A Voting-Based Ensemble Method for Deep Learning Performance Enhancement

Mohammed Abdel Razek[1], Rania Salah El-Sayed[2], Arwa Mashat[3], Shereen A. El-aal[4]*
Faculty of Science, Mathematics and Computer Science Dept., Al-Azhar University, Cairo, Egypt[1,2,4]
Faculty of Computing and Information Technology, King Abdulaziz University, Rabigh, Saudi Arabia[3]

***Abstract*—Overfitting and limited generalization remain significant challenges for deep learning models, often leading to suboptimal performance on unseen data. To address this, The Divided Ensemble Voting (DEV) method was introduced, a novel approach that strategically partitions a dataset into distinct subsets to train an independent model on each partition. This division encourages each model to specialize in unique features and patterns, thereby increasing ensemble diversity. Predictions from all models are aggregated through a majority voting mechanism to determine the final output, which mitigates overfitting and improves generalization. The proposed method was rigorously evaluated on four binary image classification tasks: Deepfake & Real, Waste Classification, Concrete & Pavement Crack, and Non & Biodegradable Material. Experimental results demonstrate that DEV consistently surpasses the performance of conventional singular models. Accuracy rates improved from 85.55% to 93.1%, 85.12% to 89.6%, 95.42% to 99.0%, and 89.00% to 93.0%, respectively, across the datasets. These findings underscore the efficacy of strategic data partitioning and ensemble consensus in advancing deep learning performance.**

***Keywords*—Divided Ensemble Voting (DEV); deep learning (DL); CNN; binary classification; performance metrics**

I. Introduction and Literature Review

Image classification is a crucial aspect of computer vision, enabling the automatic identification and categorization of images across various applications [1]. Since deep learning models have completely changed how images are processed and categorized, their significance in this domain is profound [2], [3]. These models, particularly convolutional neural networks (CNNs), are characterized by their ability to extract complex features from images, resulting in significantly improved accuracy compared to traditional methods [4]. However, achieving optimal classification remains a challenge due to the inherent variability in image data, which can be influenced by factors such as lighting, occlusion, and object variability [5]. As a result, researchers often explore various learning models and hybrid approaches, they have combined different algorithms to enhance classification performance and address these challenges [6]. Additionally, the computational time required for training and inference poses another significant hurdle, necessitating the selection of efficient models that can generalize well across diverse problems [7], [8].

On the one hand, recent research efforts have focused on improving performance challenges by developing innovative algorithms and frameworks. For instance, a study by Yu (2022) highlighted the integration of self-supervised learning techniques to improve model performance with limited labeled data [9]. The presented method, MICLe, employs a self-supervised learning approach that utilizes unlabeled medical images to enhance classification accuracy in medical diagnostics. By leveraging multiple views of the same pathology, the model effectively learns robust features, demonstrating the potential of self-supervised methods to address the challenges of optimal classification in complex image datasets. Moreover, another aspect of improving accuracy in image classification models involves processing images based on color space. Authors in [10] presented a robust clinical decision support system for diabetic retinopathy detection. This system employs a three-stage deep learning model that integrates color space-based image preprocessing to enhance image quality, followed by feature extraction and classification. Furthermore, the use of transfer learning has gained traction, allowing models pre-trained on large datasets to be fine-tuned for specific tasks, thereby improving performance on smaller, domain-specific datasets [11], [12]. This approach not only enhances accuracy but also reduces the time and resources required for training [14].

On the other hand, another significant area of research focused on optimizing deep learning models for specific datasets while ensuring their generalizability. Researchers are actively exploring various strategies to enhance the accuracy and generalization of machine learning models across diverse datasets [13]. He et al. (2024) demonstrated improved performance and accuracy on large-scale datasets like ImageNet based on a residual learning framework [15]. The framework's effectiveness is evidenced by its success in major competitions, achieving top results in various visual recognition tasks. Additionally, extracting the correct image features is crucial for optimal accuracy [16]. The authors in [17] employed the synthetic minority oversampling technique (SMOTE) to address data imbalance and developed an ML-based ensemble model to predict the probability of second primary lung cancer in patients, optimizing accuracy through feature selection. Moreover, ensemble learning methods, employing multiple learning models, have gained prominence in deep learning-based image classification. Authors in [18] studied compared four ensemble learning techniques (soft voting, weighted average voting, weighted hard voting, and stacking) applied to 16 convolutional neural networks trained on ultrasound images of liver masses. Ensemble methods significantly improved classification accuracy compared to individual CNNs and ResNet101, demonstrating the effectiveness of ensemble learning for liver mass classification in ultrasound images.

The contribution of this study is as follows: 1) Designing a convolutional neural network (CNN) architecture that automat-

*Corresponding authors.

ically learns and extracts relevant features from preprocessed images, capturing intricate patterns. 2) Utilizing four distinct image datasets to demonstrate the model's effectiveness. 3) Classifying images based on the proposed learning model, demonstrating its classification capabilities. 4) Divide each dataset into three subsets and classifying each subset independently to thoroughly train the model's performance. 5) Apply the three trained models to the test set to comprehensively evaluate their classification capabilities. 6) Implementing the Divided Ensemble Voting (DEV) method to enhance overall classification accuracy and reliability. 7) Conducting a comparative analysis of performance metrics to determine the optimal model, examining the effectiveness of using the entire training dataset versus DEV method. The subsequent sections of this paper are structured as follows: Section II presents the proposed learning model architecture and explains the implementation of the Divide Ensemble Voting (DEV) method. Section III-A describes the utilized dataset and its preparation for the learning model. Section IV provides the results and discussion, and finally, Section V offers the conclusion.

II. Research Methodology

This section introduces the Divided Ensemble Voting (DEV) methodology, detailing its operational framework and implementation mechanics for enhanced classification performance.

A. Divided Ensemble Voting (DEV)

DEV method is a novel ensemble learning approach designed to enhance classification performance through strategic data partitioning and collaborative model voting.

Let D be a labeled dataset where each instance $x \in D$ belongs to a class $C_i \in \{C_1, \ldots, C_n\}$. The goal is to learn an ensemble model that improves generalization by using partitioned subsets of D. the Divided Ensemble Voting (DEV) technique consists of two phases:

- Partitioned Training: Train N similar or diverse models on disjoint subsets of D.
- Voting-Based Inference: Aggregate predictions via majority voting.

The DEV algorithm technique starts with:

1) Data Splitting: Split D into:
   - Data for training:D_{train},
   - Data for validation: D_{val}, and
   - Data for testing: D_{test}
2) Partitioning: Divide D_{train} into N non-overlapping subsets: $\{D_{train-1}, D_{train-2}, ..., D_{train-N}\}$.

   where,

$$D_{train} = \bigcup_{i=1}^{N} D_{train-i}, \quad \& \quad D_{train-i} \cap D_{train-j} = \varnothing \quad \forall i \neq j \tag{1}$$

3) Model Training: train multiple independent similar or divers models on different chunks of the training data.
   - Start with base model architecture ($e.g.$ a CNN for images)
   - Create N identical copies of this model: $\{M_1, M_2, ..., M_N\}$
   - Each model M_i is trained only on its assigned subset $D_{train-i}$ (a small portion of the full training data).

$$\min_{\Theta_i} \sum_{(x,y) \in D_{train,i}} \tau(M_i(x, \Theta_i), y) \tag{2}$$

   where,

   - Θ_i: Trainable parameters of model M_i.
   - $D_{train,i}$: The i-th partitioned subset of training data.
   - (x, y): Input-label pair from $D_{train,i}$.
   - τ : Loss function ($e.g.$, cross-entropy for classification).
   - $M_i(x, \Theta_i)$: Prediction of model M_i with parameters Θ_i for input x.
4) Ensemble Prediction: In this step, For a test instance $x \in D_{test}$:
   - Obtain predictions $\{M_1, M_2, ..., M_N\}$
   - Compute the final prediction $P_{final}(x)$ via majority voting:

$$P_{final}(x) = mode(\{M_i(x)\}_{i=1}^{N}) \tag{3}$$

   The DEV algorithm's effectiveness stems from its foundation in ensemble learning theory and statistical learning principles. Below are its core theoretical properties:

   - Diversity Mechanism: By training each model M_i on disjoint subsets, the algorithm ensures:
     - Low Correlation Between Errors: Models make mistakes on different data points, reducing collective bias.
     - Complementary Expertise: Each model specializes in unique features of its partition.

Algorithm 1 illustrates the architecture of the proposed method DEV.

Algorithm 1 Divided Ensemble Voting

Input: Dataset D, Number of partitions N, Model architecture M
Output: Final predictions for D_{test}
Split D into D_{train}, D_{val}, and D_{test}
Partition D_{train} into N subsets: D_{train1}, D_{train2}, ..., D_{trainN}
Initialize an empty list of models: $Models = []$
for epochs = 1 to N **do**
    Clone model M to M_i
    Train M_i on $D_{train,i}$
    Add M_i to $Models$
end for
Initialize empty list for predictions: $Predictions = []$
for each x in D_{test} **do**
    Collect predictions $P_1(x), P_2(x), \ldots, P_N(x)$ from all models
    Compute final prediction: $P_{final}(x) = mode(P_1(x), P_2(x), \ldots, P_N(x))$
    Append $P_{final}(x)$ to Predictions
end for
Return Predictions

Fig. 1 illustrates the DEV method based on the CNN learning model for binary classification. As shown in the figure, the dataset is divided into three distinct subsets: training(D_{train}), validation(D_{val}), and testing(D_{test}). Subsequently, the training set is further subdivided into N sets, designated as D_{train1}, D_{train_2}, $\cdots$, D_{trainN}. Each of these training sets was utilized to train separate instances of the model, designated as M_1, M_2, $\cdots$,M_N, respectively. Specifically, D_{train1} is processed by M_1, D_{train_2} is processed by M_2 and so forth for all partitioned datasets and corresponding models. Notably, the D_{val} set remained constant across all models to ensure consistency in performance evaluation. Upon completion of the training process, N distinct trained models were obtained. Subsequently, the D_{test} set is evaluated using each of the trained models M_i, $\{i = 1, 2 \cdots, N\}$, yielding N separate accuracy results P_1, $P_2 \cdots$, P_N. To determine the optimal classification for each image, a mode mechanism was employed, reflecting the majority vote among the N models. This approach facilitated the aggregation of predictions, enhancing the robustness of the classification results.

III. Experimental Framework

This section details the experimental framework, including: 1) the benchmark datasets employed, 2) the core components of the DEV method: image processing , and baseline CNN model architecture.

A. Dataset

This subsection describes the benchmark datasets selected for this study, which were chosen based on their substantial size and representative sample diversity. The large volume of high-quality training samples inherently provides sufficient feature variation, rendering data augmentation unnecessary. This approach not only streamlines the training pipeline but also preserves the authentic statistical distribution of the original data, eliminating potential biases introduced by synthetic transformations.

1) Deepfake and Real (DS-FR): This dataset comprises a collection of manipulated and real images, specifically focusing on human faces, and contains two classes: Fake and Real [19]. The manipulated images are generated using various techniques, primarily featuring deepfake technology, which involves replacing an individual's likeness in an existing image or video with that of another person. Each image in the dataset is formatted as a 256 x 256, showcasing either real or artificially generated faces. The dataset includes 70,000 authentic, alongside 70,000 synthetic faces derived from a larger pool of 1 million fake faces generated by StyleGAN [20]. For training, 60,000 images are utilized, with 30,000 for each class; 2,000 images are allocated for validation, with 1,000 for each class; and 4,000 images are designated for testing, split evenly into 2,000 for each class. Fig. 2 presents samples from the dataset, the validation set comprises 5% of the training data. The precise numbers for D_{train}, D_{val}, and D_{test} in our proposed model are detailed in Table I.

TABLE I. Dataset Distribution

Dataset	D_{train}	D_{val}	D_{test}	Total
DS-FR	60000	2000	4000	12000
DS-OR	21435	1129	2513	25077
DS-NP	24000	2000	4000	30000
DS-BNB	59922	2000	4000	65922

2) Waste Classification (DS-OR): The dataset consists of 25,077 images of solid household waste, categorized into two classes: organic (Org) waste, comprising 13,966 images, and Recyclable ($Recy$) waste, consisting of 11,111 images [21]. The acquired images are colored $.jpg$ files of random portrait and landscape orientation with resolution ranging from 191 pixels (minimum) $\times$ 264 pixels (maximum) [22]. The number of training and testing images is 22564 and 2513, respectively. Additionally, the validation set comprises 5% of the training data. Fig. 2 presents samples of the dataset. The numbers for D_{train}, D_{val}, and D_{test} in our proposed model are shown in Table I.

3) Concrete and Pavement Crack (DS-NP): This dataset consists of 30,000 images of concrete and pavement surfaces, classified into two categories: cracked and non-cracked [23]. Collected by Omoebamije Oluwaseun, a civil engineering student at the Nigerian Army University Biu, the images were obtained using a DJI Mavic 2 Enterprise drone and a smartphone. Each image is downsized to 227 $\times$ 227 pixels and stored in RGB and JPEG format. It is then arranged into Negative (Neg) and Positive (Pos) folders for convenient access and analysis. Samples of the dataset are presented in Fig. 2, the validation set comprises 5% of the training data. The precise numbers for D_{train}, D_{val}, and D_{test} are presented in Table I.

4) Non and Biodegradable Material (DS-BNB): This dataset comprises approximately 256,000 images, derived from an original collection of 156,000, representing two waste categories: Biodegradable (Bio) and Non-biodegradable ($Non\text{-}Bio$) [24], [25]. The Bio class includes materials that can be naturally decomposed, such as food and plant waste,

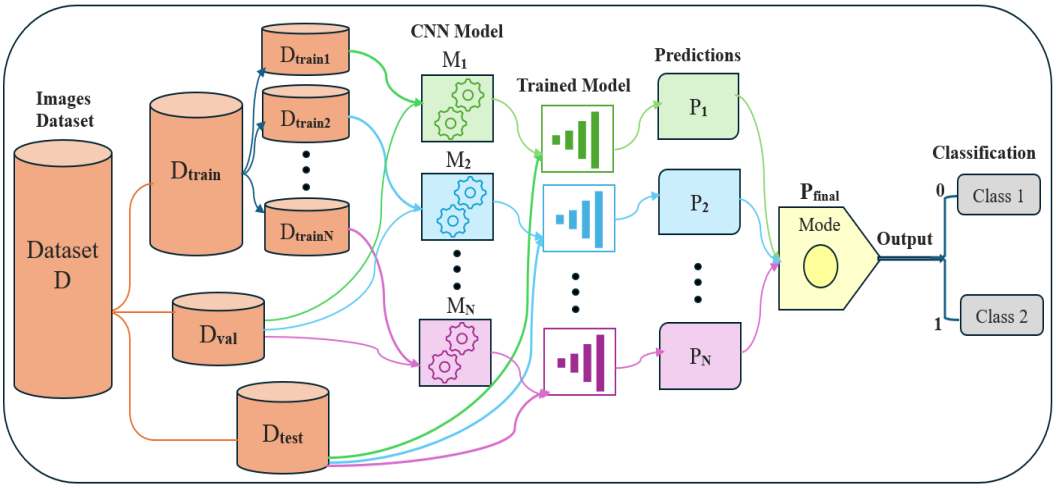

Fig. 1. Divided Ensemble Voting (DEV) model.

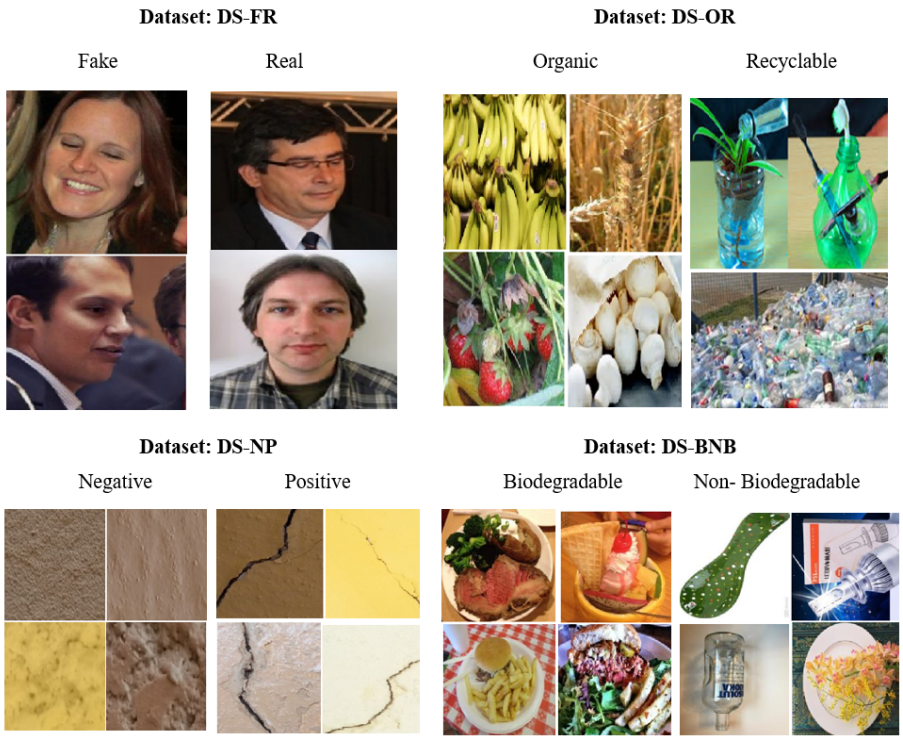

Fig. 2. Datasets samples.

while the non-biodegradable $Non\text{-}Bio$ class consists of materials like plastics and metals that cannot be decomposed. The dataset is divided into main subsets: training and testing. The training set further split into four parts, with the "TRAIN.1" folder designated for this research. Fig. 2 presents samples from the dataset. From the D_{test}: 2,000 images are randomly selected from each category, while 1,000 images from each category are randomly chosen for the validation set. The precise numbers for D_{train}, D_{val}, and D_{test} are presented in Table I.

The DEV method integrates two key components: 1) image preprocessing and 2) independent baseline CNN model training on partitioned data subsets. Each component is detailed in the following subsections.

B. Image Processing

Effective preprocessing and processing of image datasets are essential for improving the performance of classification tasks. These steps ensure that the images is formatted appropriately for analysis, thereby enhancing the model's accuracy. In this study, each image was assigned to its respective category. For each category of the dataset, the images were read using OpenCV, resized to a uniform dimension of 64x64 pixels, and

normalized by scaling pixel values to the range [0, 1]. This normalization step is essential, in order to improve the model's performance and rate of convergence.

C. Baseline CNN Model

Convolutional neural networks (CNNs), which are the basis of the deep learning model, are made to efficiently classify the processed images. The model architecture plays a crucial role in extracting relevant features from images and making accurate predictions regarding each class of the dataset.

Fig. 3 shows the baseline CNN architecture where input images pass through three convolutional blocks with increasing filters $F(32, 64, 128)$ and a fixed kernel size $K(3 \times 3)$. Each block includes convolutional layers with ReLU activation, batch normalization, max pooling, and dropout. The final block outputs a 2048-dimensional vector, followed by a dense layer $(256\ units)$ with dropout before the softmax output layer $(2\ units)$. The model uses the Adam optimizer and categorical cross-entropy loss for binary classification. Table II provides details on the configuration of the CNN model layers.

Total parameters: $815,818$ ($814,410$ trainable; $1,408$ non-trainable, from batch normalization's moving statistics). The trainable parameters primarily consist of convolutional and dense layer weights and biases, along with Batch Normalization's learnable scale (γ) and shift (β) parameters. The non-trainable parameters exclusively originate from Batch Normalization layers, representing the moving averages of mean (μ) and variance (σ^2) that are updated during inference but remain fixed during training to maintain stable feature distributions.

In the experimental implementation of the DEV method, the training set was partitioned into N=3 distinct subsets, each processed by a dedicated instance of N=3 independently trained models. Training was conducted over 20 epochs, with validation set performance monitored at each epoch to verify generalization capability beyond the training data. This rigorous evaluation protocol ensures unbiased assessment using completely distinct data partitions, maintaining the integrity of performance metrics. The partitioned approach not only facilitates specialized feature learning but also provides built-in validation through inter-model consensus.

IV. Results and Discussion

This section presents the results derived from three stages of analysis: first, performance accuracy using the entire training set; second, the accuracy after partitioning the training set into three subsets; and finally, accuracy achieved after implementing the DEV method.

A. Model Performance Evaluations on Datasets Based on the Entire Training Sets

This subsection evaluates the model's performance across four datasets, analyzing training/validation losses, accuracy metrics, and overall classification performance. The assessment is conducted using complete training sets for model development and validation.

Figures of the training and validation losses for the four datasets, along with their accuracy performance, are shown in Fig. 4 and 5, 6 and 7 across 20 epochs for DS-FR, DS-OR, DS-NP and DS-BNB datasets, respectively. Moreover, Table III illustrates the loss values for training, validation and testing across these datasets based on the entire training set.

Furthermore, Table IV presents the performance accuracies achieved for the training, validation, and testing sets of the proposed learning approach across the four datasets. Table V further illustrates the performance metrics achieved by applying the learning model approach to the test set for each dataset. Additionally, Fig. 8 demonstrates the confusion matrices for each dataset which provide a comprehensive overview of the model's figure classification performance across different labels.

For DS-FR, As demonstrated in Table III, while the model fits the training data well, it may not generalize as effectively to test data, indicated by the higher test loss of 0.5856. Furthermore, the model achieved a high training accuracy of 98.18% and a validation accuracy of 93.4%, although the test accuracy was comparatively lower at 85.55% with execution training time (ET) of 8 hrs., 21 sec, as shown in Table IV. The precision and recall values for the class labels in DS-FR dataset also reflect this trend, with precision scores of 0.87 and 0.84 for classes Fake and Real, respectively, as shown in Table V.

Additionally, in DS-OR dataset, the loss values for this dataset were higher, with a training loss of 0.2392 and a validation loss of 0.3445, as shown in Table III. Moreover, accuracies of training, validation and testing were 88.94%, 86.1% and 85.12%, respectively, as shown in Table IV. The precision for Org class was notably high at 0.98, but the recall at 0.68 suggests that the model struggled to identify all relevant instances of this class, leading to a lower f1-score of 0.80, as shown in Table V.

Furthermore, DS-NP dataset exhibited strong performance across all metrics, with a training accuracy of 96.41% and a validation accuracy of 96.95%, as shown in Table IV. The test accuracy of 95.42% was accompanied by a low-test loss of 0.1816 (Table III), indicating effective generalization. The precision and recall values for both class labels Neg and Pos achieving 1.00 and 0.92 for precision, and 0.91 and 1.00 for recall, respectively, resulting in f1-scores of 0.95 and 0.96 as shown in Table V.

Finally, DS-BNB dataset showed a training accuracy of 98.14% and a validation accuracy of 87.7%, with a test accuracy of 89%, as demonstarted in Table IV. Despite the high training accuracy, the validation loss of 0.3361 (Table III) suggests potential overfitting. As shown in Table V, the precision and recall values for class labels were relatively balanced, with precision scores of 0.93 for Bio class and 0.86 for $Non\text{-}Bio$, reflecting a strong capability in predicting instances of these classes.

B. Performance Evaluation Based on Partitioning of Training Data

This subsection evaluates model performance across the four datasets by analyzing: 1) training/validation losses, 2) accuracy metrics, and 3) overall classification effectiveness. For enhanced model evaluation, each original training set is further partitioned into three distinct training subsets.

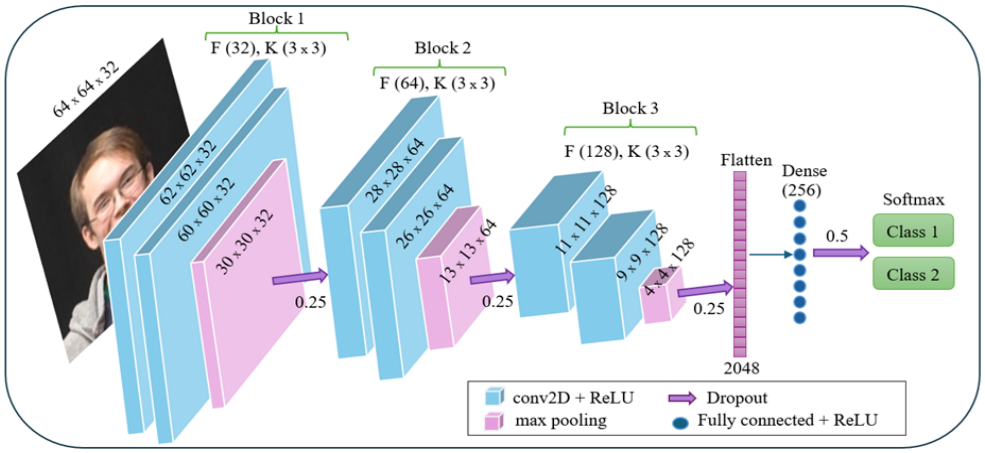

Fig. 3. Overview of the baseline CNN model architecture.

TABLE II. OVERVIEW OF THE BASELINE CNN ARCHITECTURE AND ITS PROPERTIES

Layer Type	Output Shape	Kernel Size(K)	Stride	trainable Params	Non-trainable Params
Conv2D	(62, 62, 32)	(3, 3)	(1,1)	896	0
BatchNormalization	(62, 62, 32)	–	–	128 (γ, β)	128 (μ, σ^2)
Conv2D	(60, 62, 32)	(3,3)	(1,1)	9248	0
BatchNormalization	(60, 62, 32)	–	–	128 (γ, β)	128 (μ, σ^2)
MaxPooling2D	(30, 30, 32)	(2,2)	(2,2)	0	0
Dropout (0.25)	(30, 30, 32)	–	–	0	0
Conv2D	(28, 28, 64)	(3, 3)	(1,1)	18496	0
BatchNormalization	(28, 28, 64)	–	–	256 (γ, β)	256 (μ, σ^2)
Conv2D	(26, 26, 64)	(3,3)	(1,1)	36928	0
BatchNormalization	(26, 26, 64)	–	–	256 (γ, β)	256 (μ, σ^2)
MaxPooling2D	(13, 13, 64)	(2,2)	(2,2)	0	0
Dropout (0.25)	(13, 13, 64)	–	–	0	0
Conv2D	(11, 11, 128)	(3, 3)	(1,1)	73,856	0
BatchNormalization	(11, 11, 128)	–	–	512 (γ, β)	512 (μ, σ^2)
Conv2D	(9, 9, 128)	(3,3)	(1,1)	147,584	0
BatchNormalization	(9, 9, 128)	–	–	512 (γ, β)	512 (μ, σ^2)
MaxPooling2D	(4, 4, 128)	(2,2)	(2,2)	0	0
Dropout (0.25)	(4, 4, 128)	–	–	0	0
Flatten	(2048,)	–	–	0	0
Dense	(256,)	–	–	524544	0
BatchNormalization	(256,)	–	–	1024 (γ, β)	1024 (μ, σ^2)
Dropout (0.5)	(256,)	–	–	0	0
Dense (Output)	(256,)	–	–	514	0

TABLE III. LEARNING CNN MODEL LOSSES BASED ON THE ENTIRE TRAINING SET

Dataset	training Loss	validation Loss	testing Loss
DS-FR	0.0506	0.1991	0.5856
DS-OR	0.2392	0.3445	0.3783
DS-NP	0.1362	0.1041	0.1816
DS-BNB	0.0578	0.3361	0.2806

TABLE IV. OVERALL ACCURACY AND EXECUTION TIME BASED ON THE ENTIRE TRAINING SET

Dataset	training ACC	validaton ACC	testing ACC	Execution Time(ET)
DS-FR	98.18%	93.4%	85.55%	8 hr, 21 min,11 sec
DS-OR	88.94%	86.09%	85.12%	3 hr, 4 min, 35 sec
DS-NP	96.41%	96.95%	95.42%	3 hr, 55 min, 37sec
DS-BNB	98.14%	87.70%	89.00%	7 hr, 27 min

1) Partitioning the training set: Based on DEV method, the original training set D_{train} was further subdivided into three distinct subsets referred to as D_{train1} , D_{train2} and D_{train3} using random stratified sampling to preserve class distribution across all subsets, the number of images in each subset is illustrated in Table VI.

TABLE V. PERFORMANCE METRICS OF THE CNN LEARNING MODEL ON TEST SET ACROSS ALL DATASETS

Dataset	class	precision	recall	f1-score
DS-FR	Fake	0.87	0.83	0.85
	Real	0.84	0.88	0.86
DS-OR	Org	0.98	0.68	0.80
	Recy	0.79	0.99	0.88
DS-NP	Neg	1.00	0.91	0.95
	Pos	0.92	1.00	0.96
DS-BNB	Bio	0.93	0.84	0.88
	Non-Bio	0.86	0.94	0.89

TABLE VI. PARTITIONING TRAINING SET INTO THREE SUBSETS

Dataset	D_{train1}	D_{train2}	D_{train3}
DS-FR	19800	20100	20100
DS-OR	7073	7181	7181
DS-NP	7919	8040	8041
DS-BNB	19774	20074	20074

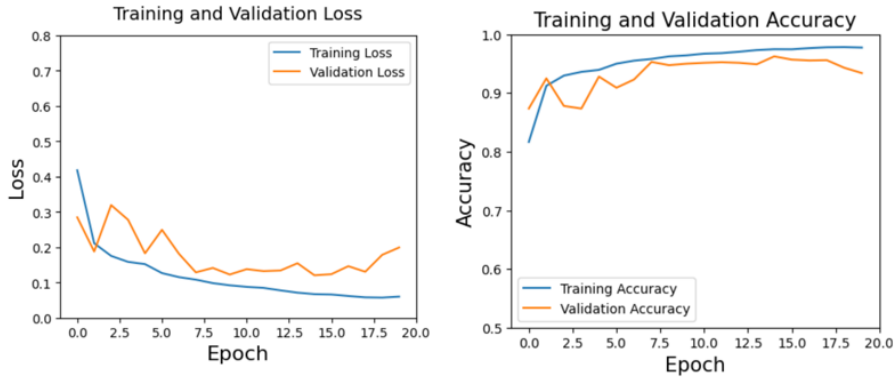

Fig. 4. DS-FR training and validation losses and its accuracy performance.

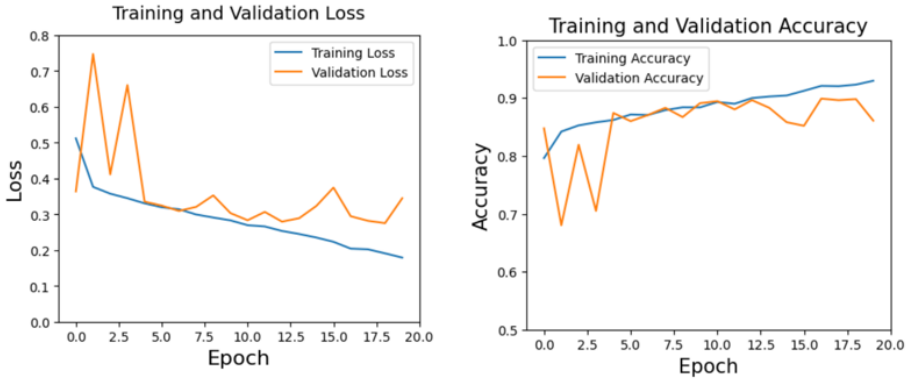

Fig. 5. DS-OR training and validation losses and its accuracy performance.

Fig. 6. DS-NP training and validation losses and its accuracy performance.

2) Models performance evaluation: Each of training subsets is used to independently train a separate instance of the CNN learning model M_i, $i = \{1, 2, 3\}$. The validation set was kept constant across all model instances to ensure consistent performance monitoring during training. Once training was completed, each of the three models was evaluated on the same independent test D_{test}, producing three individual accuracy scores.

Table VII presents the loss values for models M_1, M_2, M_3 trained on subsets D_{train1}, D_{train2} and D_{train3}, respectively. The table also includes the corresponding validation and test losses for each dataset.

Table VIII presents the performance accuracies of Models M_1, M_2 and M_3 for the training subsets $\{D_{train1}, D_{train2}, D_{train3}\}$, validation, and test sets across all four datasets, along with their execution times for each training subset. Furthermore, Table IX summarizes the performance metrics of each model (M_1, M_2 and M_3) on the test sets, evaluated per class across all datasets. Finally, Fig. 9, 10, 11 and 12 display the confusion matrix for models (M_1, M_2 and M_3) across the four datasets, providing detailed insights into their classification performance.

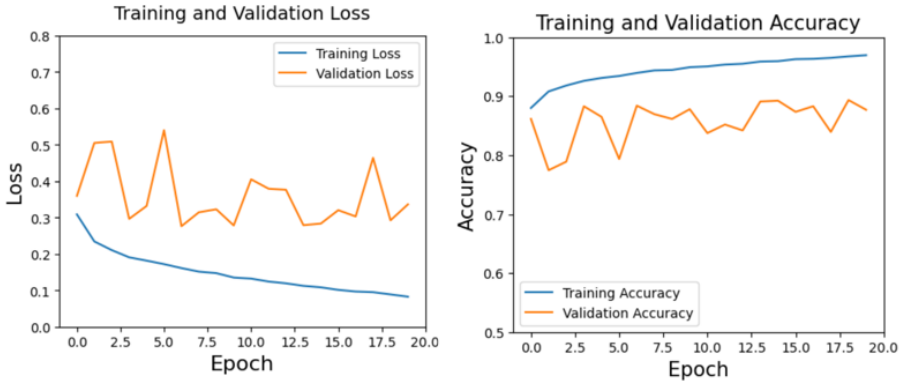

Fig. 7. DS-BNB training and validation losses and its accuracy performance.

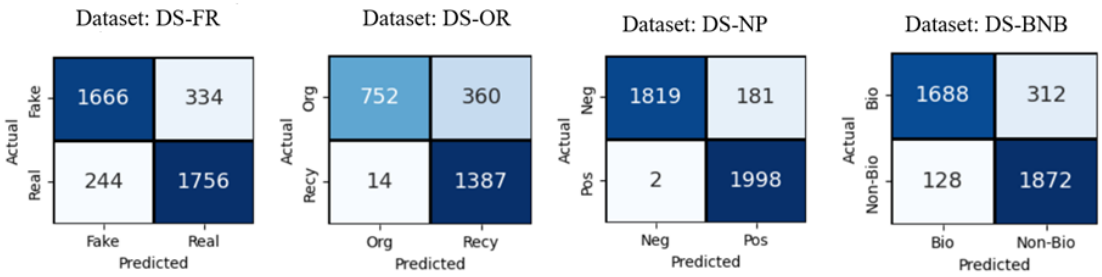

Fig. 8. Confusion matrix for all datasets based on the entire training set.

TABLE VII. TRAINING, VALIDATION AND TESTING LOSS VALUES FOR MODELS M_1, M_2 , M_3

Dataset	model	training loss	validation loss	testing loss
DS-FR	M_1	0.0189	0.2116	**0.2921**
	M_2	0.2511	0.7716	1.0366
	M_3	0.0209	0.2633	0.3174
DS-OR	M_1	0.2281	0.4054	0.4043
	M_2	0.1851	0.3830	0.4479
	M_3	0.1565	0.3518	**0.3172**
DS-NP	M_1	0.0362	0.0655	**0.0537**
	M_2	0.0540	0.0623	0.0601
	M_3	0.0424	0.0618	0.0594
DS-BNB	M_1	0.0757	0.3029	**0.2794**
	M_2	0.0755	0.3969	0.3947
	M_3	0.1324	0.3742	0.3720

For the DS-FR dataset , as summarized in the three Tables VII, VIII and IX, M_1 and M_3 showed lower losses (test loss: 0.29–0.32) compared to M_2 (1.04), indicating better generalization. M_1 and M_3 also outperformed M_2, achieving 99% training and 90.3% testing accuracy, versus M_2 93.6% (training) and 79.8% (testing) (Table VIII). Per-class metrics (Table IX) reveal M_1 balanced precision (0.90–0.91), while M_3 achieved 0.94 (Fake) and 0.87 (Real); M_2 had high Fake-class precision (0.95) but poor recall (0.63). Overall, M_1 excelled in accuracy and loss, with M_3 as a close competitor, while M_2 lagged significantly.

The results for DS-OR dataset, presented in Tables VIII, VII and IX, reveal notable differences in performance among models M_1, M_2, and M_3. M_3 achieved lowest losses (0.16 training, 0.32 testing) and the highest accuracy (94.03% training, 90.89% testing), demonstrating superior generalization. While M_1 showed strong precision for Org class (0.98) and M_2 had high $Recy$ recall (0.93), M_3 delivered balanced performance across classes (Org: 0.95 precision/0.83 recall; $Recy$: 0.88/0.97) with the best F1-scores (0.92 $Recy$). Models M_1 and M_2 exhibited trade-offs :M_1 in recall (0.70 Org) and M_2 in precision (0.69 Org), but M_3 consistently outperformed in both accuracy and class-wise metrics, establishing itself as the most robust choice for this dataset.

Moreover, For DS-NP dataset, as shown in Tables VIII, VII and IX, M_1 achieved the lowest losses (0.036 training, 0.054 testing), indicating superior generalization. Additionally, M_1 led with 99.03% training and 98.43% testing accuracy, followed closely by M_2 (98.56%/98.22%) and M_3 (slightly lower). All models excelled in class-wise metrics (precision/recall 0.98–0.99 for both Pos/Neg classes), with F1-scores consistently near 0.98. Results demonstrate exceptional performance across all models, highlighting the robustness of the learning approach on this dataset.

For the DS-BNB dataset (Tables VIII, VII and IX), M_1 achieved the lowest losses (0.3029 validation, 0.2794 testing) , alongside the best balance with 97.19% training and 89.6% testing accuracy, indicating superior generalization. While M_2 had higher training accuracy (97.83%) but poorer test performance (83.75%), and M_3 lagged in training (94.35%) and testing (86.83%), M_1 excelled in class-wise metrics: precision/recall of 0.89/0.90 (Bio) and 0.90/0.89 (Non-Bio), yielding consistent F1-scores (0.90 both classes). Though Models M_2, M_3 showed trade-offs, M_1 consistently delivered robust accuracy, loss, and per-class performance, establishing it as the most effective choice for this dataset.

TABLE VIII. ACCURACIES AND EXECUTION TIME BASED ON MODELS M_1, M_2, M_3

Dataset	model	training ACC	validation ACC	testing ACC	ET	total ET
DS-FR	M_1	99.32%	93.70%	**90.38%**	2 hr, 55 min, 49 sec	8 hr, 36 min, 25 sec
	M_2	93.59%	85.05%	79.77%	2 hr, 52 min, 25 sec	
	M_3	99.23%	92.65%	90.35%	2 hr, 48 min, 11 sec	
DS-OR	M_1	89.81%	83.17%	85.83%	1 hr, 4 min, 29 sec	3 hr, 13 min, 24 sec
	M_2	92.91%	87.16%	84.12%	1 hr, 6 min, 29 sec	
	M_3	94.03%	87.78%	**90.89%**	1 hr, 2 min, 26 sec	
DS-NP	M_1	99.03%	98.2%	**98.43%**	1 hr, 20 min, 58 sec	4 hr, 5 min, 17 sec
	M_2	98.56%	98.4%	98.22%	1 hr, 21 min, 3 sec	
	M_3	98.64%	98.05%	98.15%	1 hr, 23 min, 16 sec	
DS-BNB	M_1	97.19%	88.8%	**89.6%**	2 hr, 27 min, 22 sec	7 hr, 39 min, 8 sec
	M_2	97.83%	84.65%	83.75%	2 hr, 42 min, 37 sec	
	M_3	94.35%	86.20%	86.83%	2 hr, 29 min, 9 sec	

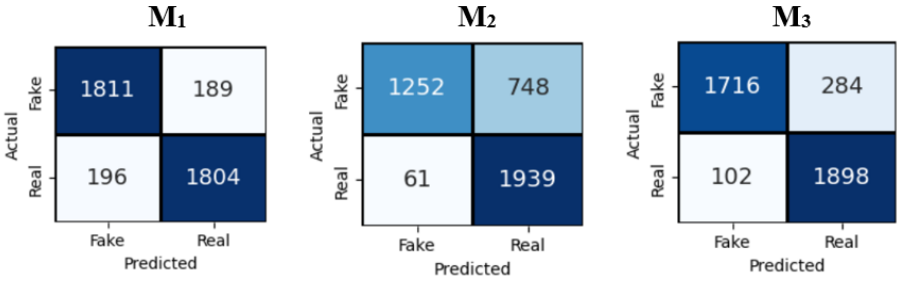

Fig. 9. Confusion matrix for models M_1, M_2, M_3 on the deepfake & real (DS-FR) dataset.

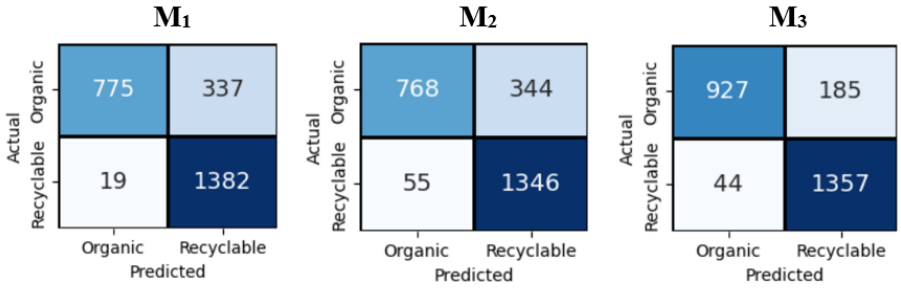

Fig. 10. Confusion matrix for models M_1, M_2, M_3 on the waste classification (DS-OR) dataset.

The comparative analysis reveals distinct performance patterns across datasets. M_1 demonstrated consistent superiority in DS-FR (high accuracy, balanced class metrics) and DS-BNB (lowest losses, strong F1-scores), establishing it as the most reliable overall. M_3 emerged as the top performer in DS-OR (best accuracy and balanced class metrics) while maintaining competitive results in DS-NP. Though M_2 showed high precision for specific classes (e.g., 0.95 Fake in DS-FR, 0.93 recall in DS-OR), it struggled with generalization (notably higher losses in DS-FR/BNB). The DS-NP dataset proved universally favorable, with all models achieving near-perfect metrics, suggesting its inherent class separability. These findings underscore that the composition and diversity of training subsets play a critical role in shaping model effectiveness on test data. To address this, DEV method is proposed as a systematic framework to optimize performance through ensemble learning.

C. Final Classification: DEV method vs. Entire Training-Based Learning

The DEV method aggregates predictions P_1, P_2 and P_3 from Models M_1, M_2, and M_3, respectively, using majority voting (mode) to determine final test-set labels. This ensemble strategy mitigates variance from weight initialization and subset-specific biases, enhancing generalization.

Initially, Performance metrics (Table X) highlight DEV's effectiveness: DS-FR achieved balanced F1-scores (0.93), DS-OR showed strong $Recy$-class recall (0.98) outperforming the whole-dataset approach in Table V, while DS-NP attained near-perfect metrics (precision/recall 0.99), and DS-BNB maintained high scores (Bio: 0.94, Non-Bio: 0.92). These results demonstrate DEV's consistency in optimizing accuracy across diverse datasets. The method's adaptability to varying class complexities underscores its practical utility.

Secondly, Comparative analysis of confusion matrices (Fig. 8 vs. 13) reveals DEV's consistent improvements across all datasets. For DS-FR, $Fake$ class detections rose to 1,756 (from 1,666) with fewer false positives (244 vs. 334). Similar gains occurred in DS-OR (Org: 876 vs. 752) and DS-NP (Pos: 1,970 vs. 1,819), while DS-BNB showed higher Bio class accuracy (1,838 vs. 1,688). These results demonstrate DEV's superiority over whole-dataset training, enhancing both precision and generalization.

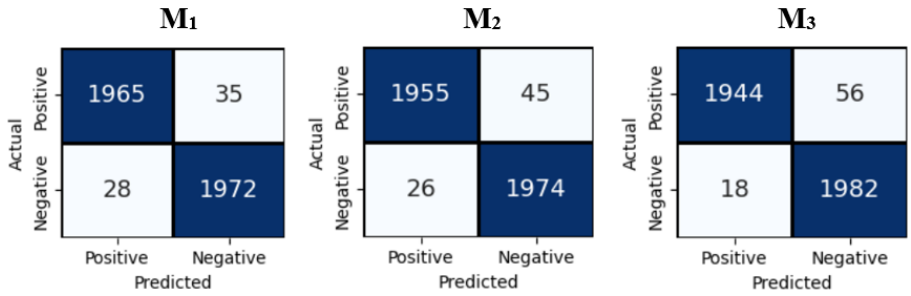

Fig. 11. Confusion matrix for models M_1, M_2, M_3 on concrete & pavement crack (DS-NP) dataset.

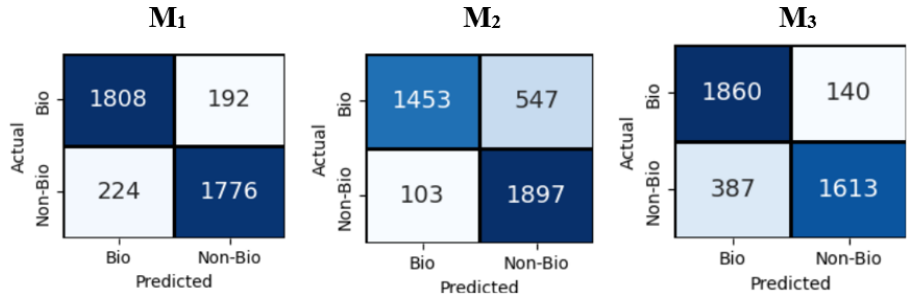

Fig. 12. Confusion matrix for models M_1, M_2, M_3 on non & biodegradable (DS-BNB) dataset.

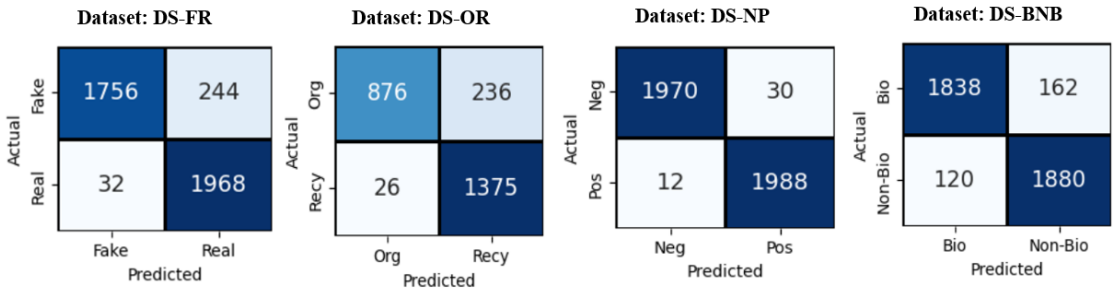

Fig. 13. Confusion matrix for test set based on DEV method.

Fig. 14 demonstrates DEV's consistent superiority over whole-dataset training, with accuracy gains across all datasets - most notably in DS-NP (95.0% to 99.0%). This improvement highlights how DEV's ensemble approach enhances classification reliability through collaborative model strengths. The findings validate DEV as an effective strategy for optimizing predictive performance.

Comparative execution time analysis (Tables IV, VIII) reveals that the additional 10 minutes required for training Models (M_1, M_2, M_3) under the DEV framework represents a negligible computational overhead relative to the significant performance gains achieved. This minor time investment is substantially outweighed by the method's demonstrated improvements in classification accuracy and robustness across all datasets.

The DEV method demonstrates superior performance over whole-dataset training, delivering consistently higher accuracy with improved true/false positive ratios across all datasets, while maintaining computational efficiency.

D. Discussion: Comparing Baseline CNN and DEV Method for the Four Datasets

This section provides a comprehensive performance comparison between the baseline CNN architecture and the DEV method across all four benchmark datasets.

1) Performance evaluation for DS-FR dataset: The baseline CNN model, trained on the full DS-FR dataset (12K images), achieved a test accuracy of 85.55%, with a notable test loss of 0.586, indicating potential overfitting (train loss: 0.051, validation loss: 0.199). The confusion matrix revealed 334 FP ($Real$ images misclassified as $Fake$) and 244 FN ($Fakes$ misclassified as $Real$), suggesting challenges in distinguishing high-quality deepfakes real images.

In contrast, the DEV method—which splits DS-FR (D_{train}) into three smaller training sets (D_{train1}: 19800, D_{train2}/D_{train3}: 20100 each) , keeping (D_{val}:2000, D_{test}:4000), yielded three models (M_1, M_2 , M_3). Model M_1 emerged as the strongest, achieving 90.4% accuracy with lower test loss (0.292), outperforming both the baseline and other models (M_2: 79.8%, M_3: 90.3% with test loss

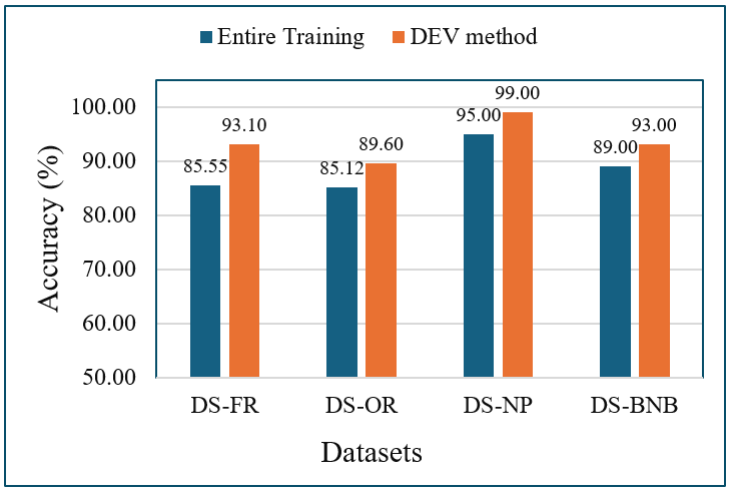

Fig. 14. Overall accuracy: Entire training set vs. DEV method across the four datasets.

TABLE IX. PERFORMANCE METRICS OBTAINED FOR EACH CLASS BASED ON MODELS M_1, M_2 , M_3

Dataset	model	class	precision	recall	f1-score
DS-FR	M_1	Fake	0.90	0.91	0.90
		Real	0.91	0.90	0.90
	M_2	Fake	0.95	0.63	0.76
		Real	0.72	0.97	0.83
	M_3	Fake	0.94	0.86	0.90
		Real	0.87	0.95	0.91
DS-OR	M_1	Org	0.98	0.70	0.81
		Recy	0.80	0.99	0.89
	M_2	Org	0.69	0.93	0.79
		Recy	0.96	0.8	0.87
	M_3	Org	0.95	0.83	0.89
		Recy	00.88	0.97	0.92
DS-NP	M_1	Pos	0.99	0.98	0.98
		Neg	0.98	0.99	0.98
	M_2	Pos	0.99	0.98	0.98
		Neg	0.98	0.99	0.98
	M_3	Pos	0.99	0.97	0.98
		Neg	0.97	0.99	0.98
DS-BNB	M_1	Bio	0.89	0.90	0.90
		Non-Bio	0.90	0.89	0.90
	M_2	Bio	0.93	0.73	0.82
		Non-Bio	0.78	0.95	0.85
	M_3	Bio	0.83	0.93	0.88
		Non-Bio	0.92	0.81	0.86

TABLE X. PERFORMANCE METRICS BASED ON DEV METHOD

Dataset	class	precision	recall	f1-score	ACC
DS-FR	Fake	0.98	0.88	0.93	93.1%
	Real	0.89	0.98	0.93	
DS-OR	Org	0.97	0.79	0.87	89.6%
	Recy	0.85	0.98	0.91	
DS-NP	Pos	0.99	0.98	0.99	99%
	Neg	0.99	0.99	0.99	
DS-BNB	Bio	0.94	0.92	0.93	93%
	Non-Bio	0.92	0.94	0.93	

1.037, 0.317, respectively) . FP were reduced by 43.4% through the implementation of Model M_1 (189 vs. baseline's 334), indicating superior $Real$ detection. This suggests that smaller, curated datasets (like D_{train1}) may mitigate overfitting and improve generalization. A relatively balanced misclassification pattern is exhibited by the CNN baseline. However, M_1 dramatically reduces both FP and, FN, demonstrating high robustness. Conversely, M_2, despite having the lowest FN (61), suffers from very high FP (748), indicating that it fails to detect a significant portion of $Real$ images. This imbalance in M_2 may stem from dataset-specific biases or class imbalance during training.

For Precision, Recall, and F1 Comparison, It can be observed that M_1 consistently demonstrates the most balanced and highest scores across all metrics, M_3 precision is the highest, but its low recall results, and these results suggest M_3 is overly moderate.

The DEV method demonstrates significant performance improvements over baseline CNN model, achieving 7.55% accuracy gain (85.55% $\Rightarrow$ 93.1%) while adding only 15 minutes to the total execution time (8h21m $\Rightarrow$ 8h36m). Notably, error rates sharply decreased, with false positives reduced by 27% (334 $\Rightarrow$ 244) and false negatives cut by 87% (244 $\Rightarrow$ 32), reflecting enhancement in both precision and recall. The confusion matrix reveals stronger class discrimination, with TP rising by 5.4% (1666 $\Rightarrow$ 1756) and TN increasing by 12% (1756 $\Rightarrow$ 1968).

2) Performance evaluation for DS-OR dataset: The baseline CNN model, trained on the full DS-OR dataset (2.5K images), achieved a test accuracy of 85.12%, with a notable test loss of 0.378, (train loss: 0.239, validation loss: 0.345). The confusion matrix revealed 360 FP (Org images misclassified as $Recy$) and 14 FN ($Recy$ misclassified as Org).

In contrast, the DEV method—which splits DS-OR (D_{train}) into three smaller training sets (D_{train1}: 7073, D_{train2}/D_{train3}: 7181 each), with (D_{val}:1129, D_{test}:2513), yielded three models (M_1, M_2 , M_3). Model M_3 emerged as the strongest, achieving 90.89% accuracy with lower test loss (0.317), outperforming both the baseline and other models (M_2: 85.8%, M_3: 84.1% with test loss 0.404, 0.448, respectively) . Experimental data show that false positives were reduced by 48.6% through the implementation of Model M_3 (185 vs. baseline's 360), indicating superior Org detection. For Precision, Recall, and F1 Comparison, it can be observed that M_3 shows the most balanced and highest scores across all metrics.

The DEV method delivered significant performance improvements over whole-dataset training, achieving 4.48% accuracy increase (85.12% ⇒ 89.6%) while adding only 8 minutes to execution time. Error analysis revealed particularly strong gains in precision, with false positives reduced by 34.4% (360 ⇒ 236), though false negatives increased (14 ⇒ 26) due to the dataset's class imbalance. The confusion matrix shows improved TP detection (16.5% increase, 752 ⇒ 876).

3) Performance evaluation for DS-NP dataset: The baseline CNN model, trained on the full DS-NP dataset (30K images), achieved a test accuracy of 95.42%, as detailed in Table IV, with a notable test loss of 0.182, indicating potential overfitting (train loss: 0.1362, validation loss: 0.1041). The confusion matrix revealed 181 FP (Pos images misclassified as Neg) and 2 FN (Neg misclassified as Pos), suggesting challenges in distinguishing high-quality concrete and pavement surfaces images.

In contrast, the DEV method—which splits DS-NP (D_{train}) into three smaller training sets (D_{train1}: 7073, D_{train2}/D_{train3}: 7181 each)—yielded three models (M_1, M_2 , M_3). Model M_1 emerged as the strongest, achieving 98.43% accuracy with lower test loss (0.0537), outperforming both the baseline and other DEV models (M_2: 98.22%, M_3: 98.15%). A significant reduction in FP was achieved by Model M_1 (35 vs. baseline's 181), indicating superior Negative detection. This suggests that smaller, curated datasets (like D_{train2}) may mitigate overfitting and improve generalization. The CNN baseline's balanced misclassification behavior is demonstrated through its FN (28 vs. baseline's 2). Regarding the precision, recall, and F1 comparison, Model M_1 exhibits the most balanced and highest scores across all metrics compared to Model M_2 and Model M_3. This suggests that Model M_1 provides the most robust and reliable performance among the three models.

The DEV method achieved remarkable performance improvements for DS-NP dataset, increasing accuracy by 3.58% (95.42% ⇒ 99%) while adding only 9 minutes and 40 seconds to execution time (3h55m37s ⇒ 4h5m17s). Error analysis shows exceptional optimization, with FP reduced by 83.4% (181 ⇒ 30) and FN increasing by 500% (2 ⇒ 12) due to the model's more conservative classification approach. The confusion matrix reveals outstanding TP detection (8.3% improvement, 1819 ⇒ 1970) with nearly perfect TN retention (1998 ⇒ 1988, -0.5%). These results demonstrate DEV's ability to push model performance to near-perfect levels while maintaining reasonable computational efficiency.

4) Performance evaluation for DS-BNB dataset: The baseline CNN model, trained on the full DS-BNB dataset (65.92K images), achieved a test accuracy of 89%, as detailed in Table IV, with a notable test loss of 0.281, indicating potential overfitting (train loss: 0.058, validation loss: 0.336). The confusion matrix revealed 312 FP (Bio images misclassified as $Non\text{-}Bio$) and 128 FN ($Non\text{-}Bio$ misclassified as Bio), suggesting challenges in distinguishing high-quality Non & Biodegradable material images.

In contrast, the DEV method—which splits DS-BNB (D_{train}) into three smaller training sets (D_{train1}: 19774, D_{train2}/D_{train3}: 20074 each)—yielded three models (M_1, M_2 , M_3). Model M_1 emerged as the strongest, achieving 89.6% accuracy with lower test loss (0.279), outperforming both the baseline and other DEV models (M_2: 83.75%, M_3: 86.83%). A significant reduction in FP was achieved by Model M_1 (192 vs. baseline's 312). Regarding the precision, recall, and F1 comparison, Model M_1 exhibits the most balanced and highest scores across all metrics compared to Model M_2 and Model M_3. This suggests that Model M_1 provides the most robust and reliable performance among the three models.

The DEV method delivered significant performance improvements for DS-BNB dataset, boosting accuracy by 4% (89% ⇒ 93%) with only 12-minute, 8-second increase in execution time (7h27m ⇒ 7h39m8s). The approach demonstrated particularly strong error reduction, with FP decreasing by 48.1% (312 ⇒ 162) while maintaining comparable FN rates (6.25%, 128 ⇒ 120). The confusion matrix reveals excellent TP improvement (8.9% increase, 1688 ⇒ 1838) with stable TN performance (1872 ⇒ 1880, 0.4%). These results highlight DEV's ability to enhance model precision while maintaining recall.

Finally, the DEV method's performance gains stem from four synergistic mechanisms. First, dataset partitioning enables specialized feature learning, where each model captures distinct data aspects, reducing collective bias. Second, the majority voting mechanism cancels individual model errors through consensus. Third, aggregated predictions leverage complementary model strengths, enhancing robustness to outliers and edge cases. Finally, the framework achieves this through a favorable balance between efficiency and accuracy, a small increase in execution time results in a disproportionate increase in accuracy, demonstrating a scalable improvement.

V. Conclusion

The Divided Ensemble Voting (DEV) methodology introduced in this research establishes a novel ensemble learning architecture aimed at augmenting the efficacy of image classification systems. Central to this framework is the strategic division of the training data into unique partitions, enabling the development of specialized convolutional neural networks tailored to distinct feature domains. The consensus-driven integration of predictions via a majority voting protocol yields a robust final classification. Empirical validation across multiple datasets confirmed a marked elevation in model accuracy, with observed gains ranging from 3.58% to 7.55%. The framework also demonstrated a profound capacity for error reduction, achieving a decrease in FP instances of up to 83.4%. These enhancements are attributed to the method's inherent resistance to overfitting, its ability to neutralize singular model biases through diversified training regimes, and its superior generalization powered by collective decision-making. Furthermore, DEV exhibited exceptional proficiency in addressing class distribution imbalances, reflected by a 16.5% increase in TP identification and F1-scores attaining 0.99. A critical advantage of this architecture is its computational efficiency, imposing a negligible runtime overhead of less than 5%, thereby affirming its practicality for operational environments. The combination of heightened precision and operational efficiency positions the DEV framework as a transformative solution for applications demanding high reliability.

REFERENCES

[1] N. Manakitsa, G. S. Maraslidis, L. Moysis, G. F. Fragulis, "A review of machine learning and deep learning for object detection, semantic segmentation, and human action recognition in machine and robotic vision", *Technologies*, vol. 12, no. 2, p. 15, 2024.

[2] R. Archana, P. E. Jeevaraj, "Deep learning models for digital image processing: a review", *Artificial Intelligence Review*, vol. 57, no. 1, p. 11, 2024.

[3] S. A. El-aal, N. I Ghali, "A Proposed Recognition System For Alzheimer's Disease Based On Deep Learning And Optimization Algorithms", *Journal of Southwest Jiaotong University* , vol. 56, no. 5. 2021.

[4] A. Hassan, M. Refaat, A. M. Hemeida, "Image classification based deep learning: A Review", *Aswan University Journal of Sciences and Technology*, vol. 2, no. 1, pp. 11-35, 2022.

[5] Z. H. Arif, M. A. Mahmoud, K. H. Abdulkareem, M. A. Mohammed, M. N. Al-Mhiqani, A. A. Mutlag, R.Damaševičius, "Comprehensive review of machine learning (ML) in image defogging: Taxonomy of concepts, scenes, feature extraction, and classification techniques", *IET Image Processing*, 16(2), pp.289-310, 2022.

[6] B. F. Azevedo, A. M. A. Rocha, A .I. Pereira, "Hybrid approaches to optimization and machine learning methods: a systematic literature review, *Machine Learning*, vol. 113, no. 7, pp.4055-4097, 2024.

[7] I .U .Khan, S .Azam, S .Montaha, A .Al Mahmud, A .R .H .Rafid, M .Z .Hasan, M .Jonkman, "An effective approach to address processing time and computational complexity employing modified CCT for lung disease classification". *Intelligent Systems with Applications*, vol. 16, p.200147, 2022.

[8] G. Menghani, "Efficient deep learning: A survey on making deep learning models smaller, faster, and better" *ACM Computing Surveys*, vol. 55, no. 12, pp.1-37, 2023.

[9] Y. Yu, "Deep learning approaches for image classification", *In Proc. of the 2022 6th international conference on electronic information technology and computer engineering*, pp. 1494-1498, 2022.

[10] S. A. El-aal, R. S. El-Sayed, A. A. Alsulaiman, M. A. Razek, "Using Deep Learning on Retinal Images to Classify the Severity of Diabetic Retinopathy", *International Journal of Advanced Computer Science & Applications*, vol. 15, no.7, pp. 346-355, 2024.

[11] G. Vrbančič, V. Podgorelec, "Transfer learning with adaptive fine-tuning", *IEEE Access*, vol. 8, pp.196197-196211, 2020.

[12] M. Gholizade, H. Soltanizadeh, M. Rahmanimanesh, S. S. Sana, "A review of recent advances and strategies in transfer learning", *International Journal of System Assurance Engineering and Management*, pp.1-40, 2025.

[13] F. Maleki, K. Ovens, R. Gupta, C. Reinhold, A. Spatz, R. Forghani, "Generalizability of machine learning models: quantitative evaluation of three methodological pitfalls", *Radiology: Artificial Intelligence*, vol. 5, no. 1, p.e220028., 2022.

[14] H. E. Kim, A. Cosa-Linan, N. Santhanam, M. Jannesari, M. E. Maros, T. Ganslandt, "Transfer learning for medical image classification: a literature review", *BMC medical imaging*, vol. 22, no. 1, p.69, 2022.

[15] K. He, X. Zhang, S. Ren, J. Sun, "Deep residual learning for image recognition. *In Proc. of the IEEE conference on computer vision and pattern recognition* ,pp. 770-778, 2016.

[16] E .S. Sabry, S. S. Elagooz, F. E. A. El-Samie, N. A. El-Bahnasawy, G. M. El-Banby, R. A. Ramadan, "Evaluation of feature extraction methods for different types of images", *Journal of Optics*, vol. 52, no. 2, pp.716-741, 2023.

[17] Y. C. Huang, C. W. Ho, W. R. Chou, M. Chen, "A framework to predict second primary lung cancer patients by using ensemble models. *Annals of operations research*, vol. 348, no. 1, pp.373-397, 2025.

[18] N. Nakata, T. Siina, "Ensemble learning of multiple models using deep learning for multiclass classification of ultrasound images of hepatic masses", *Bioengineering*, vol. 10, no. 1, p.69, 2023.

[19] Url = "https://zenodo.org/records/5528418#.YpdlS2hBzDd".

[20] Karras, T., Laine, S. and Aila, T., 2019. "A style-based generator architecture for generative adversarial networks. In Proceedings of the IEEE/CVF conference on computer vision and pattern recognition (pp. 4401-4410).

[21] Yasin, E.T. and Koklu, M., 2023, April. Classification of organic and recyclable waste based on feature extraction and machine learning algorithms. In Proceedings of the International Conference on Intelligent Systems and New Applications (ICISNA'23).

[22] N. Nnamoko,J. Barrowclough,J. Procter, "Waste Classification Dataset", *Mendeley Data*, 2022, https://doi.org/10.17632/n3gtgm9jxj.2.

[23] O. Omoebamije, Concrete & Pavement Crack Dataset, *Mendeley Data*, 2023, https://doi.org/10.17632/429vzbgmbx.1.

[24] M. Srimathi, M. Srivani, K. P. Peeyush, "Deep learning based waste material classification", *In Proc. of 2023 IEEE Silchar Subsection Conference (SILCON)*, pp. 1-6, 2023.

[25] https://www.kaggle.com/datasets/rayhanzamzamy/non-and-biodegradable-waste-dataset.