Int. j. inf. tecnol. (February 2023) 15(2):557–564
https://doi.org/10.1007/s41870-022-01149-8

ORIGINAL RESEARCH

Weighted ensemble model for image classification

Talib Iqball[1] · **M. Arif Wani**[1]

Received: 28 March 2022 / Accepted: 20 September 2022 / Published online: 23 January 2023

Abstract The Deep Convolutional Neural Network (DCNN) classification models are being tremendously used across many research fields including medical science for image classification. The accuracy of the model and reliability on the results of the model are the key attributes which determine whether a particular model should be used for a specific application or not. A highly accurate model is always desirable for all applications of machine learning as well as deep learning. This paper presents a DCNN based heterogeneous ensemble approach where all DCNN models can be trained on a single dataset and each model can contribute of towards the final output of the ensemble model. The contribution of each model is weighted according to its individual accuracy on the given dataset. Models with higher accuracy has higher contribution in the final output of ensemble model, whereas the models with lower accuracy has lower contribution. This approach, when tested on two different X-ray images datasets of Covid-19, has confirmed the significant increase in 3-class accuracy as compared to the models in literature.

Keywords Deep learning · Weighted deep ensemble model · Ensemble learning · Image classification

✉ Talib Iqball
talibiqbal247@gmail.com

M. Arif Wani
awani@uok.edu.in

[1] Department of Computer Science, University of Kashmir, Srinagar, India

1 Introduction

In ensemble learning, we use more than one model, generally called baseline models and the output of the all models are combined together to produce a single combined output. That is, we are combining the predictions/decisions of various models to produce the final prediction or decision. The bootstrapping and aggregation of the constituent baseline models are done in various ways, which result in various different ensemble strategies. Bagging [1] is one of the primary strategies for building ensemble-based algorithms which is also known as bootstrap aggregating and is used to improve the performance of an ensemble classifier. Bagging has two main sub-processes: bootstrapping and aggregation. In bootstrapping, we divide the original dataset into subsets (also called bagging samples), and each base learner is trained on one of these subsets or bagging samples. These bagging samples may or may not be overlapped. Now the base learners create an independent observations of same size and aggregation process combines these observations (commonly by voting for classification) to create a single observation which is better than the observation of a single model. Random forest [2] is the implementation of the bagging. Boosting is another ensemble method of combining the weak learners to form a strong learner. The weak learners (also called the decision stumps) are placed in linear combination one after another. The examples that are misclassified by the first stump are prioritized and are taken care of by the next stump and so on. AdaBoost [3] is the implementation of boosting algorithm. Stacking [4] on the other hand is a technique of minimizing the generalization error rate of one or more than one models or generalizers. Stacking, also called stacked generalization, uses multiple base learners to generate intermediate generalizations and

these generalizations are passed to the next models to make a global generalization.

Most of the techniques in ensemble modelling combine different models to produce a global generalization of the distribution. In case of heterogeneous mix of models, each model has its own generalization parameters, which makes it obvious that there may be varied generalization of the constituent models, i.e. each model will perform differently on different data distributions. The unweighted ensemble modelling is very common approach which is used to aggregate the generalizations of base learners like random forest [2]. The problem with this approach is that in a heterogeneous ensemble model, there may be suboptimal performance of the model due to unweighted averaging [5]. We propose a deep weighted ensemble approach for heterogeneous base learners which fuses their generalization by a weighted sum of their outputs. We used state-of-art DCNN architectures as base learners for the image classification task. We choose different models because we do not do the bagging, instead we train the base learners over the full dataset. Each model with its unique design has capability to generalize the distribution its own way. We picked ResNet101 [6], InceptionV3 [7], MobileNetV2 [8], NasNet [9] and Xception [10] models as our base learners. We fuse the outputs of these models together with given weight by our algorithm. Then weighted sum is used for the prediction of the ensemble model. We tested our model with two different datasets, which contained the normal, Covid-19 and pneumonia chest X-rays as discussed in the following sections.

2 Related work

Ensemble models have seen a pretty good success is the recent past. Various approaches have been proposed to ensemble or combine the different models [11]. References [6, 12] suggest that we can have a decent ensemble model for similar base learners (or homogeneous ensemble model) by combining these base learners by unweighted averaging. But in case of different base learners i.e. heterogeneous ensemble model, unweighted averaging may result in substandard performance of the model [5]. Unweighted average and majority voting are comparable. Rather than averaging the output probability, it counts the votes for all of the projected labels from the base learners and produces a final prediction using the label with the most votes. Or, to put it another way, it takes an unweighted average of the labels from basic learners and picks the one with the highest value [5]. The pair-wise dependencies between classifiers play a vital role in majority voting [13] and hence an ensemble model of shallow networks is best suited for majority voting than an ensemble model of deep networks [5]. Another method of combining the models is stacking [4] which minimizing the generalization error rate of one or more than one models or generalizers. Stacking, also called stacked generalization, uses multiple base learners to generate intermediate generalizations and these generalizations are passed to the next models to make a global generalization. A weighted ensemble model called super learner [14] is an extension of stacking technique. It computes the weights based of V-fold cross validation to select the best in library of models. From the application point of view, we have a lot of applications of ensemble methods. We only focused on covid-19 and pneumonia classification as we only tested our model on those datasets. Deep learning has been extensively used in the task of Covid-19 and Pneumonia detection using X-Ray images. The deep learning models tend to consume more data for training the models [15], transfer learning was to cope with issue according to recent studies. A classification model [16] was proposed to classify normal, pneumonia and Covid-19 X-rays. This method is actually based on Darknet-19 architecture [17]. In [18], the authors use a chest X-ray to differentiate between COVID and non-COVID patients. They did it using a pre-trained DenseNet model. It aided them in prioritizing the selection of patients for further RT-PCR testing, which they believe is important in an in-patient situation when current systems are unsure whether to retain the patient in the ward with other patients or segregate them in COVID-19 sections. In [19], The authors introduced CoroNet, a model based on the pre-trained Xception model [10]. COVID, Pneumonia, and Normal X-Rays were classified. SqueezeNet [20] is a lightweight model that is fine-tuned for COVID diagnosis with Bayesian optimization additive [21]. It has the same high accuracy as AlexNet but has 50X less parameters. Fine-tuned hyper-parameters and augmented dataset make their proposed network perform much better than existing network designs and to obtain a higher COVID-19 diagnosis accuracy. The accuracy of their models is compared in the next section. However, the authors of [22], stated that employing correctly optimised Generative models for data augmentation can also increase classification model accuracy. In [23], X-ray images were pre-processed using fuzzy color technique and the images that were structured with the original images were stacked. Then the authors used the stacked dataset for training the deep learning models (MobileNetV2, SqueezeNet) and the feature sets that were obtained by the said models were processed using the social mimic optimization method. Thereafter, efficient features were combined and classified using support vector machines (SVM). One of the two papers that we focused on in case of Covid-19 classification is InstaCovNet [24]. InstaCovNet-19 is a deep convolutional architecture (DCNN) that uses chest X-ray images to detect individuals with COVID-19. The authors chose to adopt transfer learning instead of starting the training from scratch because the later may be inefficient and may lead to lousy variance. The authors

used five different models trained on the common dataset which are: Inception v3 [7], MobileNetV2 [8], ResNet101 [6], NASNet [9] and Xception [10]. Authors chose these models after thorough experimentation, which concluded that these models are best in feature extraction and have unique features. Initially, these models were imported with weights (trained on ImageNet) and then these models were fine-tuned on the mentioned dataset. The fine-tuned models were then combined using the integrated stacking [5] technique, making the stacked model a larger and more robust model. Images are initially transmitted via the five heads of the input layer in the classification process, i.e., the five copies of the picture are supplied as input. The photos are then sent through various models and processed as needed. Each pre-trained model's final convolution layers are fine-tuned once again, resulting in forecasts from each model, which are then merged via a stacking layer. The combined output is now transmitted through a dense layer of 128 nodes, where the stacked model learns how to use the predictions and what modifications are to make in the sub-final model's convolution layer. The results of the final dense layer are then transmitted through a dense layer of three nodes. To produce predictions in form probability, SoftMax activation (sigmoid activation for binary classification) is utilized. The second paper that we compared our results to is DarkNet [16]. The authors started with Darknet-19 model as the starting point. The DarkNet classifier is used on the basis of this successful architecture. In comparison to the original DarkNet architecture, the authors employed fewer layers and filters. They steadily raised the number of filters, from 8 to 16 to 32. Each Dark-Net layer contains a single convolutional layer, Batch Normalization layer, and Leaky-ReLU function, and every 3-Conv layer has the same configuration three times in succession. The batch normalization operation is used to standardize the inputs, and it has other advantages such as reducing training time and improving model stability. LeakyReLU is a variant of the ReLU procedure that prevents neurons from dying. Unlike ReLU and sigmoid activation functions, which have zero value in the negative half of their derivatives, LeakyReLU features a tiny epsilon value to avoid the problem of dying neurons. The Maxpool approach is utilised in all pooling processes, similar to the Darknet-19 concept. Maxpool reduces the size of an input by taking the maximum of a filter-defined zone. The authors in [25] worked with ensemble model to predict the Marathi handwritten digits and used stacked ensemble in which they concatenated the generalizations of models in the lobby, which proved to be efficient that the models in literature.

Machine learning and deep learning have been extensively used for image classification [26], sequence to sequence classification, etc. [27]. There are other techniques as well which have performed well and have been extended to medical imaging. The authors in [28] has adopted an enforced block diagonality approach for the classification medical image patterns. The experiment was executed on the several medical datasets, which include Covid and non-Covid CT images, cell images of leukaemia and non-leukaemia patients, breast cancer dataset, etc. The induction of dictionary learning in the approach has improved its performance over some of the related techniques used for the classification of medical patterns. While others [29] have used clustering methodology by leveraging transcriptomic data for sub-typing cancer patients and is based on a non-linear dimensionality reduction technique called uniform manifold approximation and projection (UMAP) and a tool from algebraic topology called mapper. Inspired by the success of hybrid and ensemble methods for developing enhanced prediction methods, we considered ensemble approach for classification of chest X-ray images.

3 Proposed method

In this paper, we propose an ensemble approach to classify image data which is pictorially presented in Fig. 1. This approach can be extended to any library of models in an ensemble. The following subsections explain our proposed method.

3.1 Selecting the base learners

The selection of base learner may vary from one problem to another, but the essence is to select most appropriate models for the given problem. We selected five different models for classification, which are the state-of-art CNN architectures, as base learners. These base learners are ResNet101 [6], InceptionV3 [7], MobileNetV2 [8], NasNet [9] and Xception [10]. Due to their varied structure, they possess different capability to generalize the given distribution.

3.2 Training the models and their weight factor calculation

After all the models are selected, all the models are individually trained on the dataset and their percentage accuracies acc_j are recorded on validation set (Note: This validation set was not used during training). As all the models are different and they use different hyper-parameters, they learn the same given distribution differently. The models can be trained up to their convergence or up to when there is no decrease in loss value. Now as the models would be trained then these models will be evaluated on validation dataset and accuracies are recorded and these recorded accuracies are used to generate the weight factor 'α_j' using the Eqs. (1) and (2) for each respective model. Here 'j' varies from 1 to m where m being the number of models. In Eq. (2), we add 1 to the α_j

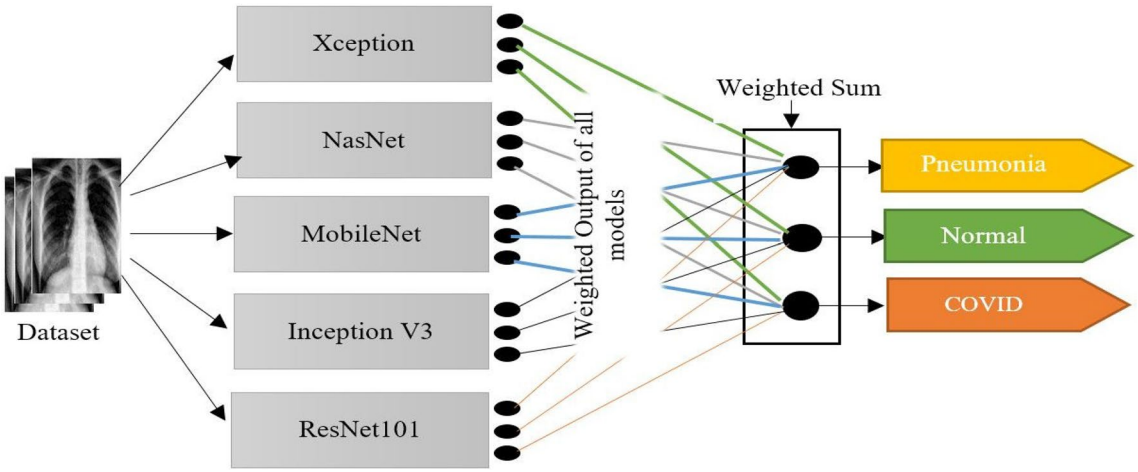

Fig. 1 Proposed weighted ensemble model for X-ray classification

because β will be accuracy of least accurate model and when substituted in Eq. (1) to give the weight $\alpha_j = 1$ to least accurate model, otherwise the least accurate model will be cut off from contributing towards the final output of ensemble model by computing $\alpha_j = 1$.

$$\alpha_j = acc_j - \beta + 1, \quad j = 1 \ldots m \tag{1}$$

$$\beta = min\left(\left[acc_j \right]_{j=1}^{m} \right) \tag{2}$$

3.3 Ensemble aggregation of models

Now, the trained models are used to make a generalization or prediction for the input sample. The output of each model after '*softmax*' function is taken. The output of the model for each input sample T_i is a probability vector C_k, where 'k' ranges from$l \ldots nc$, 'nc' represents number of classes. The output of the model $\left[M_j\right]_{j=1}^{m}$ is multiplied with its respective multiplying factorα_j. For any given class, the weighted probabilities are summed together which serves as the output for the ensemble model for that class as shown in Eq. (3). The algorithm for weighted ensemble model is shown in Algorithm 1, which makes it easy to understand and implement. In Fig. 1, as each model will have different multiplying factors (α_j), which are based on the individual accuracies of the models, they are represented by different color line. For a given model, α_j remains same. The final output for a given class is prepared with Eq. (3).

$$OutputClass_{T_i,ensemble} = max\left[\sum_{j=1}^{m} \left(\alpha_j \times \left[C_k \right]_{i,j} \right) \right]_{k=1}^{nc} \tag{3}$$

Algorithm 1 provides the set of steps for proposed weighted ensemble model. As described in Algorithm 1, we prepare our dataset split it into training, validation and testing. We load the first portion as training data into $D = \{(x_1, y_1), \ldots, (x_h, y_h)\}$ where x_h is the set of feature vectors and y_h the corresponding label. Then a part of the dataset is used for validation loaded into V and rest of the data is used for testing and is loaded into T. As we have m number of models, we train then them one by one on the training dataset D. In our case $m = 5$. After training, we used the each model to give their predictions on the validation data set $\left[V_i\right]_{i=1}^{g}$. We use these predictions to compute the accuracy of the model M_j. Then we calculate the parameter β, which helps us to compute the weight factor α_j for each model. We use the α_j as multiplying factor to the individual model predictions. The weighted predictions from various models are summed, which serve the output of the proposed weighted ensemble model (WEM). Now, in order to use that output for prediction, we apply *argmax* to the output of the proposed WEM, which gives us the class of the model predicted by the WEM for the given sample. Or we can achieve the same results if we give output of WEM to *softmax* function for class determination.

Algorithm 1: Weighted ensemble model

Input:	Datasets $D = \{(x_1, y_1), \ldots, (x_h, y_h)\}$, $V = \{(x_1, y_1), \ldots, (x_g, y_g)\}$ and $T = \{(x_1, y_1), \ldots, (x_n, y_n)\}$ for training, validation and testing respectively.
	M_j representing jth model, $j = 1 \ldots m$ to classify the data into nc classes.

Algorithm 1: Weighted ensemble model

Output: C_k is a vector of probabilities, $k = 1 \ldots \text{nc}$
$OutputClass_{T_i,M}$ is a vector representing the output class of T_i using model M.

1. For $j = 1 \ldots m$, do
   (a) Train model M_j with dataset D
   (b) Compute the class probabilities for $V_i, i = 1 \ldots g$ using model M_j $\left[C_k\right]_{i,j} = M_j\left(V_i\right)$
   (c) Compute output class of $V_i, i = 1 \ldots g$ using $\left[C_k\right]_{i,j}$ $OutputClass_{V_i,M_j} = max\left[C_k\right]_{i,j}$
   (d) Compute the accuracy acc_j of model M_j on V using $\left[OutputClass_{V_i,M_j}\right]_{i=1}^{g}$ against *True-Label*.
2. Calculate parameter $\beta = min(\left[acc_j\right]_{j=1}^{m})$
3. For $j = 1 \ldots m$, do
   (a) Calculate weight of model M_j : $\alpha_j = acc_j - \beta + 1$
4. Compute the output of proposed model for $T_i, i = 1 \ldots n$ $OutputClass_{T_i,ensemble} = max\left[\sum_{j=1}^{m}\left(\alpha_j \times [C_k]_{i,j}\right)\right]_{k=1}^{nc}$
5. Compute the accuracy $acc_{ensemble}$ using $[OutputClass_{T_i,ensemble}]_{i=1}^{n}$ against True-label.

3.4 Implementation details

The main inspiration behind the creation of our proposed model is to make a robust classification model which, irrespective of the domain where it is used, will perform the best. To test the model's performance, we tested it two different datasets for the classification of Pneumonia, Covid-19 and Normal X-ray images. We chose this dataset because of the need of an hour. The following section explains the implementation of weighted ensemble model for the mentioned datasets.

As we mentioned in earlier section that we choose five different classification models viz. ResNet101, Inception, MobileNetV2, NasNet and Xception. As the datasets provided are not large enough for training these models from scratch, transfer learning is adopted. We used these models with along the pre-trained weights from 'imagenet'. For each architecture, we used their weights when they were trained on 'imagenet' dataset, excluding the top. We flattened the last layers of each individual model and added a dense layer of 128 units with a 'reLu' activation. Lastly we added a dense layer with 3 output units for three classes of data with a '*softmax*' activation function. The three outputs (referring to three classes of data) of the model given by the probability vector C_k, which is the probability of input T_i. Each model is separately trained up to 100 epochs with Adam optimizer and learning rate of 0.001, which actually fine-tunes the model for this specific application.

4 Results and discussion

In this section, we firstly discuss the datasets that we used during the training and testing of the models. Then we discuss the results of our model on the given datasets and also the comparison of our model with other models that used the same dataset.

4.1 Dataset

The data was gathered from the COVID-19 Radiography Database on Kaggle by [30] and the Chest X-ray dataset by [31]. The creators of these datasets upload fresh photographs to them in order to enhance the size of the dataset. We employed random sampling to balance the merged dataset because it was previously skewed. 361 COVID-19 Images, 365 Normal class Images, and 362 Pneumonia class Images were randomly picked for the experiment during the random-sampling procedure. Both datasets were merged by [24], and the results are summarized in Table 1 and this dataset will be referred as Dataset-1. We chose 80% of the 1088 total photos for training purposes and 20% for testing purposes. In second selected paper, the dataset that the authors have used have been collected from two different sources. The complete dataset is available at the following link: https://github.com/muhammedtalo/COVID-19. There are now 125 X-ray images (43 female instances and 82 male instances) in the database that have been diagnosed with COVID-19. In the database, that have been found to be positive. There isn't enough information for all of the patients in this dataset. The average age of 26 COVID-19 positive participants is around 55 years old, according to the age information provided. In addition, pictures of normal and pneumonia were available from the ChestX-ray8 database. To circumvent the problem of imbalanced data, the authors that used this dataset, randomly selected 500 no-findings and

Table 1 Summary of dataset

Class	Dataset-1		Dataset-2	
	Training	Testing	Training	Testing
COVID-19	289	72	100	25
Pneumonia	289	73	400	100
Normal	292	73	400	100
Total	870	218	900	225
Total images in dataset	1088		1125	

Table 2 Summary of test accuracies of different models

Model	Class-1 accuracy (COVID)	Class-2 accuracy (Normal)	Class-3 accuracy (Pneumonia)	Total (AVG)
Xception	0.9861	0.8904	1.0000	0.9587
InceptionV3	0.9722	0.9452	0.9726	0.9633
MobileNetV2	0.9861	0.9726	0.9726	0.9771
ResNet-101	0.9583	0.9855	0.9589	0.9495
NasNet	0.9861	0.9863	0.9452	0.9725
Weighted-ensemble model	1.0000	1.0000	1.0000	1.0000

Table 3 Results on weighted ensemble model on second dataset

Model	Class-1 accuracy (COVID)	Class-2 accuracy (Normal)	Class-3 accuracy (Pneumonia)	Total (AVG)
DarkNet [16]				
Xception	0.76	0.76	0.76	0.7600
InceptionV3	0.60	0.80	0.80	0.7778
MobileNetV2	0.88	0.84	0.83	0.8400
ResNet-101	0.88	0.73	0.68	0.7244
NasNet	0.72	0.74	0.79	0.7600
Weighted-ensemble model	0.96	0.95	0.93	0.9466

500 pneumonia class frontal chest X-ray pictures from this database. We took the dataset exactly in the same manner as they did for our experiments. We will be referring this dataset as Dataset-2.

4.2 Results

First, we trained each model in the ensemble library on Dataset-1 and their accuracies where recorded on the validation data (Note: this validation set was not used in training). Based on the validation accuracies, each model M_j has the respective weight α_i as shown in the Table 6. Then after obtaining the weights for each model, we used test data to evaluate our model. We made predictions with our ensemble lobby and the predictions of the models are multiplied with the obtained weight of the model. Weighted sum of the predictions serve the output of our model. Here we applied *argmax*() to find the prediction of ensemble model. The results on the Dataset-1 are shown in Table 2. The same experiment was done on Dataset-2 which yielded the results as shown in Table 3. The confusion matrices of the weighted ensemble model are shown in Tables 4 and 5, respectively.

The comparison of our model with other ensemble model trained on same dataset are given in Table 7. At the time of writing this article, InstaCovNet [24] is the best ensemble models that we compare our work with. They have used the same dataset as we do and is shown in Table 7. Rest of the models have used the same dataset and we are comparing our model with them as well. Using Dataset-2, the best performing model was Darknet [16], we tested our model on that dataset as well and results are shown in the Table 7.

Table 4 Confusion matrix of WEM on Dataset-1

	Predicted COVID-19	Predicted Normal	Predicted Pneumonia
Actual COVID-19	72	0	0
Actual normal	0	73	0
Actual pneumonia	0	0	73

Table 5 Confusion matrix of WEM on Dataset-2

	Predicted COVID-19	Predicted Normal	Predicted Pneumonia
Actual COVID-19	24	0	1
Normal	0	95	5
Pneumonia	0	7	93

Table 6 After training the models individually, the training and validation accuracies and the weights given to each model

Model	Training accuracy	Validation accuracy	α_i
Xception	1.0000	0.9587	5.57
InceptionV3	1.0000	0.9633	6.33
MobileNetV2	1.0000	0.9771	7.71
ResNet-101	0.9723	0.9495	4.95
NasNet	1.0000	0.9725	7.25

Table 7 Comparison table weighted ensemble model with other recent models used to classify the same data

Model	3-Class accuracy	References	
COVID-Net	0.933	[32]	Dataset-1
CoroNet	0.896	[19]	
COVIDiagnosis-Net	0.983	[21]	
MobileNet V2	0.9472	[33]	
CovidAID	0.923	[18]	
MobileNetV2, SqueezeNet and SVM	0.9927	[23]	
InstaCovNet-19	0.9908	[24]	
Weighted ensemble model	**1.0000**	Proposed model	
DarkNet	0.8702	[16]	Dataset-2
Weighted ensemble model	**0.9466**	Proposed model	

Bold numbers indicate best performance

5 Conclusion and future scope

After studying various ensemble models, we discovered that even some models in ensemble has a low accuracy individually, it still equally contributes to the output of the ensemble model. Thus we proposed our model, in which each constituent model of ensemble will get a weight which is proportional to its individual accuracy. Weighted ensemble model is ensemble approach which fuses the generalizations of the models in the library of ensemble after the softmax layer. The weight is given to a particular model w.r.t. its accuracy on the given dataset. This helps us to generalize the distribution to a great extent. No matter, which model will perform better in different distributions, our model will give highest weight to the best performer. The future work may include the application of this model to various applications. Also, they may have a different lobby of models in an ensemble.

Acknowledgements The authors are thankful to the Department of Computer Science, University of Kashmir for acquiring High performance NIVIDA AI Server (DGX A100) under RUSA 2.0 grant and providing us the access to it for smooth conduction of experiments. Furthermore, the authors want to thank anonymous referees for their support and input.

Funding No funding was received.

References

1. Breiman L (1996) Bagging predictors. Mach Learn 24(2):123–140. https://doi.org/10.1007/BF00058655
2. Breiman L (2001) Random forests. Mach Learn 45(1):5–32. https://doi.org/10.1023/A:1010933404324
3. Freund Y, Schapire RE, Hill M (1996) Experiments with a New Boosting Algorithm Rooms f 2B-428, 2A-424 g
4. Wolpert DH (1992) Stacked generalization. Neural Netw 5(2):241–259. https://doi.org/10.1016/S0893-6080(05)80023-1
5. Ju C, Bibaut A, van der Laan M (2018) The relative performance of ensemble methods with deep convolutional neural networks for image classification. J Appl Stat 45(15):2800–2818. https://doi.org/10.1080/02664763.2018.1441383
6. He K, Zhang X, Ren S, Sun J (2016) Deep residual learning for image recognition. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition, vol 2016-December, pp 770–778. https://doi.org/10.1109/CVPR.2016.90
7. Szegedy C, Vanhoucke V, Ioffe S, Shlens J, Wojna Z (2016) Rethinking the inception architecture for computer vision. In: Proceedings of the IEEE computer society conference on computer vision and pattern recognition, vol 2016-December, pp 2818–2826. https://doi.org/10.1109/CVPR.2016.308
8. Sandler M, Howard A, Zhu M, Zhmoginov A (2018) MobileNetV2: inverted residuals and linear bottlenecks mark, pp 4510–4520
9. Zoph B, Vasudevan V, Shlens J, Le QV (2018) Learning transferable architectures for scalable image recognition. In: Proceedings of the IEEE computer society on conference computer vision and pattern recognition, pp 8697–8710. https://doi.org/10.1109/CVPR.2018.00907
10. Chollet F (2017) IEEE conference on computer vision and pattern recognition (CVPR), 2017, pp 1800–1807. https://doi.org/10.1109/CVPR.2017.195
11. Sofi MA, Wani MA (2022) Improving prediction of protein secondary structures using attention-enhanced deep neural networks. In: 9th international conference on computing for sustainable global development (INDIACom), 2022, pp 664–668. https://doi.org/10.23919/INDIACom54597.2022.9763114
12. Simonyan K, Zisserman A (2015) Very deep convolutional networks for large-scale image recognition. CoRR, vol. abs/1409.1
13. Kuncheva LI, Whitaker CJ, Shipp CA, Duin RPW (2003) Limits on the majority vote accuracy in classifier fusion. Pattern Anal Appl 6(1):22–31. https://doi.org/10.1007/s10044-002-0173-7
14. van der Laan M, Polley E, Hubbard A (2008) Super Leaner. UC Berklely Div. Biostat. Work. Pap. Ser.
15. Wani MA, Bhat FA, Afzal S, Khan AI (2019) Advances in deep learning, vol 57
16. Ozturk T, Talo M, Yildirim EA, Baloglu UB, Yildirim O, Rajendra Acharya U, Ozturk T, Talo M, Yildirim EA, Baloglu UB, Yildirim O, Rajendra Acharya U (2020) Automated detection of COVID-19 cases using deep neural networks with X-ray images. Comput Biol Med 121:103792. https://doi.org/10.1016/j.compbiomed.2020.103792
17. Redmon J, Farhadi A (2017) YOLO9000: Better, Faster, Stronger Joseph. *Cvpr* no. April, pp 187–213, 2016, [Online]. Available: https://doi.org/http://www.worldscientific.com/doi/abs/10.1142/9789812771728_0012

18. Mangal A et al (2020) CovidAID: COVID-19 Detection Using Chest X-Ray, pp 1–10, [Online]. Available: https://doi.org/http://arxiv.org/abs/2004.09803
19. Khan AI, Shah JL, Bhat MM (2020) CoroNet: a deep neural network for detection and diagnosis of COVID-19 from chest x-ray images. Comput Methods Programs Biomed 196:105581. https://doi.org/10.1016/j.cmpb.2020.105581
20. Iandola FN, Han S, Moskewicz MW, Ashraf K, Dally WJ, Keutzer K (2016) SqueezeNet: AlexNet-level accuracy with 50x fewer parameters and <0.5 MB model size, pp 1–13, [Online]. Available: https://doi.org/http://arxiv.org/abs/1602.07360
21. Ucar F, Korkmaz D (2020) COVIDiagnosis-Net: deep Bayes-SqueezeNet based diagnosis of the coronavirus disease 2019 (COVID-19) from X-ray images. Med Hypotheses 140:109761. https://doi.org/10.1016/j.mehy.2020.109761
22. Ali Y, Wani MA (2021) Improving training of generative adversarial networks. In: 8th international conference on computing for sustainable global development (INDIACom), 2021, pp 81–86. https://doi.org/10.1109/INDIACom51348.2021.00016
23. Toğaçar M, Ergen B, Cömert Z (2020) COVID-19 detection using deep learning models to exploit Social Mimic Optimization and structured chest X-ray images using fuzzy color and stacking approaches. Comput Biol Med. https://doi.org/10.1016/j.compbiomed.2020.103805
24. Gupta A, Gupta S, Katarya R (2020) InstaCovNet-19: A deep learning classification model for the detection ofCOVID-19 patients using Chest X-ray. ELSEVIER no January 13. https://doi.org/10.1016/j.asoc.2020.106859
25. Mane DT, Tapdiya R, Shinde SV (2021) Handwritten Marathi numeral recognition using stacked ensemble neural network. Int J Inf Technol 13(5):1993–1999. https://doi.org/10.1007/s41870-021-00723-w
26. Iqball T, Wani MA (2021) X-ray images dataset augmentation with progressively growing generative adversarial network. In: 8th international conference on computing for sustainable global development (INDIACom), 2021, pp 93–97. https://doi.org/10.1109/INDIACom51348.2021.00018
27. Sofi MA, ArifWani M (2021) Improving prediction of amyloid proteins using secondary structure based alignments and segmented-PsSm. In: 8th international conference on computing for sustainable global development (INDIACom), 2021, pp 87–92. https://doi.org/10.1109/INDIACom51348.2021.00017
28. Sheikh IM, Chachoo MA, Rather AA (2022) An efficient biomedical cell image fusion method based on the multilevel low rank representation. Int J Inf Technol. https://doi.org/10.1007/s41870-022-01002-y
29. Rather AA, Chachoo MA (2022) UMAP guided topological analysis of transcriptomic data for cancer subtyping. Int J Inf Technol. https://doi.org/10.1007/s41870-022-01048-y
30. Chowdhury MEH et al (2020) Can AI help in screening viral and COVID-19 pneumonia? IEEE Access 8:132665–132676. https://doi.org/10.1109/ACCESS.2020.3010287
31. Maguolo G, Nanni L (2019) A critic evaluation of methods for COVID-19 automatic detection from X-ray images, pp 1–10
32. Gunraj H, Wang L, Wong A (2020) COVIDNet-CT: a tailored deep convolutional neural network design for detection of COVID-19 cases from chest CT images. Front Med 7:608525. https://doi.org/10.3389/fmed.2020.608525
33. Apostolopoulos ID, Mpesiana TA (2020) Covid-19: automatic detection from X-ray images utilizing transfer learning with convolutional neural networks. Phys Eng Sci Med 43(2):635–640. https://doi.org/10.1007/s13246-020-00865-4