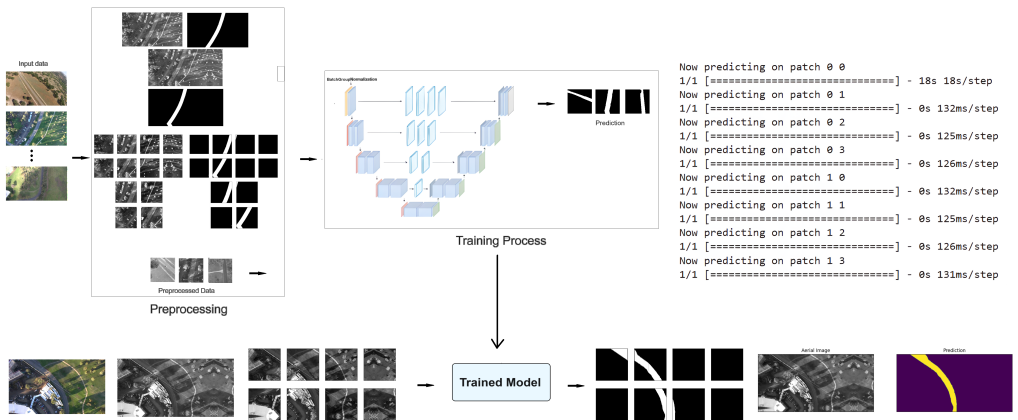


Fig. 7. Flowchart of the proposed methodology.

hyperparameters, as outlined in the Table 1, encompass settings for the training process and data information. In general, our model produces favorable results with minimal distortion and negligible false detections, as shown in Figure 8. In Figure 9, four randomly selected patches from the test set are showcased, segmented using the BNG-UNet model. These patches present varying levels of difficulty while consistently demonstrating amazing segmentation results.

Tab. 1. Model hyperparameters.

Parameter	Value
Learning Rate	1×10^{-3}
Loss Function	Binary Cross-entropy
Epochs	25
Batch Size	16
Group	32
Optimizer	Adam Optimizer
Validation Split	0.20, Random State = 42
Total Params	31 065 921 (118.51 MB)
Trainable Params	31 054 145 (118.46 MB)
Non-Trainable Params	11 776 (46.00 KB)
Image Data Shape	(906, 256, 256, 1)
Mask Data Shape	(906, 256, 256, 1)
Max Pixel Value in Image	255
Labels in the Mask	[0, 255]
Patch Size	$256 \times 256 \times 1$



Fig. 8. Semantic Segmentation of the Aerial Road Image with BGN-UNet

4.2. Contributions and novelty

This paper presents several significant contributions and novel aspects in the domain of aerial road segmentation using an Enhanced U-Net model, specifically incorporating Batch Group Normalization technique (BGN). The key contributions are outlined below:

A novel normalization technique, Batch Group Normalization (BGN), has been incorporated into the U-Net architecture. This method addresses the performance limitations of Batch Normalization (BN) at very small or extremely large batch sizes by leveraging the grouping strategy employed in Group Normalization (GN). It combines the channel, height, and width dimensions into a unified representation, partitions this dimension into feature groups, and computes the statistics across both the feature groups and the entire mini-batch to enhance performance. By effectively stabilizing training dynamics, BGN enhances model convergence and accuracy.

Building upon this, the proposed BGN-UNet architecture modifies the traditional U-Net framework by implementing BGN layers, which allows for better feature extraction and representation in aerial imagery. This adaptation is particularly beneficial for binary semantic segmentation tasks, where precise delineation of road areas is critical.

In line with these theoretical improvements, our experimental results demonstrate that the BGN-UNet achieves a Mean Intersection over Union (IoU) of 98.4% (See section 5), significantly outperforming traditional normalization techniques such as Batch Normalization (BN) and Group Normalization (GN). This remarkable accuracy underlines the effectiveness of our proposed model in real-world applications.

Furthermore, a thorough comparative analysis of the BGN-UNet model is provided against several state-of-the-art models for aerial road segmentation. This evaluation highlights the superior performance of the approach and discusses its robustness in handling complex urban environments with various road types and conditions.

In summary, the distinctive contributions of this paper lie in the innovative integration of BGN into the U-Net architecture, achieving superior segmentation accuracy, conducting comprehensive evaluations against existing models, and addressing practical

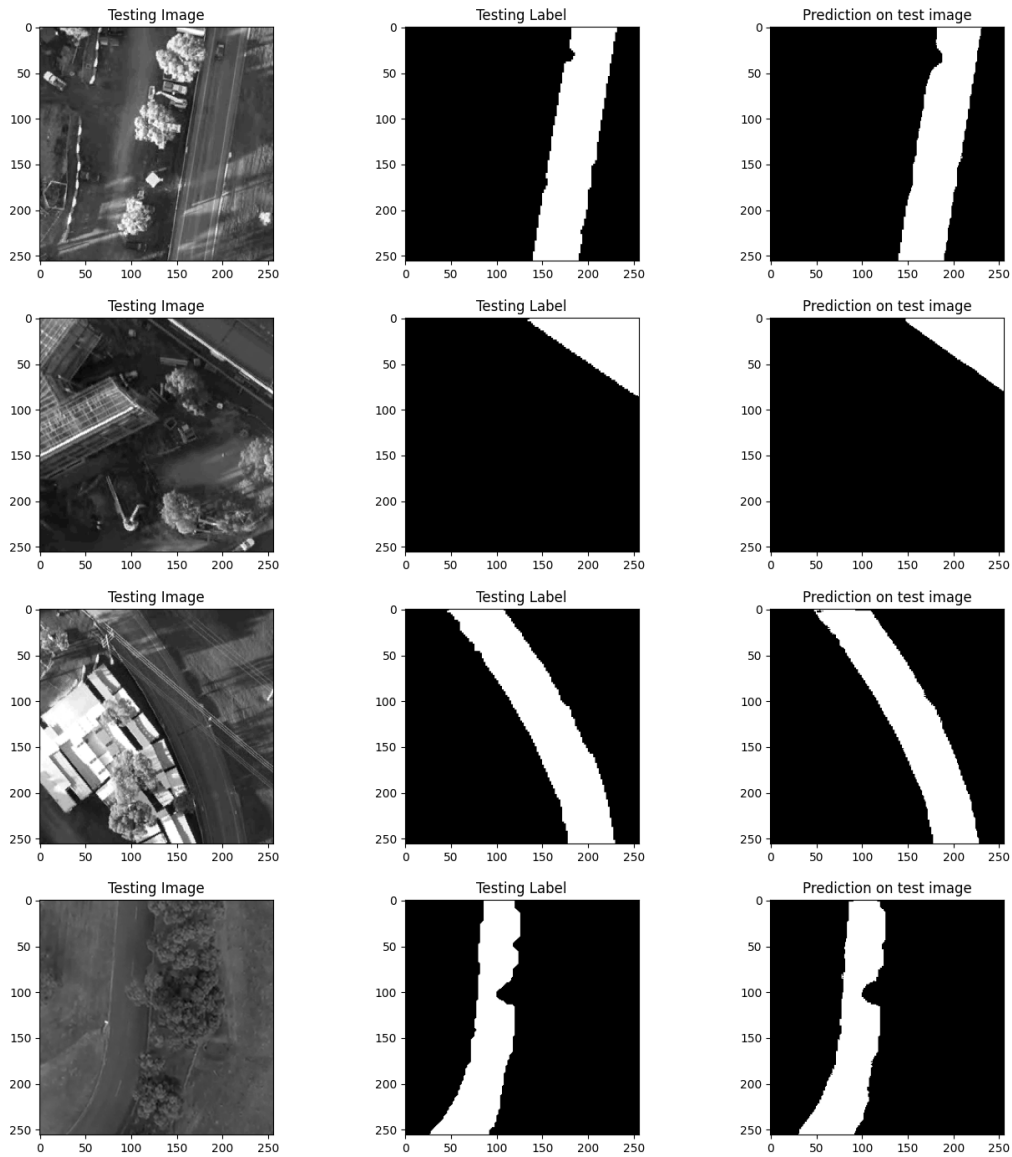


Fig. 9. Road segmentation using BGN-UNet

challenges in aerial road detection. These elements collectively underscore the novelty and significance of our research in advancing methodologies for semantic segmentation in aerial imagery.

5. Results and discussion

Among the various normalization methods, two specific techniques, Batch Normalization (BN) and Group Normalization (GN), have been chosen as the baseline methods to be used in the U-Net architecture. These methods will serve as reference points for comparing the performance of the novel Batch-Group Normalization (BGN) technique when combined with the UNet architecture, which is being introduced and evaluated. By comparing BGN-UNet to these established normalization techniques, we can assess its effectiveness and potential advantages. This evaluation allows us to understand the performance of the novel Batch-Group Normalization (BGN) technique when integrated with the U-Net architecture.

In our UNet model, BGN has been utilized to optimize the network's performance. Testing was conducted in image segmentation, a challenging task, with a specific focus on its performance with aerial imagery. Initial results suggest that BGN might offer advantages over traditional BN and GN methods.

In our comprehensive experimental design, an assessment of three distinct models BN-UNet, GN-UNet, and BGN-UNet was conducted. The impact of varying batch sizes (2, 8, and 16) across a consistent 25 epochs training period was systematically explored. Despite the constraints of a small dataset and only one GPU, our model still achieves good results. The selection of the Adam optimizer, a widely recognized choice in deep learning, greatly facilitated our training process by ensuring efficient convergence and adaptive learning rates [22]. To fit our approach to binary segmentation, focusing on the detection of road and non-road classes, the binary cross-entropy loss function was employed. Cross-entropy serves as a loss function in neural networks in machine learning, offering a metric to gauge the likeness between predicted and actual values [13]. While our resources may appear limited, our results consistently demonstrated that BGN-UNet outperformed both BN-UNet and GN-UNet, particularly in terms of Mean Intersection over Union (IoU).

In our experiments with three different normalization methods BN-UNet, GN-UNet, and BGN-UNet using varying batch sizes of 2, 8, and 16, interesting results were observed in terms of Mean IoU, which is a measure of segmentation accuracy. For the smallest batch size 2, the BGN-UNet achieved a Mean IoU of 0.9727, while BN-UNet and GN-UNet had Mean IoU scores of 0.9673 and 0.9687, respectively. As the batch size was increased to 8, it was observed that BGN-UNet outperformed both GN-UNet and BN-UNet, achieving a higher accuracy with a Mean IoU value of 0.9729. In comparison, GN-UNet had a Mean IoU of 0.9724, while BN-UNet lagged behind with a score of

Tab. 2. Mean IoU for Different Batch Sizes and Models

Model	Batch Size 2	Batch Size 8	Batch Size 16
BN-UNet	0.96733713	0.96600366	0.97286165
GN-UNet	0.9687331	0.97235453	0.9730376
BGN-UNet	0.9726844	0.9729512	0.9740099
BGN-UNet+data augmentation	0.98135	0.9821	0.9840

0.9660. When using a batch size of 16, BGN-UNet recorded the highest Mean IoU of 0.9740, outperforming both BN-UNet 0.9729 and GN-UNet 0.9730. In summary, BGN-UNet consistently achieved the best segmentation accuracy across all batch sizes, outperforming both BN-UNet and GN-UNet. These results highlight the effectiveness of Batch-Group Normalization (BGN) in improving the U-Net model's performance in semantic segmentation tasks. The Table 2 shows the Mean IoU values for various batch sizes in the BN-UNet, GN-UNet, and BGN-UNet models.

When predicting patches from a large image (1024×512) using the BGN-UNet model, it has been observed that the first patch takes approximately 18 seconds to process (compared to 12 seconds for the BN-UNet model and 15 seconds for the GN-UNet model.), while the remaining 7 patches are predicted almost instantaneously. This initial delay is due to model initialization overhead, which includes tasks such as loading weights, setting up the neural network in memory, and performing preprocessing steps. Once these tasks are completed, the model is fully operational, allowing for the rapid prediction of subsequent patches. BN-UNet is generally faster because it relies on batch normalization, which uses batch statistics, resulting in faster initial processing. On the other hand, GN-UNet uses group normalization, which normalizes within groups without depending on batch size, leading to a middle-ground processing time compared to BN-UNet and BGN-UNet. In summary, the 18 seconds for the first patch mainly come from initialization and setup overhead, while the near-zero time for the following patches reflects efficient reuse of the initialized model, significantly speeding up the process for the remaining patches – see Fig. 7.

Our results show that BGN-UNet is very adaptable and efficient, making it a valuable tool for tasks like binary segmentation, especially in situations where data is limited, and computational resources are constrained. To summarize, BGN-UNet appears to perform well with small datasets in binary segmentation tasks, as our findings indicate. The visual representations of training and validation Metrics as shown in Figs. 10, 11, and 12 provide a comprehensive overview of the performance evaluation of the three models under varying batch conditions. Favorable results with minimal distortion and negligible false detections can be observed in the BGN-UNet graphs.

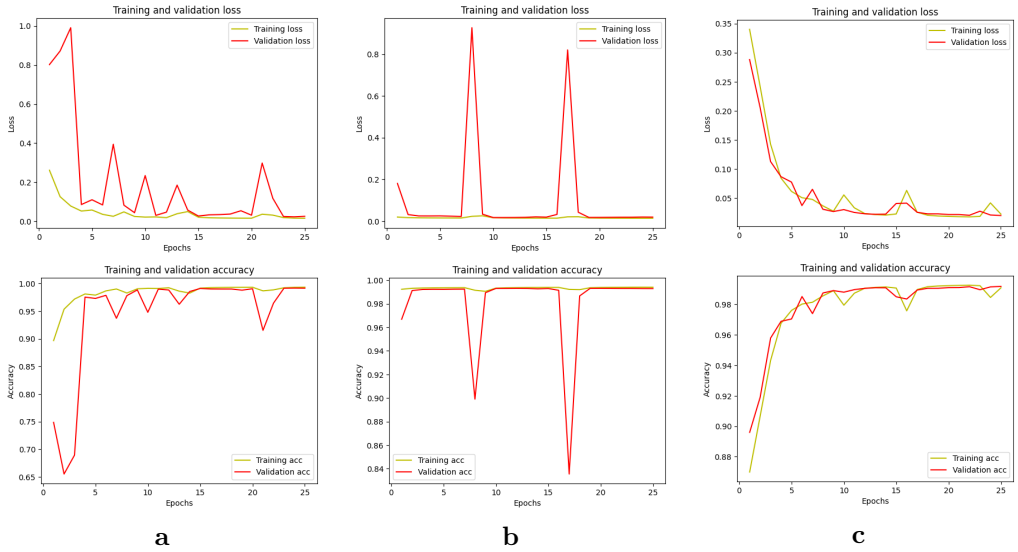


Fig. 10. Training and validation metrics for batch size 2: (a) BN-UNet; (b) GN-UNet; (c) BGN-UNet.

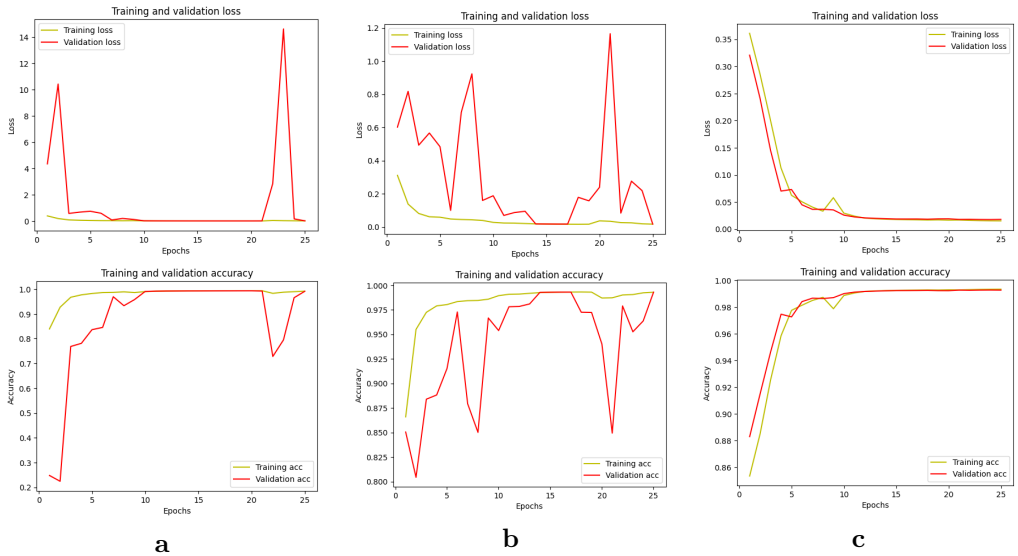


Fig. 11. Training and validation metrics for batch size 8: (a) BN-UNet; (b) GN-UNet; (c) BGN-UNet.

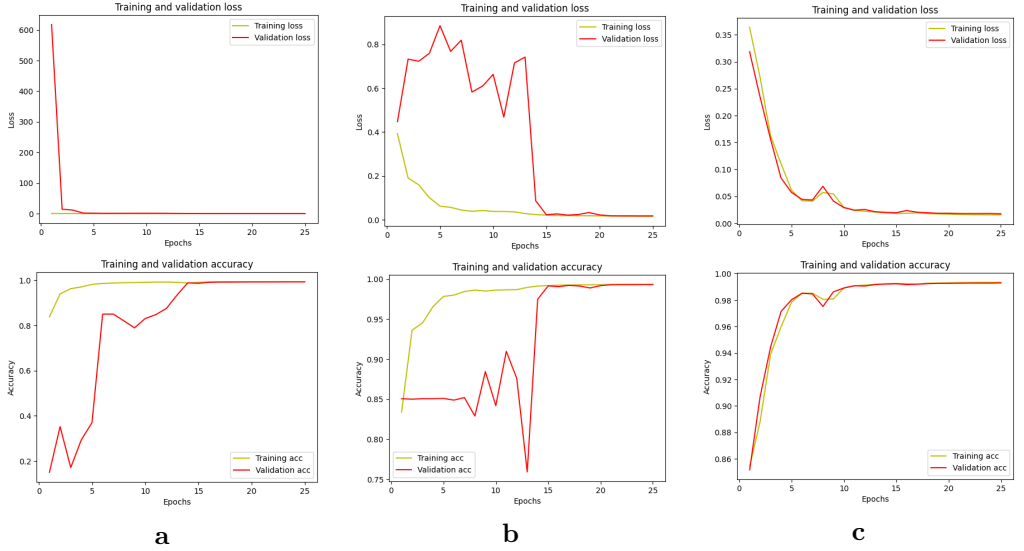


Fig. 12. Training and validation metrics for batch size 16: (a) BN-UNet; (b) GN-UNet; (c) BGN-UNet.

5.1. Ablation study

The choice of batch size significantly influences the training dynamics and convergence of deep learning models. To understand the effect of batch size on model performance, we conducted experiments using three different batch sizes: 2, 8, and 16. The performance metric used for evaluation was the Mean IoU (Intersection over Union). The results for the BGN-UNET model are as follows: for Batch Size 2, the Mean IoU was 0.98135; for Batch Size 8, the Mean IoU increased slightly to 0.9821; and for Batch Size 16, the highest Mean IoU of 0.9840 was achieved.

As the batch size increased, we observed a slight improvement in Mean IoU, with batch size 16 yielding the highest performance. No signs of overfitting were detected, and the model remained stable across all batch sizes, indicating that varying the batch size had minimal impact on overall performance. However, the slight increase in performance with larger batch sizes suggests that batch size 16 may have been more effective in stabilizing gradients during training, leading to better results.

In contrast, when using traditional normalization techniques, we observed different performance metrics. The results when using Batch Normalization are as follows: for Batch Size 2, the Mean IoU was 0.9673; for Batch Size 8, the Mean IoU slightly decreased to 0.9660; and for Batch Size 16, the Mean IoU improved to 0.9729. When using Group Normalization, the results were: for Batch Size 2, the Mean IoU was 0.9687; for Batch

Size 8, the Mean IoU increased to 0.9724; and for Batch Size 16, the highest Mean IoU of 0.9730 was achieved.

These results indicate that while larger batch sizes can enhance performance in some cases, the integration of Batch Group Normalization (BGN) allows for effective training even with small batch sizes, mitigating some of the common issues associated with traditional normalization techniques that struggle under similar conditions.

Overall, these findings illustrate how batch size affects various aspects of model training, including convergence behavior and generalization capability. The slight improvements observed with larger batch sizes suggest a potential benefit in stabilizing gradients during training. However, BGN's effectiveness with smaller batches highlights its advantage in scenarios where data availability is limited. In future work, we aim to explore larger batch sizes of 32 and 64 to further assess their impact on model performance and training dynamics.

5.2. Practical applications of BGN-UNet in road detection

The BGN-UNet model offers significant potential in road detection tasks across multiple domains due to its advanced segmentation capabilities and adaptability to high-resolution imagery. This subsection explores several practical applications where BGN-UNet could contribute to improving accuracy and efficiency, including urban road mapping, autonomous vehicle navigation, infrastructure monitoring, disaster response, and traffic analysis. Each example highlights specific contexts in which BGN-UNet's performance may address current challenges and enhance practical outcomes.

• Urban Road Mapping

BGN-UNet can be employed to accurately map urban road networks from high-resolution satellite or aerial imagery. Example: Similar to the C-UNet model, which improved road extraction accuracy in remote sensing images, BGN-UNet could enhance urban planning and traffic management by providing precise road layouts.

• Autonomous Vehicle Navigation

In autonomous driving systems, BGN-UNet can be utilized for real-time lane and road boundary detection. Example: A project using a UNet model for lane detection demonstrated high accuracy on diverse driving scenarios, showcasing how deep learning models can effectively identify drivable areas under various conditions.

• Infrastructure Monitoring

BGN-UNet can assist in monitoring the condition of roads by detecting cracks and other surface anomalies. Example: Research has shown that U-Net architectures can be adapted for crack detection in tunnels and roads, emphasizing the model's capability to automate infrastructure inspections and enhance maintenance strategies.

• Disaster Response and Recovery

After natural disasters, BGN-UNet can help assess road damage by analyzing satellite imagery to identify blocked or damaged routes. Example: Similar methodologies have been applied in post-disaster scenarios where rapid assessment of road conditions is crucial for effective emergency response.

- **Traffic Analysis and Management**

The model can be used to analyze traffic patterns by segmenting roads from video feeds or images captured by drones. Example: The integration of deep learning models has shown promise in extracting road features from very-high-resolution images, which could be adapted for real-time traffic analysis.

5.3. Challenges and considerations for practical implementation of BGN-UNet

While our paper primarily focuses on the advantages of the BGN-UNet model compared to traditional normalization techniques like Batch Normalization (BN) and Group Normalization (GN), we acknowledge the importance of discussing the potential challenges associated with implementing BGN-UNet in practical applications. Key considerations include the following.

- **Execution runtime**

Implementing BGN can significantly extend the execution runtime during model training compared to simpler normalization techniques. However, the improved results it yields justify the added computational cost.

- **Memory Requirements**

BGN-UNet can demand higher memory usage due to the need to maintain statistics for multiple groups within a batch. As seen in the literature [20], larger batch sizes need more memory for activations and gradients.

- **Sensitivity to Hyperparameters**

The effectiveness of BGN-UNet may depend on the careful tuning of hyperparameters, such as the number of groups and batch size. Our results demonstrate that, despite variations in batch size, BGN-UNet consistently outperforms the other models.

- **Generalization Across Domains**

While BGN-UNet has shown promise in specific tasks like aerial road segmentation, its generalizability to other domains remains an open question. While U-Net performs well in biomedical applications [32], the integration of BGN would further enhance its performance. Future work will explore how BGN-UNet performs across various tasks. This investigation will help clarify the effectiveness of BGN-UNet in diverse applications and identify potential limitations in its adaptability.

- **Complexity of Implementation**

Incorporating BGN into existing architectures may require more complex modifications compared to standard normalization techniques. This can be a challenge for practitioners with limited experience in deep learning.

In conclusion, while BGN-UNet offers notable advantages for training stability and performance, it is essential to consider these potential challenges when implementing it in practical scenarios. Future work could explore these aspects further, providing insights into optimizing BGN-UNet for various applications and understanding its limitations.

6. Conclusion

Our study presents an innovative approach by incorporating Batch Group Normalization (BGN) technique into the well-known UNet architecture for binary semantic segmentation, with a particular focus on road detection. We evaluated the performance of BGN-UNet in comparison to BN-UNet and GN-UNet, and our experimental results underscored the superior performance of the BGN-UNet model. The careful preprocessing of the dataset played a significant role in the success of this segmentation task. Integrating BGN as a custom layer in the Keras deep learning framework allowed us to make good use of its benefits and incorporate it into the UNet architecture. The research concluded that BGN-UNet is a valuable network for aerial road segmentation, even in situations with limited data and constrained computational resources. The overall outcome of our study was to enhance UNet model, offering an innovative approach to semantic segmentation with consistently superior results. Our proposed model experienced relatively faster convergence compared to baseline networks such as BN-UNet and GN-UNet, easily achieving 0.984 Mean IoU benchmark within only 25 epochs of training. We believe that further enhancements can be made to our model, not only in terms of training on a single GPU and limiting the training to 25 epochs with a maximum batch size of 16, but also in exploring further optimizations in hyperparameter tuning, dataset augmentation, and potentially leveraging distributed computing resources. These steps may enhance our model's performance even further. While existing research has demonstrated the effectiveness of various Convolutional Neural Networks (CNNs) for aerial image analysis, there remains a gap in the application of advanced normalization techniques to improve segmentation accuracy specifically for road detection. This study aims to address these gaps by introducing the enhanced U-Net model with Batch Group Normalization (BGN). By advancing the U-Net model with Batch Group Normalization, we not only aim to bridge existing gaps in segmentation accuracy but also to inspire innovation and improvement in the broader field of computer vision, underscoring the importance of continuous advancements in all domains.

BGN-UNet has performed well in aerial road segmentation, but its ability to handle other tasks like medical image segmentation and autonomous driving is still uncertain. While U-Net was originally designed for biomedical image segmentation, the differences in data types and challenges in these tasks may influence the performance of BGN-UNet. However, it is expected that BGN-UNet could enhance the capabilities of U-Net in these areas. Future research will test BGN-UNet in these tasks to better understand its effectiveness and potential limitations across a wider range of applications. Also, we will keep improving our model and see how well it works with different types of datasets.

Acknowledgement

We acknowledge the use of Google Colab [11] for providing access to free GPU instances, which were instrumental in conducting the research and performing the necessary computations related to the semantic segmentation of aerial images using the UNet model and batch group normalization.

References

- [1] Z. S. Abdallah, L. Du, and G. I. Webb. Data preparation. In: D. Phung, G. I. Webb, and C. Sammut (Eds.), *Encyclopedia of Machine Learning and Data Science*, pp. 1–10. Springer US, New York, NY. 2023. doi:10.1007/978-1-4899-7502-7_62-2. Living reference work entry [Accessed: 2023].
- [2] A. Abdollahi, B. Pradhan, and A. Alamri. VNet: An end-to-end fully convolutional neural network for road extraction from high-resolution remote sensing data. *Ieee Access* 8:179424–179436. 2020. doi:10.1109/ACCESS.2020.3026658.
- [3] A. Abdollahi, B. Pradhan, and N. Shukla. Road extraction from high-resolution orthophoto images using convolutional neural network. *Journal of the Indian Society of Remote Sensing* 49:569–583. 2021. doi:10.1007/s12524-020-01228-y.
- [4] A. Abdollahi, B. Pradhan, N. Shukla, S. Chakraborty, and A. Alamri. Deep learning approaches applied to remote sensing datasets for road extraction: A state-of-the-art review. *Remote Sensing* 12(9):1444. 2020. doi:10.3390/rs12091444.
- [5] M. I. Ahmed, M. Foysal, M. D. Chaity, and A. B. M. A. Hossain. DeepRoadNet: A deep residual based segmentation network for road map detection from remote aerial image. *IET Image Processing* 18:265–279. 2023. doi:10.1049/ipr2.12948.
- [6] S. Arkhangelskiy. Data augmentation on GPU in Tensorflow. In: *Becoming Human. Exploring Artificial Intelligence & What it Means to be Human*. Medium. 2017. <https://becominghuman.ai/data-augmentation-on-gpu-in-tensorflow-13d14ecf2b19>.
- [7] J. L. Ba, J. R. Kiros, and G. E. Hinton. Layer normalization. In: *Proc. Neural Information Processing Systems – NIPS 2016 Deep Learning Symposium*. 2016. https://openreview.net/forum?id=BJLa_ZC9.
- [8] C.-I. Cira, M.-Á. Manso-Callejo, R. Alcarria, B. Bordel Sánchez, and J. González Matesanz. State-level mapping of the road transport network from aerial orthophotography: An end-to-end road extraction solution based on deep learning models trained for recognition, semantic segmentation and post-processing with conditional generative learning. *Remote Sensing* 15(8):2099. 2023. doi:10.3390/rs15082099.
- [9] L. Dai, G. Zhang, and R. Zhang. RADANet: Road augmented deformable attention network for road extraction from complex high-resolution remote-sensing images. *IEEE Transactions on Geoscience and Remote Sensing* 61:1–13. 2023. doi:10.1109/TGRS.2023.3237561.
- [10] A. Fakhri and R. Shah-Hosseini. Improved road detection algorithm based on fusion of deep convolutional neural networks and random forest classifier on VHR remotely-sensed images. *Journal of the Indian Society of Remote Sensing* 50(8):1409–1421. 2022. doi:10.1007/s12524-022-01532-9.
- [11] Google Research. Welcome to Colab. 2023. <https://research.google.com/colaboratory/>. Online Service [Accessed: 2023].
- [12] S. Hao, Y. Zhou, and Y. Guo. A brief survey on semantic segmentation with deep learning. *Neurocomputing* 406:302–321. 2020. doi:10.1016/j.neucom.2019.11.118.

- [13] X. Hu and H. Yang. DRU-net: a novel U-net for biomedical image segmentation. *IET Image Processing* 14(1):192–200. 2020. doi:10.1049/iet-ipr.2019.0025.
- [14] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In: *Proc. 32nd International Conference on Machine Learning*, vol. 37 of *Proceedings of Machine Learning Research*, pp. 448–456. PMLR. 2015. <https://proceedings.mlr.press/v37/ioffe15.html>.
- [15] V. M. Ionescu. CPU and GPU gray scale image conversion on mobile platforms. In: *Proc. 2017 9th International Conference on Electronics, Computers and Artificial Intelligence (ECAI)*, pp. 1–6. IEEE. 2017. doi:10.1109/ECAI.2017.8166501.
- [16] R. Jaturapitpornchai, M. Matsuoka, N. Kanemoto, S. Kuzuoka, R. Ito, et al. Newly built construction detection in SAR images using deep learning. *Remote Sensing* 11(12):1444. 2019. doi:10.3390/rs11121444.
- [17] B. Li, F. Wu, K. Q. Weinberger, and S. Belongie. Positional normalization. In: *Proc. 32th Int. Conf. Neural Information Processing Systems (NeurIPS)*, pp. 1622–1634. Curran Associates. 2019. <https://proceedings.neurips.cc/paper/2019/hash/6d0f846348a856321729a2f36734d1a7-Abstract.html>.
- [18] K. Li, M. Tan, D. Xiao, T. Yu, Y. Li, et al. Research on road extraction from high-resolution remote sensing images based on improved UNet++. *IEEE Access* 12:50300–50309. 2024. doi:10.1109/ACCESS.2024.3385540.
- [19] S. Lin, X. Yao, X. Liu, S. Wang, H.-M. Chen, et al. MS-AGAN: Road extraction via multi-scale information fusion and asymmetric generative adversarial networks from high-resolution remote sensing images under complex backgrounds. *Remote Sensing* 15(13):3367. 2023. doi:10.3390/rs15133367.
- [20] P. Micikevicius, S. Narang, J. Alben, G. Diamos, E. Elsen, et al. Mixed precision training. In: *Proc. 6th International Conference on Learning Representations (ICLR)*. 2018. <https://openreview.net/forum?id=r1gs9JgRZ>.
- [21] N. Narisetti, M. Henke, K. Neumann, F. Stolzenburg, T. Altmann, et al. Deep learning based greenhouse image segmentation and shoot phenotyping (deepshoot). *Frontiers in Plant Science* 13:906410. 2022. doi:10.3389/fpls.2022.906410.
- [22] R. O. Ogundokun, R. Maskeliunas, S. Misra, and R. Damaševičius. Improved CNN based on batch normalization and Adam optimizer. In: *Proc. Computational Science and Its Applications – ICCSA 2022 Workshops*, vol. 13381 of *Lecture Notes in Computer Science*, pp. 593–604. Springer. 2022. doi:10.1007/978-3-031-10548-7_43.
- [23] O. Ronneberger, P. Fischer, and T. Brox. U-Net: Convolutional networks for biomedical image segmentation. In: *Medical Image Computing and Computer-Assisted Intervention – Proc. MICCAI 2015: 18th International Conference*, vol. 9351 of *Lecture Notes in Computer Science*, pp. 234–241. Springer. 2015. doi:10.1007/978-3-319-24574-4_28.
- [24] A. Safonova, G. Ghazaryan, S. Stiller, M. Main-Knorn, C. Nendel, et al. Ten deep learning techniques to address small data problems with remote sensing. *International Journal of Applied Earth Observation and Geoinformation* 125:103569. 2023. doi:10.1016/j.jag.2023.103569.
- [25] T. Salimans and D. P. Kingma. Weight normalization: A simple reparameterization to accelerate training of deep neural networks. In: *Advances in Neural Information Processing Systems 29 (NIPS 2016)*. Curran Associates. 2016. <https://proceedings.neurips.cc/paper/2016/hash/ed265bc903a5a097f61d3ec064d96d2e-Abstract.html>.
- [26] C. Shorten and T. M. Khoshgoftaar. A survey on image data augmentation for deep learning. *Journal of Big Data* 6:60. 2019. doi:10.1186/s40537-019-0197-0.

- [27] F. Sultonov, J.-H. Park, S. Yun, D.-W. Lim, and J.-M. Kang. Mixer U-Net: An improved automatic road extraction from UAV imagery. *Applied Sciences* 12(4):1953. 2022. doi:10.3390/app12041953.
- [28] T. Tiwari and M. Saraswat. A new modified-unet deep learning model for semantic segmentation. *Multimedia Tools and Applications* 82(3):3605–3625. 2023. doi:10.1007/s11042-022-13230-2.
- [29] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Instance normalization: The missing ingredient for fast stylization. arXiv, arXiv:1607.08022. 2016. doi:10.48550/arXiv.1607.08022.
- [30] R. Wang, M. Cai, Z. Xia, and Z. Zhou. Remote sensing image road segmentation method integrating CNN-Transformer and UNet. *IEEE Access* 11:144446–144455. 2023. doi:10.1109/ACCESS.2023.3344797.
- [31] A. Wanto, A. P. Windarto, D. Hartama, and I. Parlina. Use of binary sigmoid function and linear identity in artificial neural networks for forecasting population density. *International Journal of Information System and Technology* 1(1):43–54. 2017. doi:10.30645/ijistech.v1i1.6.
- [32] G. Wieczorek, I. Antoniuk, M. Kruk, J. Kurek, A. Orłowski, et al. BCT Boost segmentation with U-net in Tensorflow. *Machine Graphics and Vision* 28(1/4):25–34. 2019. doi:10.22630/MGV.2019.28.1.3.
- [33] Y. Wu and K. He. Group normalization. In: *Computer Vision – Proc. ECCV 2018*, vol. 11217 of *Lecture Notes in Computer Science*, pp. 3–19. 2018. doi:10.1007/978-3-030-01261-8_1.
- [34] X. Yang, X. Li, Y. Ye, R. Y. Lau, X. Zhang, et al. Road detection and centerline extraction via deep recurrent convolutional neural network U-Net. *IEEE Transactions on Geoscience and Remote Sensing* 57(9):7209–7220. 2019. doi:10.1109/TGRS.2019.2912301.
- [35] Z. Yin, B. Wan, F. Yuan, X. Xia, and J. Shi. A deep normalization and convolutional neural network for image smoke detection. *Ieee Access* 5:18429–18438. 2017. doi:10.1109/ACCESS.2017.2747399.
- [36] S. Zagoruyko and N. Komodakis. Paying more attention to attention: Improving the performance of convolutional neural networks via attention transfer. In: *Proc. International Conference on Learning Representations (ICLR)*. 2017. https://openreview.net/forum?id=Sks9_ajex.
- [37] W. Zeng. Image data augmentation techniques based on deep learning: A survey. *Mathematical Biosciences and Engineering* 21(6):6190–6224. 2024. doi:10.3934/mbe.2024272.
- [38] X. Zhang, X. Han, C. Li, X. Tang, H. Zhou, et al. Aerial image road extraction based on an improved generative adversarial network. *Remote Sensing* 11(8):930. 2019. doi:10.3390/rs11080930.
- [39] H. Zhou, H. Kong, L. Wei, D. Creighton, and S. Nahavandi. Efficient road detection and tracking for unmanned aerial vehicle. *IEEE transactions on intelligent transportation systems* 16(1):297–309. 2014. doi:10.1109/TITS.2014.2331353.
- [40] H. Zhou and L. Wei. UAV dataset. <https://sites.google.com/site/hailingzhouwei/>. Unaccessible at present.
- [41] H. Zhou and L. Wei. UAV dataset. https://drive.google.com/file/d/1DiQBsm5wmN0fFNnZs6i7oG_SHEo6HLej/. Copy of [40].
- [42] X.-Y. Zhou, J. Sun, N. Ye, X. Lan, Q. Luo, et al. Batch group normalization. arXiv, arXiv:2012.02782. 2020. doi:10.48550/arXiv.2012.02782.



Rayene Doghmane received her Master's degree in Electronics, specializing in Networks and Multimedia, from Badji Mokhtar University, Annaba, Algeria, in 2015. She is currently a Ph.D. student at the LASA laboratory, University of Annaba. Her research interests include image processing, computer vision, machine learning and network systems.



Karima Boukari is a senior lecturer at the University of Badji Mokhtar Annaba in Algeria. She received her engineering degree in automation from the University of Badji Mokhtar Annaba in 1996 and her Ph.D. in 2009. She is currently a lecturer at the Faculty of Technology of the same university. She is also the director of the Laboratory of Study and Research in Instrumentation and Communication of Annaba (LERICA). Her main research interests are: image processing, biometrics, diagnosis and fault detection.

HUMAN IRIS CLASSIFICATION THROUGH HISTOGRAM OF ORIENTED GRADIENT FEATURES WITH VARIOUS DISTANCE METRICS

Arnab Mukherjee¹ , Md. Zahidul Islam²  and Lasker Ershad Ali^{3,*} 

¹*Department of Quantitative Sciences (Mathematics),*

International University of Business Agriculture and Technology, Dhaka, Bangladesh

²*School of Engineering, Design and Built Environment,*

Western Sydney University, Kingswood, Australia

³*Mathematics Discipline, Khulna University, Khulna, Bangladesh*

**Corresponding author: Lasker Ershad Ali (ershad@math.ku.ac.bd)*

Abstract Human iris classification remains an active research area in the fields of biometrics as well as computer vision. In iris biometrics, most of the visible or near-infrared (NIR) eye images suffer from multiple noise sources, and the dispersive spectrum changes hugely. These changes occur due to spattering, albedo, and spectrum absorbance selectively. However, accurate iris classification for distance images is still a challenging task. To solve it effectively, we propose a machine learning (ML)-based iris classification employing a dense feature extraction method with various distance metrics. More specifically, this learning model focuses on the Histogram of Oriented Gradients (HOG) descriptor and K-Nearest Neighbour (K-NN) classifier with various distance metrics. The HOG descriptor has some advantages for this proposed distant-based iris classification, for example, insensitive to multiple lighting and noises, shift invariance, capacity to tolerate iris variations within the classes, etc. Additionally, this study investigates the most reliable distance metric that is less affected by different levels of noise. A publicly accessible CASIA-V4 distance image database is conducted for the experimental evaluation. To evaluate the performance of the classification models, we consider different measures such as recall, precision, F_1 -score, and accuracy. The reported results are tabulated as well as optimized through Receiver Operating Characteristic (ROC) curves. The experimental results demonstrate that the Canberra distance metric with low dimensional HOG features provides better recognition accuracy (90.55%) compared to other distance metrics.

Keywords: iris classification, image gradient, Histogram of Oriented Gradient features, distance metrics, confusion matrix, ROC curves.

1. Introduction

Iris recognition is a cutting-edge biometric technique that recognizes or confirms the identity of a person swiftly and efficiently by performing a set of mathematical operators on the stored biometric characteristics. Besides, physical contact is completely absent here to isolate iris images and analyze their patterns because this identification process is completely non-invasive. As a result, the demands for reliable security in offline and online authentications are constantly growing. In our networked society, biometric technologies have a variety of applications namely, ATM card authentication, e-commerce,

banking, access control to restricted zones, border-crossing, access to control computers, database access control in distributed systems, verification of suspects in crowds at airports and stations, identification of missing children, law enforcement activities and so on [34]. Without specific tools or automatic machine-learning techniques, it is very tough for a human operator to maintain high-security surveillance in these cases at a distance.

Nowadays the security fields follow different types of technologies to verify individual identities. Token-based and knowledge-based methods are two traditional ways of identifying an individual. The knowledge-based identifiers like personal identification numbers (PINs), usernames, and passwords can be forgotten or guessed by a third party. The token-based identifiers, for example, driver's licenses, passports, smart cards, ATM cards, and identification cards may be stolen or lost [39]. Recently, several studies have demonstrated that biometric traits are the most reliable and accurate authentication systems than conventional knowledge-based and token-based techniques. Even if it cannot be forgotten, stolen, or borrowed, and practically, forging is not possible. Among various physical traits, the iris has more advantages over other fingerprints, faces, eyes, ears, retina, DNA, palm print recognition, etc. [38]. Since iris is an externally visible internal organ that is highly protected from varied environmental conditions. It has unique patterns for an individual that are not related to any genetic factor. Iris texture has a high degree of randomness and individuality and remains unchanged from the age of three through the whole life, which is observed through the clinical evidence in [16]. In real life, there is no chance of a person having the right and left iris patterns or identical twins, or even two human iris textures being identical [38]. The above characteristics make it a promising biometric trait to verify and authenticate individual identities. However, this biometric technology has only been utilized in highly secure applications for government and civil society because of real-time constraints. Though the iris is a visible biometric characteristic like an eye, face, or finger, it is not as easy to recognize as those recognitions due to environmental conditions. The camera distances make it more challenging to capture clear iris texture during eye image acquisition. Capturing eye images in controlled or under less controlled conditions affects the quality of iris images greatly because of uncontrolled light sources. Under less controlled environments, the eye images captured at long distances with near-infrared imaging have multiple sources of noises, for instance, partial eye images, eyelids, glasses, eyelashes, defocus blur, etc., as illustrated in the Fig. 1. These types of noises demote drastically the image quality, pose difficulties in depicting distinct iris textures, and influence the further stages. In a controlled environment, the quality of eye images is high due to capturing at a close distance.

The previous attempts were only focused on close-distant images with controlled environments and global feature descriptors like wavelet filters. Those descriptors cannot extract the below-mentioned enormous iris patterns. Even, the local intensity color

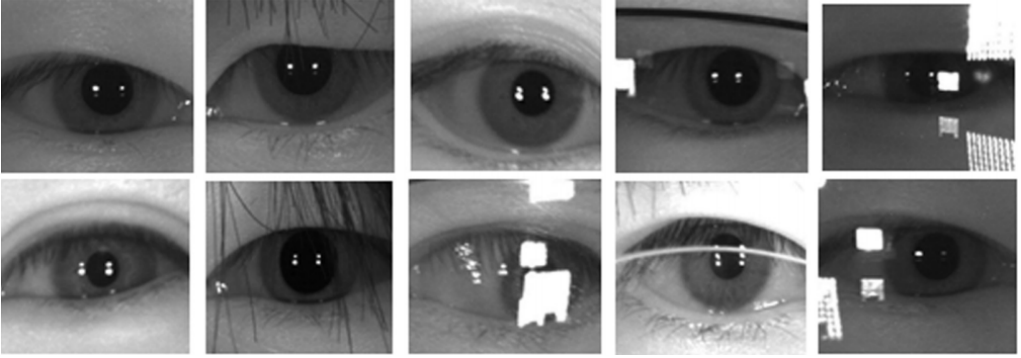


Fig. 1. Long-range eye images from CASIA-v4 database.

density, and pixel position of an image are not utilized there. While iris textures provide high distinctiveness, freckles, wrinkles, a variety of colors, zigzag patterns, etc. [3]. In reality, these features of various forms like textural, structural, and statistical features are highly required to recognize an individual at a distance. This work emphasizes block-based representation of local image contrast to overcome the limitations of wavelet filters. Herein, image gradient characterizes the structure or shape of iris patterns using local intensity gradient distributions and edge detection. More specially, the gradient features pool the edge orientations into small spatial regions to retrieve both micro-structures and macro-structures of iris patterns.

To consider the iris's textural characteristics and imaging conditions, this paper proposes a Histogram of Oriented Gradient feature descriptors to retrieve the spatial characteristics strategically from the local illumination variations of an iris image. The next stage is to explore a classifier that can recognize swiftly the iris features, which have the lowest implications on a variety of noises as well as enhance the classification performance. The classification stage perfect for a distance-based classifier because of being our experimental database imbalance [21,35]. The instances of minority subjects/classes are often sparse and scattered in imbalanced datasets and the majority subjects dominate the feature space. Consequently, misclassifications may occur as a result of higher distances between the instances of minority subjects and lower distances between the majority subjects. To address this issue, it is urgent to investigate deeply the influence of choosing various distance metrics during the classification of a large number of real-time images. For example, Euclidean distance is effective for numerical features like weight, height, salary, etc., that have equal importance over the continuous feature space. While Manhattan distance works effectively with categorical or binary feature points like DNA sequences. Manhattan distance is less affected by outliers compared to Euclidean distance. In this perspective, the mix of categorical and numerical features for

each type can be handled by the K-Nearest Neighbour classifier (K-NN) with a weighted distance metric. A perfect distance metric aids in the learning of the input iris patterns by computing the similarity between iris images and concluding informative decisions. The primary objective of the work is to consistently overcome existing shortcomings and find a supervised algorithm for remote iris recognition in less constrained environments.

The following sections organize the rest of this work: Section 2 reviews a few recent studies on the iris. Section 3 designs the architectural diagram of the iris classification approach. Experimental settings and evaluations are done sequentially in section 4. A statistical analysis is given graphically in section 5. Lastly, a brief conclusion is provided in section 6.

2. Related works

This section reviews a few recent research findings that are very similar to image gradient and distance-based iris recognition. The integrated stages of effective iris recognition are eye image capturing, iris region segmentation, normalization, feature extraction, classification, and iris recognition. The initial stage of eye image acquisition faces various challenges, for example, low resolution, off-axis, blur, motion, occlusion, and specular reflections in real-time environments and degrade the further processes [26]. Specialized, hybrid, and deep learning methods are enlisted to address these challenges [2]. Specialized methods use prior information about iris shape like annular iris/ elliptical shape, and dark area of the pupil. The iris trait features are identified unambiguously in the iris image [11, 15, 43]. These methods are fast without training images and effective for high-quality constrained iris images but not for completely unconstrained irises. Hybrid methods combine the specialized methods with ML algorithms to enhance iris segmentation performance [19, 32, 33, 42]. The ML algorithms produce coarse segmentation, and then a specialized approach is employed to generate the desired segmentation. The hybrid method adjusts these ML algorithms relying on the training process and the conditions of iris images. They can be more accurate due to employing iris priors and ground truth but not fast like specialized methods. However, these approaches are not enough to meet the challenges that arise in unconstrained iris images as still those are heavily dependent on iris priors. Recently, deep learning methods [2, 36, 47], follow semantic segmentation methods to alleviate the influence of unreliable iris priors. These methods provide more accuracy compared to previous methods as well as do not need to handcrafted features. Typically, they are slower due to tuning more parameters, required large-scale training data, and high computational cost.

The earlier approaches were developed based on wavelet filters and distance metrics for iris recognition. The drawbacks of these works are to ensure equal good-quality eye images from constraint environments [3]. Also, the recognition performance reduces significantly due to noisy artifacts, visible iris images, uncontrolled light sources, etc.

Pourreza et al. [7]. noticed that most of the wavelet transforms cannot extract spatial information practically and introduced contourlet transform to address such issues.

Tan and Kumar devised several approaches in [22,23,43,44] to deal with the existing problems for both visible and near-infrared imaging. Among them, the integration of integrating fragile bit [18] and weight map methods [12] through a weighted sum technique obtained the highest accuracy of 93.8% on the CASIA-V4 distance database.

Li et al. provided a weighted histogram of co-occurrence phases to extract the characteristics of local iris texture [25]. Bhattacharyya distance matched these distinctive and insensitive phase histograms with varying levels of illumination and noises. To overcome the challenge of matching low-resolution probe iris images with high-resolution enrolled iris images, Liu et al. developed a metric learning system [27]. The process has been carried out by learning the Mahalanobis distance and measuring appropriate pairwise similarities on the training set to minimize the divergence between the learned matching results and ideal matching outcomes.

The above-mentioned methods lost distinctive information on iris images due to environmental challenges and iris texture deformation. Ali et al. modified the contrast-limited adaptive histogram to alleviate the loss of information that helps to retrieve informative characteristics with speeded-up robust feature descriptor [4]. The proposed SURF-based algorithm achieved 99% and 99.5% recognition accuracy for left and right irises respectively using the CASIA-V4 distance database. Additionally, they noticed that fusion rule selection influences the classification performance at a certain level. The prior wavelet descriptor cannot account for singularities along lines or curves. To capture two-dimensional singularities, Ali et al. designed a feature-level fusion that concatenates the gradient, contourlet, Log-Gabor wavelet, and deep features with equal dimension [5]. The simple feature concatenation shows robustness against different physical challenges. To get over the drawbacks of wavelet and contourlet transforms, Ali et al. developed the Log-Gabor wavelet-based contourlet transform [6]. The merged descriptor extracts the edge and texture information in a variety of directions more compactly than the Log-Gabor or contourlet transform. The concept of remote iris recognition was first presented by Fancourt et al. for high surveillance. The eye images were captured at a 10-meter distance from the acquisition device and obtained an accuracy of 95-100% taking only 50 iris images [14]. Umer et al. retrieved the coarse and spatial properties of iris texture patterns efficiently using textural edges descriptors [46]. The recognition rate of the linear support vector machine (SVM) was 95% with k-fold cross-validation on the UBIRIS.v1 dataset.

Most of the feature-level fusions cannot integrate the discriminative iris patterns efficaciously with optimizing fused features of multimodal approaches due to a lack of homogeneity, adaption, and flexibility. To address these problems, Zhang et al. adopted an adaptive weighted sum method to concatenate the periocular and iris features for enhancing recognition performance [48]. In real-life situations, it is automatically learned

by neural networks to find out the optimum weights of iris-periocular fusion. Severo et al. proposed an approach that can encompass the iris region as the delimitation of the smallest squared bounding box [40]. This approach retrieves firstly multi-scale features of the iris and then a multi-SVM classifier utilizes the concatenation of HOG and cell mean intensity features. Recently, a few authors have focused on different types of Fourier transform (FT) for enhancement, analysis, restoration, and compression. FT decomposed the iris image into its sine and cosine components, which are considered as features. The authors in [17] evaluate the effects of applying principal component analysis (PCA) on FT except for accurate iris segmentation and feature properties using three distance metrics. Among the distance metrics, Manhattan distance achieved the highest accuracies of 96% and 94% for FT and PCA approaches using only 300 iris images of 50 persons from the CASIA-v1.0 database [2018].

Tarhouni et al. integrated the Fourier histograms of uniform local binary patterns (LBP) and pyramid histograms of gradient magnitudes through PCA [13]. The experiments show a promising result for challenging the CASIA-v4 database by mitigating the effects of the noisy artifacts from multiple sources like reflections, illumination variations, obstacles, and so on. Szymkowski adopted discrete fast FT components selected by PCA to describe iris texture [41]. The database was composed of 510 iris images from CASIA-IrisV4 and the reported average accuracies were 82.8% for K-NN, 86.6% for SVM, and 78.7% for ANN classifiers. The drawbacks of all the Fourier transforms are sensitivity to noise, boundary effects, and computationally intensive, especially for multi-dimensional images, which lead to retrieving inaccurate iris features. Arnab et al. introduced the local adaptive threshold method and k means clustering based color image segmentation to consider background clutter, changes in scale, partial occlusions, illumination, and color variation, which are common phenomenons for distant images [20, 28]. The author also developed a human identification scheme using an oriented autocorrelation feature descriptor and correlation distance classifier. The method performs effectively for distant captured iris images with losing shift-invariance but shows robustness against various noisy artifacts, rotation, occlusion, and illumination variation challenges [31].

As discussed above, remote iris classification is still very challenging for visible and near-infrared imaging at a distance in the fields of biometrics. The aim of combining HOG and K-NN with various distances is to find an appropriate balance that can overcome those shortcomings. Hope, this work contributes to selecting the perfect distance metric for further studies depending on the specific characteristics of data points, especially distance-based algorithms. This work is motivated by the computational simplicity of HOG descriptor [10], the robustness of various distance metrics in [29, 31], and the gradient strengthens against local illumination changes, etc.

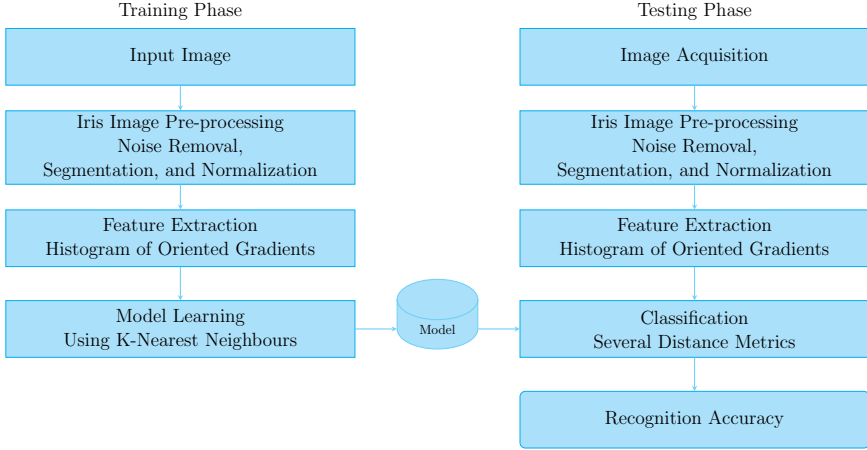


Fig. 2. The proposed architectural framework.

3. Methodology

This section provides a summary of the suggested machine learning algorithm, which integrates the HOG and distance classifier. Firstly, a reflection removal technique, a single-scale retinex algorithm is adopted to suppress the influence of different reflections. Secondly, the annular iris is separated using a random walker scheme from an eye image with low computational complexity. Thirdly, the segmented iris image is remapped by Daugman's rubber sheet model into a fixed dimension i.e., iris normalization to make a direct comparison between the iris images. Then, the images are sent to the automated HOG descriptor to extract distinctive iris patterns as feature vectors. Finally, the recognition accuracy is attained using the extracted feature vectors as input to the distance classifier. In the training phase, the model learns to train from the training images, and its performance is measured in the testing phase using test images. The architectural framework depicts the iris recognition system consecutively in Fig. 2.

3.1. Image pre-processing

Non-uniform illumination is a familiar phenomenon for distantly acquired eye images in real environments caused by uncontrolled light sources and multiple sources of noise. These certain noises create obstacles to separate accurate iris from the eye images. So, iris image pre-processing is required to address these issues. We adopt a single-scale retinex algorithm to improve image quality through high dynamic range compression [42].

Mathematically, the algorithm can be expressed as the following equation.

$$R_{\text{Im}}(p, q) = \log \frac{\text{Im}(p, q)}{G_{\tau} * \text{Im}(p, q)} . \quad (1)$$

Here, $\text{Im}(p, q)$ is a grey scale eye image, $G_{\tau}(p, q) = C \exp[-(p^2 + q^2)/\tau^2]$ is a Gaussian kernel, “*” denotes the convolution operator and $\tau = 1.5$ refers to the standard deviation.

3.2. Iris segmentation

Iris segmentation refers to the scheme of iris localization and separation automatically from the eye images. Due to poor segmentation, the feature descriptor fails to extract iris textures from the less discriminative regions which leads to incorrect iris recognition. The further stages such as feature extraction, classification, and recognition intricately rely on the quality of iris segmentation. To consider those issues, a graph theory-based random walker algorithm is employed to obtain the coarsely segmented binary iris masks in this work. The binary iris masks from the coarse iris segmentation are utilized to know detailed information about the estimation of iris center. The initial center of the iris and pupil is fixed using both the iris image and the corresponding binary mask. After that, the flash-points of papillary and limbic are approximately located with the help of a circular model. The iris segmentation stage is finished after eliminating occlusion noises. To understand more about the random walker segmentation algorithm deeply, the work in [6, 43] might be seen at a glance.

3.3. Iris normalization

The size and shape of the irises may change due to varying imaging distances and rotation of the acquisition device or eye. Illumination variation is also the cause of iris contraction or dilation. So, it is more conducive to removing the dimensional inconsistencies for matching two irises. Once a segment iris image is obtained, we follow the most commonly used Daugman’s rubber sheet model to make up elastic deformation of iris textures [11]. It is performed by re-mapping every pixel $\text{Im}(p, q)$ of the iris region from raw cartesian coordinates (p, q) to a pairwise non-concentric polar coordinates (r, θ) i.e., $r \in [0, 1]$ and $\theta \in [0, 2\pi]$. Mathematically, the re-mapping process may be expressed as:

$$\left. \begin{aligned} \text{Im}(p(r, \theta), q(r, \theta)) &\rightarrow \text{Im}(r, \theta) \\ p(r, \theta) &= (1 - r)p_{\text{pu}}(\theta)rp_{\text{bp}}(\theta) \\ q(r, \theta) &= (1 - r)q_{\text{pu}}(\theta)r q_{\text{bp}}(\theta) \end{aligned} \right\} \quad (2)$$

where $\text{Im}(p, q)$ represents the intensity value of the iris region image at each point (p, q) . The parameters $p(r, \theta)$ and $q(r, \theta)$ denote the co-ordinates of pupil $(p_{\text{pu}}(\theta), q_{\text{pu}}(\theta))$ and iris boundary points $(p_{\text{bp}}(\theta), q_{\text{bp}}(\theta))$ along the θ direction. The outcomes of noise removal from eye images, iris separation, and normalization are illustrated sequentially in Fig. 3.

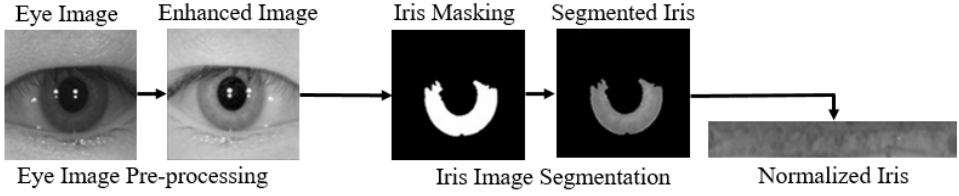


Fig. 3. Iris image pre-processing flow chart.

3.4. Gradient feature extraction

The researchers of INRIA (French National Institute for Research in Computer Science and Automation), Dalal, and Triggs devised the Histograms of Oriented Gradients feature descriptor [10]. This descriptor has been derived from scale-invariant feature transforms, and also parallels edge orientation histograms as well as shape contexts. The details of local object shape and appearance by capturing the edge or gradient structure using regional intensity variations are the primary purpose of this descriptor. Practically, this is done by splitting the localized image into blocks (grid) and each block is subdivided into smaller connected cells. All the locally oriented gradients within the cell are reshaped into a cell histogram. The cell histograms must be locally normalized within a block to account for the variations in illumination and contrast.

In addition, it does not change with photometric and geometric transformations because of local cell operation. The HOG feature scheme follows the steps in a local portion of an image to count the occurrences of oriented gradients.

Step 1: Gradient computation

In an image, the gradient strengths and orientations rely on the local properties of each pixel i.e., directional sub-divided color or intensity. The gradient values are computed along the vertical and horizontal directions by convolving the input image $Im(p, q)$ with 1D centered point discrete derivative masks $D_p = [1 \ 0 \ -1]$ and $D_q = [1 \ 0 \ -1]^T$. If the horizontal and vertical gradients are $G_p(p, q) = Im(p, q) * D_p$ and $G_q(p, q) = Im(p, q) * D_q$, respectively, the gradient magnitude M_G and orientation θ_G will be computed at the point (p, q) as follows:

$$M_G(p, q) = \sqrt{(G_p(p, q))^2 + (G_q(p, q))^2}, \quad (3)$$

$$\theta_G = \tan^{-1}(G_p(p, q)/G_q(p, q)) = \tan^{-1} \left(\frac{\partial G}{\partial q} / \frac{\partial G}{\partial p} \right). \quad (4)$$

Step 2: Orientation binning

The cell histogram is to be constituted in step 2. The 1D histogram is constructed by reshaping the local gradient orientations over the pixels of a cell into angular bins

(ranging from 0 to 360°). The local orientations are assigned to the nearest bins by voting weights for each pixel over the local spatial region, i.e., a cell. Then, the gradient orientations of all pixels $\text{Im}(p, q)$ in a cell α are distributed into N bins. The gradient magnitudes with a gradient angle of $\Delta\theta$ degree are accumulated in the respective bin h_l , which denotes the heights of the bins. Finally, the discretization of orientations into N bins each of $\Delta\theta$ degrees constructs the 1D histogram H_i as follows:

$$H_i = [h_l]_{l=1}^N = \sum_{\text{Im}(p,q) \in \alpha} M_G(p, q); \theta_G(p, q) \in \Delta\theta, \quad (5)$$

where $\Delta\theta = 360^\circ/N$.

Step 3: Block descriptor

Step 3 requires grouping the cell histograms into larger spatial connected blocks F_i .

Step 4: Block normalization

The cell histograms are to be locally normalized within a block for counting the variations in illumination, and contrast [10]. As each block is composed of a group of cells, a cell may be contained in various block normalizations for the overlapping block.

Step 5: Concatenation of histogram features

Finally, the histogram of oriented feature vectors ϑ is constituted by integrating cell histograms across from all the normalized blocks in a sliding window, which represents a one-dimensional array of histograms.

To obtain fixed feature dimensions, the input images must be resized with 64×64 pixels. The size of blocks is set to 2×2 cells and every single cell consists of 8×8 pixels. 9 orientation bins between -180° and $+180^\circ$ (signed gradients) are used to construct histogram bins so that the HOG features can be organized sequentially according to their properties. A total of 49 blocks is computed for an image with 64×64 pixels. The final HOG feature descriptor is formulated as follows:

$$\vartheta = [F_1, F_2, \dots, F_i, \dots, F_{36}], \quad (6)$$

where ϑ denotes the HOG feature descriptor, and F_i is the normalized block vector in i th block. Every block has four cell histograms with nine bins, $F_i = [\bar{h}_{1,i}, \bar{h}_{2,i}, \dots, \bar{h}_{i,j}, \dots, \bar{h}_{36,i}]$, where $\bar{h}_{i,j}$ is the j th normalized value of i th block. The flow diagram of the HOG feature extraction is given systematically in Fig. 4.

3.5. K-Nearest Neighbour classification

K-Nearest Neighbour (K-NN) is a distance-based supervised machine learning algorithm that performs pattern recognition tasks for classifying objects based on various features. It ensures better performance in bio-informatics, data mining, and image classification

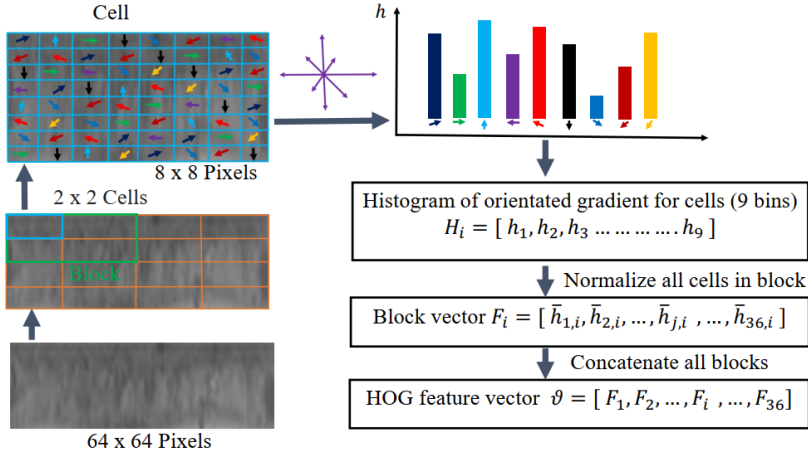


Fig. 4. The flow diagram of HOG feature extraction.

when the features are labeled prior with low dimensionality as well as scaled in equal weight. The main advantage over the other classifiers is that there's no need for pre-training, the model does not learn in the training phase, even needs not to tune more parameters. It also assumes that similar things exist nearby and classifies the test iris images based on the similarity measure of prior stored feature vectors. To supervise the HOG model for such properties, the K-NN algorithm compels us to follow instead of the other classification methods. Besides, K-NN functions as an outlier detector by locating feature points that have few or no neighbours within a fixed radius.

The review section 2 shows that the Euclidean and Hamming distances are used widely in classification problems but in most of the cases, accurate predictions depend on feature properties, distance metrics, and so on. The following metrics are utilized to compare and measure the distances between test and training images [29] as there is no comparative study of distance metrics. Let $p = (p_1, p_2, p_3, \dots, p_n)$ and $q = (q_1, q_2, q_3, \dots, q_n) \in R^n$ be the two feature vectors in n -dimensional space. The distance measure between p and q vectors may be defined as

Euclidean distance (Eucl) It is a straight line distance, which represents the sum of the squared differences of two attribute vectors by taking the square root.

$$D_{\text{Eucl}}(p, q) = \sqrt{\sum_{j=1}^n (p_j - q_j)^2}, \quad (7)$$

where p_j is the j th component in the vector p and q_j is the j th component in the vector q .

Sokalmichener distance (Soka) The distance computes the Sokal-Michener dissimilarity between Boolean 1D arrays p and q . The Boolean array p is a sequence of numerics that consists of 0 (false) and 1 (true) and has no intermediate values. The Boolean 1D arrays with a threshold value and Sokalmichener distance are defined as

$$p = \begin{cases} 1 & \text{if } p_j \geq 0.1667, \\ 0 & \text{if } p_j < 0.1667, \end{cases} \quad (8)$$

$$D_{\text{Soka}}(p, q) = \frac{2(m_{10} + m_{01})}{m_{11} + m_{00} + 2(m_{10} + m_{01})}. \quad (9)$$

Here, m_{xy} counts the number of occurrences of $p_j = x$ and $q_j = y$ for $j < n$, and n defines the total number of points in a feature vector p .

Yule distance (Yule) Yule distance is a measure of dissimilarity between two probability distributions based on their overlap. The Yule dissimilarity is defined as

$$D_{\text{Yule}}(p, q) = \frac{2m_{10}m_{01}}{m_{11}m_{00} + m_{10}m_{01}}. \quad (10)$$

Jaccard distance (Jacc) It is a statistical metric that is generated from the Jaccard index and measures the diversity of iris patterns. The Jaccard dissimilarity between boolean 1D arrays p and q is defined as

$$D_{\text{Jacc}}(p, q) = \frac{m_{10} + m_{01}}{m_{11} + m_{10} + m_{01}}. \quad (11)$$

Dice distance (Dice) The distance is close to the Jaccard index which measures the dissimilarity of two patterns. The Dice dissimilarity between p and q iris vectors is

$$D_{\text{Dice}}(p, q) = \frac{m_{10} + m_{01}}{2m_{11} + m_{10} + m_{01}}. \quad (12)$$

Bray-Curtis distance (Bray) The Braycurtis distance is the absolute differences between two attribute vectors with taking the summation, which differences are divided by their summed attribute values.

$$D_{\text{Bray}}(p, q) = \frac{\sum_{j=1}^n |p_j - q_j|}{\sum_{j=1}^n |p_j| + \sum_{j=1}^n |q_j|}. \quad (13)$$

Canberra distance (Canb) It is an extension of L_1 distance that includes weights and measures the dissimilarity of ranked lists. The sum of absolute differences is divided by their summation between a pair of points over a vector space.

$$D_{\text{Canb}}(p, q) = \sum_{j=1}^n \frac{|p_j - q_j|}{|p_j| + |q_j|}. \quad (14)$$

Manhattan distance (Manh) It is the sum of absolute differences of two opposite attributes over the normed vector space. It is most preferable for high dimensionality and provides more reliable results due to the absolute value of distance.

$$D_{\text{Manh}}(p, q) = \sum_{j=1}^n |p_j - q_j|. \quad (15)$$

Cosine distance (Cosi) This distance measures the angle between two vectors of inner product space with magnitude and is computed from one minus the cosine of the included angle between two attribute vectors.

$$D_{\text{Cosi}}(p, q) = 1 - \frac{\sum_{j=1}^n p_j q_j}{\sqrt{\sum_{j=1}^n p_j^2 \sum_{j=1}^n q_j^2}}, \quad (16)$$

where p_j is the j th value in the vector p and q_j is the j th value in the vector q .

Correlation distance (Corr) Between two feature vectors, the linear relationship is measured in this case by subtracting Pearson's correlation coefficient from one.

$$D_{\text{Corr}}(p, q) = 1 - \frac{\sum_{j=1}^n (p_j - \bar{p})(q_j - \bar{q})}{\sqrt{\sum_{j=1}^n (p_j - \bar{p})^2 \sum_{j=1}^n (q_j - \bar{q})^2}} \quad (17)$$

where $\bar{p}$ is the mean of a feature vector p , i.e., $\bar{p} = \frac{1}{n} \sum_{j=1}^n p_j$ and $\bar{q}$ is the mean of a feature vector q , i.e., $\bar{q} = \frac{1}{n} \sum_{j=1}^n q_j$.

4. Database and experimental setup

In this section, a publicly accessible database of remotely captured face images, CASIA-v4 is employed to conduct all of the experiments. The Chinese Academy of Science's Institute of Automation (CASIA), Beijing, China has provided the database to explore iris-based biometric recognition [45]. The facial images are acquired remotely with the help of near-infrared cameras. The distance is three meters away from the subjects under less controlled environments. The full database comprises 142 subjects including 2567 facial images. First, all right and left eyes were separated from the face images and a total of 5134 eye images were obtained. After that, the imbalanced eye images are categorized into 142 subjects i.e., each subject has not an equal number of images, and most of the subjects include regular-irregular images. From the facial images, all the images of the eye cannot locate or isolate exactly owing to having obstruction of glasses and occlusion of eyelids or eyelashes. We only use the regular iris images of the first 14 subjects for parameter tuning and explore that Canberra distance performs the best. Further experiments are conducted using this distance metric. We selected randomly

3233 images as training data from each of the subjects 15-142 to learn the classification model. Similarly, we select 742 images to evaluate the performance of the model as testing data. All the experiments are conducted by dint of Python 3.7 and MATLAB R2018a (Intel Core i5).

K-NN classifier treats all features equally by default to contribute to the distance calculations. Otherwise, the distance measurement will be dominated by the larger feature points if they are on different scales and ranges. Min-max normalization helps to address this issue without distorting the larger differences of those features. Additionally, weighted distances assign higher weights to privilege the important features. The histogram-oriented gradient features V including training and test features from (6) are normalized with the help of the following equation.

$$V = (\vartheta_j - \vartheta_{min}) / (\vartheta_{max} - \vartheta_{min}), \quad (18)$$

where ϑ_j denotes the j^{th} value of feature vector ϑ with maximum value ϑ_{max} and minimum value ϑ_{min} . Challenges such as data sparsity, distance loss of meaning, overfitting, and increased computation cost arise in high dimensional spaces. Therefore, it is more convenient to lessen the dimensionality of HOG features using PCA without loss of useful information. For this purpose, the PCA transform matrix is obtained from training features and then utilized in test features. These low dimensional, labeled, and scaling features help to train the K-NN model effectively, otherwise affect the majority voting in classification.

4.1. Performance measure

The effectiveness of a classification model may be measured using a variety of assessment indicators. The most often used confusion matrix is utilized to determine the model's accuracy and correctness. Accuracy measures the effectiveness of the classifier by its percentage of samples classified accurately. Classification accuracy is defined by

$$\text{Accuracy} = \frac{\text{Number of accurate classified samples}}{\text{Number of all samples}}. \quad (19)$$

The evaluation systems are designed by calculating the following measures to assess the recognition performance within each class of the database.

- (i) True positives (tp): number of positive (p) predictions that are true (t).
- (ii) True negatives (tn): number of negative (n) predictions that are true (t).
- (iii) False positives (fp): number of positive (p) predictions that are false (f).
- (iv) False negatives (fn): number of negative (n) predictions that are false (f).

Tab. 1. Performance measurement for each distance metric.

Distance Metrics	Avg. Precision	Avg. Recall	F_1 -measure	Accuracy (%)
Euclidean distance	0.7931	0.7647	0.7543	75.70
Sokalmichener distance	0.8452	0.8311	0.8193	82.32
Braycurtis distance	0.8772	0.8535	0.8483	85.02
Canberra distance	0.9190	0.9102	0.9053	90.55
Manhattan distance	0.8564	0.8333	0.8262	82.86
Yule distance	0.8622	0.8397	0.8309	83.53
Jaccard distance	0.8508	0.8363	0.8250	83.13
Dice distance	0.8508	0.8363	0.8250	83.13
Cosine distance	0.7951	0.7623	0.7519	75.70
Correlation distance	0.7966	0.7605	0.7505	75.70

The average precision, recall, and F_1 -measure values and accuracy of a multi-class classification system are defined by

$$\text{Average Precision} = \frac{1}{N_c} \sum_{j=1}^{N_c} \frac{\text{tp}_j}{\text{tp}_j + \text{fp}_j}, \quad (20)$$

$$\text{Average Recall} = \frac{1}{N_c} \sum_{j=1}^{N_c} \frac{\text{tp}_j}{\text{tp}_j + \text{fn}_j}, \quad (21)$$

$$F_1\text{-measure} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}, \quad (22)$$

$$\text{Accuracy} = \frac{1}{N_c} \sum_{j=1}^{N_c} \frac{\text{tp}_j + \text{tn}_j}{\text{tp}_j + \text{tn}_j + \text{fp}_j + \text{fn}_j}, \quad (23)$$

where N_c is the number of classes; tp_j , fn_j , fp_j and tn_j are the number of true positive, false negative, false positive, and true negative classifications for class j , respectively. The confusion matrix helps to derive these performance measures, which are illustrated graphically in both predicted and actual classification [30] corresponding to their subjects or classes. The performance of dense HOG descriptor with K-NN classifier is computed for each distance metric by measuring average precision, recall, F_1 -measure, and classification accuracy as enlisted in Table 1.

We observe that Canberra distance metric provides the highest average precision (0.9190), recall (0.9102), F_1 -measure (0.9053), and overall classification accuracy (90.55%) among the 10 distance metrics from the Table 1 in case of using HOG features. It is visible from Fig. 5 that the iris feature vector consists of several criteria such as bi-

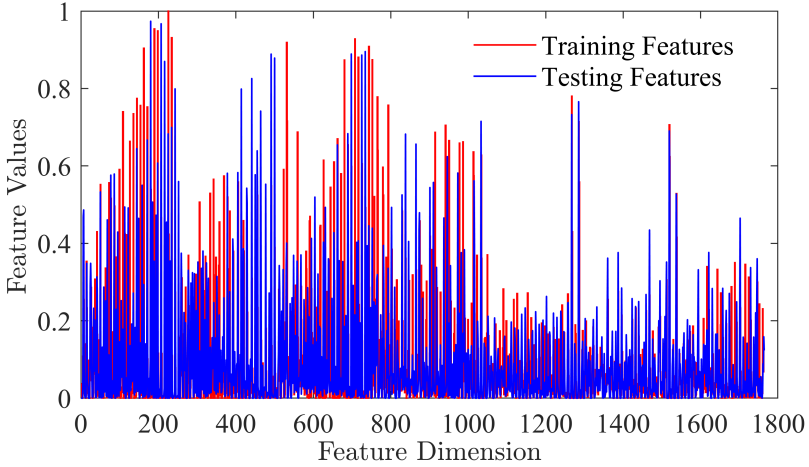


Fig. 5. The HOG feature distribution.

nary features – has any iris shape or not; ordered categorical features – iris shape very densely/iris shape moderately densely/iris shape not densely; numerical features – a measurement like color density in dpi. Canberra Distance utilizes the criteria to recognize the test images in their subjects according to how similar or dissimilar they are with training images. Also, the distance metric deals with mixed types of feature points and sorts the iris features into groups that are more closely or distantly related to each other. Due to appropriate balance, the HOG descriptor shows better performance with the weighted Canberra distance over the other distance metrics.

In addition, we have experimented with edge orientation histograms (EOH), contourlet transform (CT), and uniform gradient local binary patterns (GLBP) features to compare the discriminatory power of HOG features utilizing the Canberra distance metric. These experimental outcomes are enlisted in Table 2 in terms of precision, recall, F_1 -measure, and classification accuracy. It can be found from Table 2 that the Canberra distance-based classification provides the highest result for HOG features compared to the other three feature descriptors. Therefore, it is confident that the HOG descriptor locally extracted more relevant iris textures from the complicated images due to the orientation of the iris image gradient.

4.2. Performance study

This sub-section depicts the performances of feature descriptor as well as the impacts of various parameter selection for optimal classification. The Receiver Operating Characteristic (ROC) curves are plotted concerning false positive rate and true positive rate

Tab. 2. Efficacy of HOG with other feature descriptors.

Feature Descriptors	Avg. Precision	Avg. Recall	F_1 -measure	Accuracy (%)
EOH	0.7129	0.6175	0.6132	61.40
CT	0.7444	0.7107	0.6880	69.63
GLBP	0.8249	0.7818	0.7709	77.32
HOG	0.9190	0.9102	0.9053	90.55

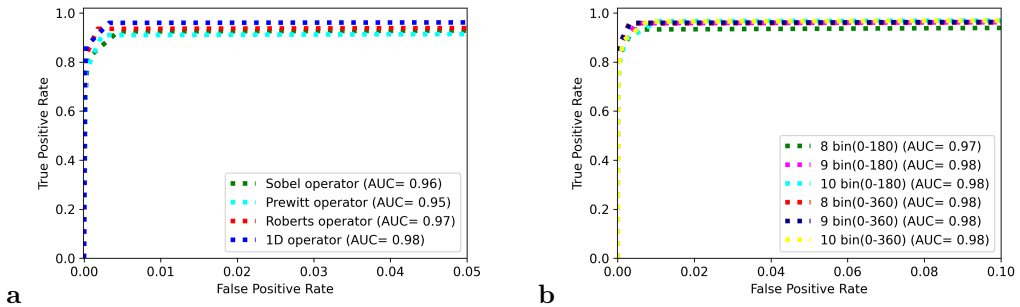


Fig. 6. True versus false positive rates for (a) gradient operators; (b) orientation bins.

with the help of classification threshold values. The performance is indicated by the closed curve to the top-left corner. The study explains the influence of HOG descriptor with the help of ROC curves graphically and confusion matrix from Fig. 6a to Fig. 8a, and obtain the optimal parameters to extract the gradient features like as 64×64 pixels sliding window, one-dimensional derivative masks $[1 \ 0 \ -1]$; 9 orientation bins between -180° and $+180^\circ$ (signed gradients); 2×2 pixel blocks of four 8×8 pixel cells and L_1 -sqrt normalization scheme respectively. The effects of chosen distances are shown not only graphically Figs. 8b, 9a but also numerically in Tables 1, 2.

Gradient computation

Image gradient computation is the first step of retrieving gradient features. Roberts, 1D centered derivatives $[1 \ 0 \ -1]$, Sobel and Prewitt operators are followed to compute image gradient. The one-dimensional derivatives perform best among those operators with the lowest computational cost. The 3×3 Prewitt and Sobel masks reduce the classification performance by around 4% and 5% compared to 1D derivatives. Whereas, the centered 2D derivative masks i.e., 2×2 diagonal Roberts's filter slightly improves the performance by 2% than 3×3 derivative masks. The performances are reduced significantly in Fig. 6a with increasing the size of derivative masks.

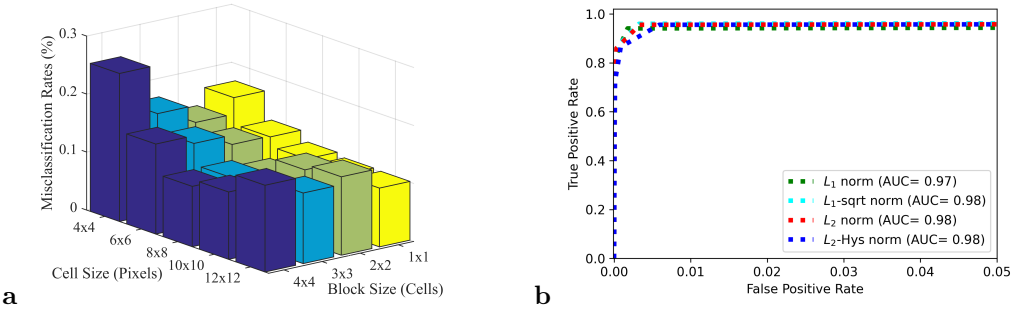


Fig. 7. (a) Misclassification rates. (b) True versus false positive rates for block normalization.

Orientation binning

In the next step, we compute the weighted votes for every pixel to form a histogram based on the oriented vector of iris texture patterns. Then, the computed votes are gathered into orientation bins as cells from the local spatial regions. Regarding rectangular cells, the orientation values of the gradient are evenly spaced in bins between 0° and $+180^\circ$ (unsigned gradients) or between -180° and $+180^\circ$ (signed gradients). The orientation bins of the gradient can be allocated into several bins. The number of orientation bins changes from 8 bins to 10 bins to visualize their performances and the signed gradients of 9 bins perform better among them as shown in Fig. 6b.

Blocks descriptor

The variations of gradient strength occur extensively due to illumination changes and local contrast. So, it is more convenient to find an effective local contrast normalization that contributes to integrating all the cells into larger spatial blocks. Each block normalizes the local contrast individually. We use square R-HOG blocks, which are formulated by $\zeta \times \zeta$ cells per block with $\alpha \times \alpha$ pixels per cell and each cell consists of β orientation bins per cell histogram with the ζ, α, β parameters.

A 3D bar graph is plotted to visualize the misclassification rate for cell size and block size in cells in Fig. 7a. The variation of block sizes is shown with the help of different colors. The 2×2 cell blocks of 8×8 pixel cells work best with a 9.45% misclassification rate among the block descriptors. The 1×1 , 3×3 and 4×4 blocks with their corresponding cells 6×6 , 8×8 perform also good but decrease performance around 1% compared to 2×2 blocks.

Block normalization

The block normalization techniques are adopted for better invariance to illumination and shadowing. If ϑ is the non-normalized feature vector that holds all the histogram of orientations in a block, the normalization schemes $\|\vartheta\|_b$ are defined as (a) L_2 -norm is the square root of the sum of squared values, i.e., $\|\vartheta\|_{b=2} = (\sum_{j=1}^n \vartheta_j^2)^{1/2}$ and L_2 -norm:

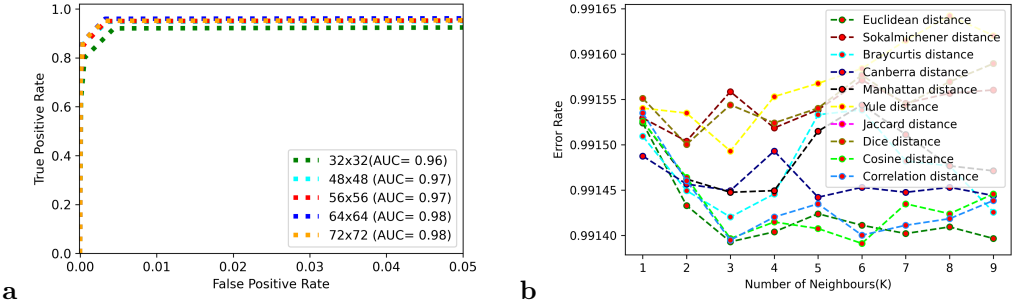


Fig. 8. True versus false positive rates for (a) sliding windows; (b) number of neighbours K.

$v \rightarrow \vartheta / (||\vartheta||_2 + \varepsilon)$; (b) L_2 -Hys: $v \rightarrow \vartheta / (||\vartheta||_2 + \varepsilon)$, $\vartheta \leq 0.2$; (c) L_1 -norm is the sum of absolute values i.e., $||\vartheta||_{b=1} = \sum_{j=1}^n |\vartheta_j|$ and L_1 -norm: $v \rightarrow \vartheta / (||\vartheta||_1 + \varepsilon)$ and (d) L_1 -sqrt: $v \rightarrow \vartheta / \sqrt{||\vartheta||_1 + \varepsilon}$. Figure 7b shows that L_1 -sqrt and L_2 -norm provide almost similar performance but L_2 -norm reduces performance by 0.27%. Whereas, L_1 - sqrt increases performance significantly by 1.76% and 1.08% than the L_2 -Hys and L_2 -norms. Extensive experiments are carried out to measure the optimal value of ε over a wide range and conclude that there are no more effects of the constant ε on overall results.

Sliding window

The variations of sliding windows pose difficulties in the feature extraction stage and influence the classification results greatly as the computational simplicity relies on sliding windows at large. Figure 8a reports that the performances of sliding window increase up to about 64×64 pixels but decrease performance with 72×72 pixels window. Among them, a window of 64×64 pixels produces a substantial amount of context to recognize iris patterns. The other sliding windows are seen the same on the ROC curves but their evaluation matrices are different. The sliding window 32×32 , 48×48 , 56×56 and 72×72 pixels reduces performance 7.42%, 1.35%, 1.49% and 0.27% respectively on the recognition accuracy due to a lack of sufficient contextual information.

Number of K neighbours

The K parameter refers to the number of nearest neighbours to be considered while making the prediction. This affects the sensitivity of the algorithm to local patterns in the feature space. A smaller K leads to low bias but increases the impact of outliers with complexity and makes the model more prone to overfitting. Whereas, the model arises with less complexity with a larger K, which assists in avoiding overfitting but ignores potential local patterns. The odd number of K defines always a majority class that helps to decide the predicted class. It is clear from Fig. 8b that the 3 nearest neighbours show better performance with the highest value by avoiding ties in voting. The figures of Jaccard and Dice distance overlap due to having similar properties that are assumed

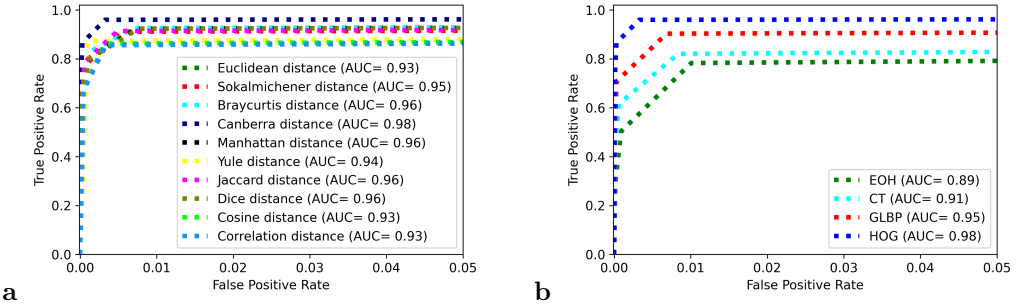


Fig. 9. True versus false positive rates for (a) distance metrics; (b) feature descriptors.

from the measurements in Table 1. It is also visible that there is no optimal value of K, which performs equally well for all the distances. Fig. 8b ensures that the choice of distance metric affects the classification performance significantly but not the selection of the right K in K-NN.

Distance metrics

In this study, the above-mentioned distance metrics are employed to investigate, which distance is less affected by the noise implications. We obtain the optimal performance with the highest outcome (90.55%), while the Canberra distance is applied to train the model instead of mostly using Euclidean and Hamming distance metrics.

The K-NN classifier with Euclidean, Cosine, and Correlation distances reduces performance by around 15% compared to Canberra distance. Although the accuracies of Jaccard and Dice distances are almost the same, the recognition performance is different as shown in Fig. 9a. Seemingly, the Yule distance shows poor performance on ROC curves but its accuracy is so much better than Manhattan, Sokalmichener, and Jaccard distances in Table 1. The comparative study suggests that Canberra distance may be an effective metric for gradient feature classification with the highest possible accuracy.

Feature descriptors

A feature descriptor is an algorithm that extracts only the most informative features of an object in terms of a set of numbers. We compare the performance of the dense descriptor HOG with EOH, CT, and GLBP feature descriptors using the Canberra distance-based classifier. These descriptors focus on the shape or structure of iris patterns as they utilize the magnitude and orientations of the gradient to extract features. They provide several computation costs because of extracting various feature properties and measuring the distance between the feature points. Among these descriptors, the HOG feature-based technique classifies the iris images more swiftly than others. The EOH descriptor reduces performance by around 19% consuming huge run time. It is visible from Fig. 9b that

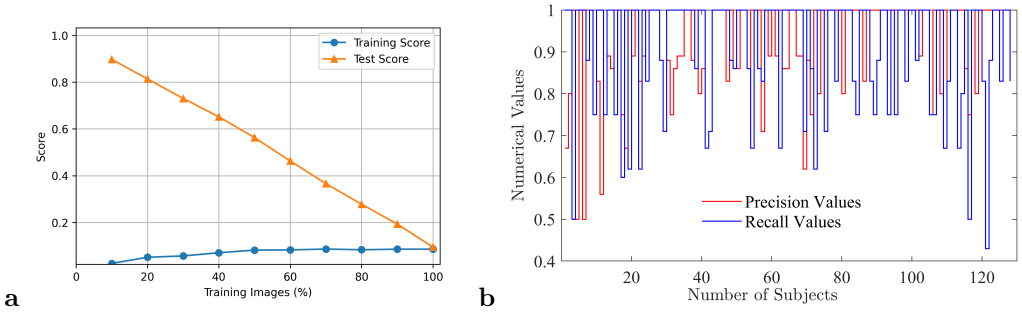


Fig. 10. (a) Plot of K-NN learning curves using HOG features. (b) Subject-based precision-recall curves.

there is a scarcity of threshold points to make the ROC curve smoother because of the imbalance of subjects in training and test sets.

5. Statistical analysis

We assess the performance of the HOG feature descriptor by dint of learning curves and precision-recall curves for every subject concerning their values that are accomplished by the K-NN classifier with Canberra distance. The learning curves show the robustness of the model as well as the scope of using a large number of images in real-life applications. The precision-recall curves interpret the types of noisy images and illustrate the complexity of images within each subject of a database.

In this study, the learning curves diagnose the model's learning and generalization behavior to make a marginal decision. The above yellow and blue learning curves are plotted by using the Canberra distance-based K-NN classifier with the histogram of oriented features. The test score (yellow curve) shows how well the model fits new data, whereas the training score (blue curve) shows how well the model fits the training set. In the beginning, the large gap between the training and test performance shows that the model is under-fitting, which is probably due to the small size of images and database from different distributions. With the increasing of training images, the curves are going to converse a satisfactory score as shown in Fig. 10a. The training scores are enhanced through the iterations of up to 50 percent of training images. After that, those scores are constant and the HOG model fails to obtain 100% accuracy in the training phase while the test scores are increasing and converse to the highest possible score in the end. The learning curves demonstrate that the testing scores could be made better by increasing training images and making the dataset balanced.

Figure 10b plots the precision and recall curves for each subject concerning precision and recall values. The blue color is used for the precision curve, while the red color is

Tab. 3. Performance evaluation with existing cutting-edge approach

Proposed Approaches	Accuracy
Symlet wavelet filter and Spearman distance [29]	80.00%
Principal Component Analysis and Braycurtis distance [37]	80.00%
Level set and local binary pattern with Manhattan distance [9]	81.45%
Discrete fast Fourier transform and Braycurtis distance [41]	82.80%
Radon Transform and Euclidean distance [8]	84.17%
Uniform LBP and Euclidean distance [24]	84.77%
LBP and Euclidean distance [38]	84.88%
GLCM and Euclidean distance [1]	85.00%
CNN feature descriptor and Euclidean distance [5]	86.94%
Contourlet Transform and Hamming distance [7]	88.00%
Histogram of Oriented Gradients and Canberra distance – Proposed	90.55%

used for the recall curve. We can see that 51 subjects in precision curves and 48 subjects in recall curves cannot attain the maximum values due to having a variety of obstacles such as eyelids, eyelashes, illumination, and internal eye variations in these subjects. The images of 46 subjects within 128 subjects and 672 images among the 742 test images are classified accurately, while 70 images of the rest subjects are not recognized by their corresponding subjects. For example, the 18th and 29th subjects have the lowest recall value of (0.50) and the 17th subject has the lowest precision value of (0.50), indicating that some but not all of the subjects' test images are recognized. In practice, the images of these 82 subjects are imbalanced and contain several noises during the acquisition process, which impede an illustration of clear iris textures. For weak segmentation of irises, the HOG descriptor cannot retrieve relevant features and learn robustly from those iris textures. Thus, the K-NN model fails to classify the complicated iris images accurately and does not obtain overall 100% recognition accuracy. The performance scores of the F_1 -measure are overlapping, which indicates that the feature distributions are irregular in these subjects. In the other subjects, the high success rates of evaluation metrics show that the models are successful in iris image classification.

Table 3 provides a comparison with earlier approaches including the reported results. Few of these techniques examined the performance of classification on various types of databases with various numbers of training and test images. For instance, Tan and Kumar only performed their experiments using the first 8 right or left eye images from the CASIA-v4 distance database [45]. The overall recognition accuracy was 93.90% employing training images from the first 10 subjects and test images from subjects 11-141. The recognition rate of Chan method was 90.43% considering 79 training images

from the 10 subjects and 961 test images from the remaining 131 subjects from the proposed database [23].

It is remarkably worthy that the authors in [5] reported several efficiencies in adopting multi-feature descriptors. The accuracy of 98.17% is the highest among the outcomes, however, their performances are not explicitly comparable to our work. Because the authors not only considered iris images but also included contextual eye images having pupil, eyelash, eyelid, sclera, and so on. Therefore, the experimental results of the proposed framework would not be feasible to make a comparison with other experimental outcomes directly. We employ 3975 images including 3233 images for training and 742 images for testing; and also do not consider the regular eye images of the first 14 subjects. It is clear from Fig. 1 that our experimental database consists of more complicated and non-linear images than others.

However, the supervised approach is better as compared to existing methods concerning near-infrared distance iris images having various illumination conditions and multiple sources of noise. Also, accurate iris segmentation, informative features with lower dimensionality, distance metric of lowest noise implications, and computational simplicity can be considered measurable parameters of good classification performance. Finally, the comparative studies validate that the Canberra distance metric may be applied in the lieu of most widely used Euclidean or Hamming distance metric for noisy datasets and distance-based approaches.

6. Conclusion and future work

This paper has introduced an image gradient and distance-based machine learning algorithm for remote iris recognition. The HOG descriptor captures intuitively the shape of structures in the region by capturing information about gradients. The discriminative power of HOG is to extract successively both microstructures and macro structures of iris patterns from the local contrast and illumination variations. To classify the imbalanced iris images, a weighted distance classifier is needed to explore which is less affected by different levels of noise. Like other classifiers, K-NN is prone to become biased towards the majority of instances of training features but can be handled effectively with the help of weighted Canberra distance. The distance metric emphasizes the larger differences between the iris features and outliers and is more robust to outliers than other distance metrics. The experimental evaluation demonstrates that Canberra distance provides the highest possible classification accuracy (90.55%) with the lowest noise implications.

The combination of HOG and K-NN classifier shows its robustness against local contrast, illumination changes, and occlusions. It is regarded as one of the most influential machine learning algorithms because all the parameters are intricately connected. Though the HOG descriptor extracts the iris features efficiently while retaining robustness to irrelevant variations resulting from environmental changes, it lost shift-invariance

and additivity. The concept of correlation will be adopted in self-similarity to address these issues. It will exploit the spatial and orientational auto-correlations from the local image gradient that prioritize the closer iris patterns in its local neighbourhoods. In the future, a proximity-weighted evidential K-NN classifier will be applied to give more priority to the instances of minority subjects or classes.

References

- [1] E. Acar. Extraction of texture features from local iris areas by GLCM and iris recognition system based on KNN. *European Journal of Technic*, 6(1):44–52, 2016. <https://api.semanticscholar.org/CorpusID:209081348>.
- [2] S. Ahmad and B. Fuller. Unconstrained iris segmentation using convolutional neural networks. In: *Computer Vision – Proc. 14th Asian Conference on Computer Vision (ACCV) 2018 Workshops*, vol. 11367 of *Lecture Notes in Computer Science*, pp. 450–466, 2019. doi:[10.1007/978-3-030-21074-8_36](https://doi.org/10.1007/978-3-030-21074-8_36).
- [3] A. S. Al-Waisy, R. Qahwaji, S. Ipson, S. Al-Fahdawi, and T. A. Nagem. A multi-biometric iris recognition system based on a deep learning approach. *Pattern Analysis and Applications*, 21(3):783–802, 2018. doi:[10.1007/s10044-017-0656-1](https://doi.org/10.1007/s10044-017-0656-1).
- [4] H. S. Ali, A. I. Ismail, F. A. Farag, and F. E. A. El-Samie. Speeded up robust features for efficient iris recognition. *Signal, Image and Video Processing*, 10:1385–1391, 2016. doi:[10.1007/s11760-016-0903-8](https://doi.org/10.1007/s11760-016-0903-8).
- [5] L. E. Ali, J. Luo, and J. Ma. Iris recognition from distant images based on multiple feature descriptors and classifiers. In: *Proc. 2016 IEEE 13th International Conference on Signal Processing (ICSP)*, pp. 1357–1362. IEEE, 2016. doi:[10.1109/ICSP.2016.7878048](https://doi.org/10.1109/ICSP.2016.7878048).
- [6] L. E. Ali, J. Luo, and J. Ma. Effective iris recognition for distant images using log-gabor wavelet based contourlet transform features. In: *Proc. 13th International Conference on Intelligent Computing Theories and Application (ICIC)*, vol. 10361 of *Lecture Notes in Computer Science*, pp. 293–303, 2017. doi:[10.1007/978-3-319-63309-1_27](https://doi.org/10.1007/978-3-319-63309-1_27).
- [7] A. Azizi and H. R. Pourreza. A new method for iris recognition based on contourlet transform and non linear approximation coefficients. In: *Emerging Intelligent Computing Technology and Applications – Proc. 5th International Conference on Intelligent Computing (ICIC)*, vol. 5754 of *Lecture Notes in Computer Science*, pp. 307–316, 2009. doi:[10.1007/978-3-642-04070-2_35](https://doi.org/10.1007/978-3-642-04070-2_35).
- [8] B. V. Bharath, A. S. Vilas, K. Manikantan, and S. Ramachandran. Iris recognition using radon transform thresholding based feature extraction with gradient-based isolation as a pre-processing technique. In: *Proc. 2014 9th International Conference on Industrial and Information Systems (ICIIS)*, pp. 1–8, 2014. doi:[10.1109/ICIINFS.2014.7036572](https://doi.org/10.1109/ICIINFS.2014.7036572).
- [9] B. Connor and K. Roy. Iris recognition using level set and local binary pattern. *International Journal of Computer Theory and Engineering*, 6(5):416–420, 2014. doi:[10.7763/IJCTE.2014.V6.901](https://doi.org/10.7763/IJCTE.2014.V6.901).
- [10] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In: *Proc. 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, pp. 886–893, 2005. doi:[10.1109/CVPR.2005.177](https://doi.org/10.1109/CVPR.2005.177).
- [11] J. Daugman. How iris recognition works. In: A. Bovik, ed., *The Essential Guide to Image Processing*, chap. 5, pp. 715–739. Elsevier, 2009. doi:[10.1016/B978-0-12-374457-9.00025-1](https://doi.org/10.1016/B978-0-12-374457-9.00025-1).

- [12] W. Dong, Z. Sun, and T. Tan. Iris matching based on personalized weight map. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(9):1744–1757, 2011. doi:[10.1109/TPAMI.2010.227](https://doi.org/10.1109/TPAMI.2010.227).
- [13] W. El-Tarhouni, A. Abdo, and A. Elmegreisi. Feature fusion using the Local Binary Pattern Histogram Fourier and the Pyramid Histogram of Feature fusion using the Local Binary Pattern Oriented Gradient in iris recognition. In: *Proc. 2021 IEEE 1st International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering (MI-STA)*, pp. 853–857, 2021. doi:[10.1109/MI-STA52233.2021.9464473](https://doi.org/10.1109/MI-STA52233.2021.9464473).
- [14] C. Fancourt, L. Bogoni, K. Hanna, Y. Guo, R. Wildes, et al. Iris recognition at a distance. In: *Proc. 5th International Conference on Audio-and Video-Based Biometric Person Authentication (AVBPA)*, vol. 3546 of *Lecture Notes in Computer Science*, pp. 1–13. Springer, 2005. doi:[10.1007/11527923.1](https://doi.org/10.1007/11527923.1).
- [15] A. Gangwar, A. Joshi, A. Singh, F. Alonso-Fernandez, and J. Bigun. IrisSeg: A fast and robust iris segmentation framework for non-ideal iris images. In: *Proc. 2016 International Conference on Biometrics (ICB)*, pp. 1–8. IEEE, 2016. doi:[10.1109/ICB.2016.7550096](https://doi.org/10.1109/ICB.2016.7550096).
- [16] L. George and E. Saad. Iris recognition based on the low order norms of gradient components. *International Journal of Computer and Information Engineering*, 8(8):1240–1246, 01 2014. <https://publications.waset.org/9999069.pdf>.
- [17] M. H. Hamd, S. K. Ahmed, et al. Biometric system design for iris recognition using intelligent algorithms. *International Journal of Modern Education and Computer Science*, 10(3):9–16, 2018. doi:[10.5815/ijmecs.2018.03.02](https://doi.org/10.5815/ijmecs.2018.03.02).
- [18] K. P. Hollingsworth, K. W. Bowyer, and P. J. Flynn. The best bits in an iris code. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 31(6):964–973, 2009. doi:[10.1109/TPAMI.2008.185](https://doi.org/10.1109/TPAMI.2008.185).
- [19] Y. Hu, K. Sirlantzis, and G. Howells. Improving colour iris segmentation using a model selection technique. *Pattern Recognition Letters*, 57:24–32, 2015. doi:[10.1016/j.patrec.2014.12.012](https://doi.org/10.1016/j.patrec.2014.12.012).
- [20] M. Z. Islam, S. Nahar, S. S. Islam, S. Islam, A. Mukherjee, et al. Customized K-Means clustering based color image segmentation measuring PRI. In: *Proc. 2021 International Conference on Electronics, Communications and Information Technology (ICECIT)*, pp. 1–4. IEEE, 2021. doi:[10.1109/ICECIT54077.2021.9641094](https://doi.org/10.1109/ICECIT54077.2021.9641094).
- [21] M. E. Kadir, P. S. Akash, S. Sharmin, A. A. Ali, and M. Shoyaib. A proximity weighted evidential k nearest neighbor classifier for imbalanced data. In: *Advances in Knowledge Discovery and Data Mining – Proc. 24th Pacific-Asia Conference (PAKDD)*, vol. 12085 of *Lecture Notes in Computer Science*, pp. 71–83, 2020. doi:[10.1007/978-3-030-47436-2_6](https://doi.org/10.1007/978-3-030-47436-2_6).
- [22] A. Kumar and T.-S. Chan. Iris recognition using quaternionic sparse orientation code (QSOC). In: *Proc. 2012 IEEE Computer Society Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 59–64, 2012. doi:[10.1109/CVPRW.2012.6239216](https://doi.org/10.1109/CVPRW.2012.6239216).
- [23] A. Kumar, T.-S. Chan, and C.-W. Tan. Human identification from at-a-distance face images using sparse representation of local iris features. In: *Proc. 2012 5th IAPR International Conference on Biometrics (ICB)*, pp. 303–309, 2012. doi:[10.1109/ICB.2012.6199824](https://doi.org/10.1109/ICB.2012.6199824).
- [24] C. Li, W. Zhou, and S. Yuan. Iris recognition based on a novel variation of local binary pattern. *The Visual Computer*, 31(10):1419–1429, 2015. doi:[10.1007/s00371-014-1023-5](https://doi.org/10.1007/s00371-014-1023-5).
- [25] P. Li, X. Liu, and N. Zhao. Weighted co-occurrence phase histogram for iris recognition. *Pattern Recognition Letters*, 33(8):1000–1005, 2012. doi:[10.1016/j.patrec.2011.06.018](https://doi.org/10.1016/j.patrec.2011.06.018).
- [26] Y.-H. Li, W. R. Putri, M. S. Aslam, and C.-C. Chang. Robust iris segmentation algorithm

- in non-cooperative environments using interleaved residual U-Net. *Sensors*, 21(4):1434, 2021. doi:[10.3390/s21041434](https://doi.org/10.3390/s21041434).
- [27] J. Liu, Z. Sun, and T. Tan. Distance metric learning for recognizing low-resolution iris images. *Neurocomputing*, 144:484–492, 2014. doi:[10.1016/j.neucom.2014.05.016](https://doi.org/10.1016/j.neucom.2014.05.016).
- [28] B. Madhu, A. Mukherjee, M. Z. Islam, G. Mamun-Al-Imran, R. Roy, et al. Depth motion map based human action recognition using adaptive threshold technique. In: *Proc. 2021 5th International Conference on Electrical Information and Communication Technology (EICT)*, pp. 1–6, 2021. doi:[10.1109/EICT54103.2021.9733611](https://doi.org/10.1109/EICT54103.2021.9733611).
- [29] A. Mukherjee, M. Z. Islam, G. Mamun-Al-Imran, and L. E. Ali. Iris recognition using wavelet features and various distance based classification. In: *Proc. 2021 International Conference on Electronics, Communications and Information Technology (ICECIT)*, pp. 1–4, 2021. doi:[10.1109/ICECIT54077.2021.9641118](https://doi.org/10.1109/ICECIT54077.2021.9641118).
- [30] A. Mukherjee, M. Z. Islam, R. Roy, and L. E. Ali. Block-based local binary patterns for distant iris recognition using various distance metrics. *International Journal of Image, Graphics and Signal Processing*, 16(3):83–99, 2024. doi:[10.5815/ijigsp.2024.03.07](https://doi.org/10.5815/ijigsp.2024.03.07).
- [31] A. Mukherjee, K. S. N. Ripon, L. E. Ali, Z. Islam, and G. Mamun-Al-Imran. Image gradient based iris recognition for distantly acquired face images using distance classifiers. In: *Computational Science and Its Applications – Proc. ICCSA Workshops*, vol. 13381 of *Lecture Notes in Computer Science*, pp. 239–252, 2022. doi:[10.1007/978-3-031-10548-7_18](https://doi.org/10.1007/978-3-031-10548-7_18).
- [32] H. Proenca. Iris recognition: On the segmentation of degraded images acquired in the visible wavelength. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 32(8):1502–1516, 2010. doi:[0.1016/j.dsp.2017.02.003](https://doi.org/10.1016/j.dsp.2017.02.003).
- [33] A. Radman, N. Zainal, and S. A. Suandi. Automated segmentation of iris images acquired in an unconstrained environment using HOG-SVM and GrowCut. *Digital Signal Processing*, 64:60–70, 2017. doi:[10.1016/j.dsp.2017.02.003](https://doi.org/10.1016/j.dsp.2017.02.003).
- [34] K. S. N. Ripon, L. E. Ali, N. Siddique, and J. Ma. Convolutional neural network based eye recognition from distantly acquired face images for human identification. In: *Proc. 2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8, 2019. doi:[10.1109/IJCNN.2019.8852190](https://doi.org/10.1109/IJCNN.2019.8852190).
- [35] M. Salauddin Khan, T. D. Nath, M. Murad Hossain, A. Mukherjee, H. Bin Hasnath, et al. Comparison of multiclass classification techniques using dry bean dataset. *International Journal of Cognitive Computing in Engineering*, 4:6–20, 2023. doi:[10.1016/j.ijcce.2023.01.002](https://doi.org/10.1016/j.ijcce.2023.01.002).
- [36] M. Sardar, S. Banerjee, and S. Mitra. Iris segmentation using interactive deep learning. *IEEE Access*, 8:219322–219330, 2020. doi:[10.1109/ACCESS.2020.3041519](https://doi.org/10.1109/ACCESS.2020.3041519).
- [37] Y. Sari, M. Alkaff, and R. A. Pramunendar. Iris recognition based on distance similarity and PCA. In: *Human-Dedicated Sustainable Product and Process Design: Materials, Resources, and Energy: Proc. 4th International Conference on Engineering, Technology, and Industrial Application (ICE-TIA) 2017*, vol. 1977(1) of *AIP Conference Proceedings*, p. 020044, 2018. doi:[10.1063/1.5042900](https://doi.org/10.1063/1.5042900).
- [38] S. N. Sarode and A. M. Patil. Iris recognition using LBP with classifiers-KNN and NB. *International Journal of Science and Research*, 4(1):1904–1908, 2015. <https://www.ijsr.net/getabstract.php?paperid=SUB15735>.
- [39] M. Savoj and S. A. Monadjemi. Iris localization using circle and fuzzy circle detection method. *International Journal of Computer and Information Engineering*, 6(1):91–93, 2012. <https://publications.waset.org/1468.pdf>.
- [40] E. Severo, R. Laroca, C. S. Bezerra, L. A. Zanlorensi, D. Weingaertner, et al. A benchmark for iris location and a deep learning detector evaluation. In: *Proc. 2018 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–7, 2018. doi:[10.1109/IJCNN.2018.8489638](https://doi.org/10.1109/IJCNN.2018.8489638).

- [41] M. Szymkowski, P. Jasiński, and K. Saeed. Iris-based human identity recognition with machine learning methods and discrete fast fourier transform. *Innovations in Systems and Software Engineering*, 17:309–317, 2021. doi:[10.1007/s11334-021-00392-9](https://doi.org/10.1007/s11334-021-00392-9).
- [42] C.-W. Tan and A. Kumar. Unified framework for automated iris segmentation using distantly acquired face images. *IEEE Transactions on Image Processing*, 21(9):4068–4079, 2012. doi:[10.1109/TIP.2012.2199125](https://doi.org/10.1109/TIP.2012.2199125).
- [43] C.-W. Tan and A. Kumar. Towards online iris and periocular recognition under relaxed imaging constraints. *IEEE Transactions on Image Processing*, 22(10):3751–3765, 2013. doi:[10.1109/TIP.2013.2260165](https://doi.org/10.1109/TIP.2013.2260165).
- [44] C.-W. Tan and A. Kumar. Accurate iris recognition at a distance using stabilized iris encoding and Zernike moments phase features. *IEEE Transactions on Image Processing*, 23(9):3962–3974, 2014. doi:[10.1109/TIP.2014.2337714](https://doi.org/10.1109/TIP.2014.2337714).
- [45] T. Tan and Z. Sun. CASIA Iris Image Database. <http://biometrics.idealtest.org/#/datasetDetail/4>, V. 4 [Accessed: 2018-06-16].
- [46] S. Umer, B. C. Dhara, and B. Chanda. An iris recognition system based on analysis of textural edginess descriptors. *IETE Technical Review*, 35(2):145–156, 2018. doi:[10.1080/02564602.2016.1265904](https://doi.org/10.1080/02564602.2016.1265904).
- [47] C. Wang, J. Muhammad, Y. Wang, Z. He, and Z. Sun. Towards complete and accurate iris segmentation using deep multi-task attention network for non-cooperative iris recognition. *IEEE Transactions on Information Forensics and Security*, 15:2944–2959, 2020. doi:[10.1109/TIFS.2020.2980791](https://doi.org/10.1109/TIFS.2020.2980791).
- [48] Q. Zhang, H. Li, Z. Sun, and T. Tan. Deep feature fusion for iris and periocular biometrics on mobile devices. *IEEE Transactions on Information Forensics and Security*, 13(11):2897–2912, 2018. doi:[10.1109/TIFS.2018.2833033](https://doi.org/10.1109/TIFS.2018.2833033).

Arnab Mukherjee received a B.Sc. degree (Hons.) in Mathematics and M.Sc. degree in Applied Mathematics from the Mathematics Discipline, Khulna University, Khulna, Bangladesh, in 2018 and 2020, respectively. He is currently working as a Lecturer of Mathematics under the Department of Quantitative Sciences, International University of Business Agriculture and Technology, Dhaka, Bangladesh. He has several research articles published in internationally accredited journals and conferences. He is researching the applications of artificial intelligence for high-security surveillance. His research interests include computer vision, especially statistical learning and information intelligence, data science, image processing, biometric recognition, machine learning algorithms, deep learning, and applied mathematics.

Md. Zahidul Islam completed his B.Sc. (Hons.) in Mathematics in 2019 and also completed his M.Sc. in Applied Mathematics in 2021 from the Mathematics Discipline, Khulna University, Khulna, Bangladesh. Currently, he is doing Ph.D. in the School of Engineering, Design, and Built Environment at Western Sydney University, Australia. His research interests include image processing, artificial intelligence, machine learning, deep learning, neural networks, infrastructure, computer vision, and applied mathematics.

Dr. Lasker Ershad Ali received a Bachelor of Science (B.Sc.) degree in Mathematics and a Master of Science (M.Sc.) degree in Applied Mathematics from Khulna University in 2006 and 2008, respectively. He also received a Doctor of Natural Science degree from Peking University in 2018. After working as, a Lecturer (from 2008), an Assistant Professor (from 2010), and an Associate Professor (from 2015), he has been a Professor of Mathematics at Khulna University since 2019. Currently, he is supervising several undergraduate and postgraduate students and conducting some projects as a principal investigator and a co-investigator. He has authored or coauthored around 40 publications in different conferences and peer-reviewed journals. His research interests include biometrics recognition, pattern recognition, signal and image processing, machine learning, deep learning, artificial intelligence, computer vision, and applied mathematics. He is a life member of the Bangladesh Mathematical Society (BMS).