Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website.

Informatics in Medicine Unlocked 36 (2023) 101158

Contents lists available at ScienceDirect

Informatics in Medicine Unlocked

journal homepage: www.elsevier.com/locate/imu

Swin-textural: A novel textural features-based image classification model for COVID-19 detection on chest computed tomography

Ilknur Tuncer [a], Prabal Datta Barua [b,c], Sengul Dogan [d,*], Mehmet Baygin [e], Turker Tuncer [d], Ru-San Tan [f,g], Chai Hong Yeong [h], U. Rajendra Acharya [i,j,k]

[a] Elazig Governorship, Interior Ministry, Elazig, Turkey
[b] School of Business (Information System), University of Southern Queensland, Toowoomba, QLD, 4350, Australia
[c] Faculty of Engineering and Information Technology, University of Technology Sydney, Sydney, NSW, 2007, Australia
[d] Department of Digital Forensics Engineering, College of Technology, Firat University, Elazig, Turkey
[e] Department of Computer Engineering, Faculty of Engineering, Ardahan University, Ardahan, Turkey
[f] Department of Cardiology, National Heart Centre Singapore, Singapore
[g] Duke-NUS Medical School, Singapore
[h] School of Medicine, Faculty of Health and Medical Sciences, Taylor's University, 47500, Subang Jaya, Malaysia
[i] Ngee Ann Polytechnic, Department of Electronics and Computer Engineering, 599489, Singapore
[j] Department of Biomedical Engineering, School of Science and Technology, SUSS University, Singapore
[k] Department of Biomedical Informatics and Medical Engineering, Asia University, Taichung, Taiwan

ARTICLE INFO

Keywords:
Swin
Textural feature extraction
Swin-textural
COVID-19 classification
Computed tomography images

ABSTRACT

Background: Chest computed tomography (CT) has a high sensitivity for detecting COVID-19 lung involvement and is widely used for diagnosis and disease monitoring. We proposed a new image classification model, swin-textural, that combined swin-based patch division with textual feature extraction for automated diagnosis of COVID-19 on chest CT images. The main objective of this work is to evaluate the performance of the swin architecture in feature engineering.

Material and method: We used a public dataset comprising 2167, 1247, and 757 (total 4171) transverse chest CT images belonging to 80, 80, and 50 (total 210) subjects with COVID-19, other non-COVID lung conditions, and normal lung findings. In our model, resized 420 × 420 input images were divided using uniform square patches of incremental dimensions, which yielded ten feature extraction layers. At each layer, local binary pattern and local phase quantization operations extracted textural features from individual patches as well as the undivided input image. Iterative neighborhood component analysis was used to select the most informative set of features to form ten selected feature vectors and also used to select the 11th vector from among the top selected feature vectors with accuracy >97.5%. The downstream kNN classifier calculated 11 prediction vectors. From these, iterative hard majority voting generated another nine voted prediction vectors. Finally, the best result among the twenty was determined using a greedy algorithm.

Results: Swin-textural attained 98.71% three-class classification accuracy, outperforming published deep learning models trained on the same dataset. The model has linear time complexity.

Conclusions: Our handcrafted computationally lightweight swin-textural model can detect COVID-19 accurately on chest CT images with low misclassification rates. The model can be implemented in hospitals for efficient automated screening of COVID-19 on chest CT images. Moreover, findings demonstrate that our presented swin-textural is a self-organized, highly accurate, and lightweight image classification model and is better than the compared deep learning models for this dataset.

* Corresponding author.
E-mail addresses: ilknur.tuncer@icisleri.gov.tr (I. Tuncer), prabal.barua@usq.edu.au (P.D. Barua), sdogan@firat.edu.tr (S. Dogan), mehmetbaygin@ardahan.edu.tr (M. Baygin), turkertuncer@firat.edu.tr (T. Tuncer), tanrsnhc@gmail.com (R.-S. Tan), Chaihong.yeong@taylors.edu.my (C.H. Yeong), aru@np.edu.sg (U.R. Acharya).

https://doi.org/10.1016/j.imu.2022.101158
Received 13 December 2022; Received in revised form 30 December 2022; Accepted 30 December 2022
Available online 31 December 2022

1. Introduction

The COVID-19 (Coronavirus disease 2019) pandemic has claimed millions of lives worldwide since it is a global pandemic [1,2]. The virus can invade the lungs, resulting in structural alterations in lung tissues that may be detected on medical imaging. Computed tomography (CT) has been used to diagnose COVID-19 and monitor patients' progress [3]. Several studies have demonstrated high sensitivity of chest CT for COVID-19 detection, with reports suggesting that CT abnormalities may precede positive virological assay results [4,5]. Chest CT also allows physicians to assess the pathological condition of the lungs, stage the disease, and formulate a treatment plan for the patient [6]. Not surprisingly, interest in chest CT for diagnosing and managing COVID-19 patients has grown apace. To harmonize the assessment and reporting of COVID-19 lung involvement in patients on CT images, the consensus COVID-19 Reporting and Data System has been developed [7], which uses a scaling method to visually evaluate CT images. However, this process can be manually intensive and is subject to human biases.

To overcome the limitations of manual reading of medical images, various machine learning methods have been proposed to facilitate automated detection of COVID-19 using medical images, including CT chest (Table 1).

Deep learning methods are frequently employed in deep models, with the majority of the studies reporting classification accuracy rates of over 90%. However, the computational complexities of these deep models are high, which make real-world implementation challenging. Hence, we have been motivated to develop an accurate yet efficient model for image-based COVID-19 classification using handcrafted feature engineering. To this end, we have been inspired by the reported success of computer vision-based image classification models like vision transformer [8], multilevel perceptron-mixer (MLP-mixer) [9], ConvMixer [10], and Swin transformer [11], use the strategy of dividing the input images into patches for multilevel downstream feature extraction. In this study, we adapted the Swin transformer-based patch division to build a new handcrafted feature engineering chest CT image classification model for automated COVID-19 diagnosis.

Table 1 summarizes some state-of-art Covid19 detection models. As highlighted in Table 1, COVID-19 detection models were used advanced computer vision models such as CNNs [49,50]. These models solved many image classification problems. Symptoms of COVID-19 can be seen from medical images. Hence, these models can easily attain high classification rates from the COVID-19 images.

Moreover, these works generally used COVID-19, pneumonia, and control classes. The symptoms of these categories are different. Therefore, advanced image classification models can detect these differences. The categories in our used dataset are COVID-19, control, and others. In the third category (other), there are variable pulmonary disorders, and the symptom of some of them are similar to COVID-19. Therefore, deep learning models (CNNs) did not attain high classification performance. To solve this problem, a patch-based model must be used [51]. Patch-based models can detect small changes since a patch-based can attain high classification performances [52]. In addition, the detected literature gaps are given below.

- As can be seen from Table 1, most of the Covid19 detection models have used CNN models to get classification results since CNNs are the best way to get high classification for images. Nowadays, there are new generation approximations to get higher classification models than CNNs, and one of them is swin architecture. The previously presented models were not used swin architecture to test their results.
- Most studies have used Covid, Control, and Pneumonia classes. They are specific disorders. We have used other classes in this work to classify other pulmonology disorders.
- CNN models have a high (exponential) time burden.

Table 1
State-of-the-art machine learning methods for COVID-19 detection on medical images.

Paper	Method	Dataset information	Results (%)	Key point(s)
Muhammad & Hossain 2021 [12]	Convolutional Neural Network (CNN)	3-class (COVID-19, pneumonia, healthy); lung US images; POCUS dataset [13]	Acc: 91.8 Pre: 92.5 Rec: 93.2	- High-time complexity - Low accuracy - Small data
Wang 2021 [14]	Image preprocessing; data augmentation; CNN	4-class (COVID-19, pneumonia, tuberculosis, healthy); chest CT; own dataset	Ma_F1: 97.04	- High-time complexity - Data augmentation
Keidar 2021 [15]	Image preprocessing; data augmentation; image segmentation; ensemble CNN (ResNet34, ResNet50, ResNet152, VGG16, CheXpert)	2-class (COVID-19, non-COVID-19); CXR images; own dataset	Acc: 90.3 Spe: 90.0 Sen: 90.5	- Data augmentation - Low accuracy - High-time complexity - Low number of classes
Kc 2021 [16]	CNN (DenseNet121)	4-class (COVID-19, bacterial pneumonia, viral pneumonia, healthy); CXR images; combined 2 datasets [17,18]	Acc: 98.69 Ma:_F1 99.0	- High-time complexity
Ravi 2022 [19]	Image preprocessing; deep feature extraction using global average pooling layer of EfficientNet; feature merging; principal component analysis; random forest & SVM classifiers	2-class (COVID-19, non-COVID-19); CXR & chest CT images; 2 datasets [20]	X-ray Acc: 99.48 CT Acc: 99.46	- Low number of classes
Barua 2021 [21]	Deep feature extraction using fully connected layer of Exemplar, iterative neighborhood component analysis feature selection; SVM-classifier	DB1: 4-class (COVID-19, bacterial pneumonia, viral pneumonia, healthy) [22,23]; DB2: 3-class (COVID-19, pneumonia, healthy) [17]; DB3: 3-class (COVID-19, pneumonia, healthy) [24,25]; DB4: 2-class (COVID-19,	DB1 Acc: 97.6 DB2 Acc 89.96 DB3 Acc: 98.84 DB4 Acc: 99.64	- High-time complexity

(continued on next page)

Table 1 (*continued*)

Paper	Method	Dataset information	Results (%)	Key point(s)
		healthy); CXR images		
Saad 2022 [26]	Image segmentation; custom-designed CNN-, ResNet18- & GoogleNet-based deep feature extraction; feature merging; classification	2-class (COVID-19, non-COVID-19); CXR & CT images; extensive COVID-19 dataset [27]	CT Acc 98.9 X-ray Acc: 99.3	- High-time complexity - Low number of classes
Sousa 2022 [28]	Image preprocessing; data augmentation; custom-designed CNN	2-class (COVID-19, non-COVID-19); DB1: combined 2 CXR datasets [17,29]; DB2: CXR & CT images [30]	DB1 Acc: 97.87 DB2 Acc: 98.39	- High-time complexity - Low number of classes - Data augmentation
Aslan et al. [31]	CNN, iterative neighborhood component analysis and iterative ReliefF	3-class (COVID-19, non-COVID-19, Viral Pneumonia) CXR images [30]	Acc: 99.14	- High-time complexity - Small data
Garg et al. [32]	CNN	3- class (COVID-19, Healthy, Others) chest CT image dataset [33]	Acc: 98.83	- High-time complexity - Data augmentation
Luz et al. [34]	EfficientNet family	3- class (COVID-19, Normal, Pneumonia) CXR images [35]	Acc: 93.90	- High-time complexity - Data augmentation
Bhattacharyya [36]	VGG-19, Random forest	3- class (COVID-19, Normal, Pneumonia) CXR images [35]	Acc: 96.60	- High-time complexity - Data augmentation
Agrawal and Choudhary [37]	CNN	3- class (COVID-19, Normal, Pneumonia) CXR images [35]	Acc: 95.20	- High-time complexity - Data augmentation
Haghanifar [38]	CheXNet	2- class (COVID-19, Normal) CXR images [39]	Acc: 96.58	- High-time complexity - Low number of classes - Data augmentation
Ortiz et al. [40]	CNN	3- class (COVID-19, Normal, Pneumonia) CT images [41]	Acc: 92.00 Pre: 69.00 Rec: 59.00 F1: 75.00	- High-time complexity - Low accuracy
Polat [42]	Modified DeepLabV3+	2- class (COVID-19, Normal) CT images [43]	Sen: 90.00 Spe: 97.80	- High-time complexity - Low number of classes - Low sensitivity
	CNN			

Table 1 (*continued*)

Paper	Method	Dataset information	Results (%)	Key point(s)
Wang et al. [44]		2- class (COVID-19, Normal) CT images from the combined dataset	Acc: 99.37 Rec: 99.81	- High-time complexity - Low number of classes - Data augmentation
Padmapriya et al. [45]	CNN	2- class (COVID-19, Normal) CT images [46] and CXR images [17, 47]	Acc: 99.75 Sen: 100.0 Spe: 99.00 Pre: 98.00 F1: 98.00	- High-time complexity - Data augmentation - Low number of classes
Xu et al. [48]	CNN	3- class (COVID-19, Influenza-A, Normal) CT images	Acc: 86.70	- High-time complexity - Low accuracy - Data augmentation

*Acc, accuracy; CNN, convolutional neural network; CXR, chest X-ray; DB, database; Ma_F1, macro averaged F1 score; Pre, precision; Rec, recall; Sen, sensitivity; Spe, specificity; SVM, support vector machine; US, ultrasound.

- There is no need to apply augmentation. Other previously presented works generally used augmentation to get
- Some of the used datasets are small. Therefore, some models used augmentation.
- In Table 1, there is no feature engineering model since they cannot perform like CNNs. However, a highly accurate feature engineering model can be presented.

As mentioned above, the researchers have generally preferred the popular CNNs to get high classification performances on image datasets. Still, CNN's have an exponential time burden since they assign millions of parameters. Therefore, a lightweight model must be proposed to get high classification performance for COVID-19 detection.

Motivation, innovations, and contributions are given below.

1.1. Motivation and our model

In this work, we have selected a relatively extensive and public CT dataset [33] to detect Covid19 and other disorders. The new version of this dataset contains three classes, and CNNs cannot have very high classification results (they resulted in <95% classification accuracy). Our main objective is to get >95% classification performance on this dataset with a lightweight image classification model.

In this wprk, we have proposed a new feature engineering architecture to get a high-classification model with low-classification performance. As stated in the research gap, CNNs models have generally been used to detect COVID-19, but these CNNs have exponential computational complexity. Therefore, we need both highly accurate and lightweight models. The best way to propose a lightweight model feature engineering is the best option. However, feature engineering models cannot get high classification performances like deep learning. New solutions have been presented in the literature, one of which is swin architecture. Our main motivation is to present a new swin-based feature engineering model for image classification. Therefore, we have used two image descriptors in this architecture: textural feature extractors. In the swin architecture, variable-sized windows have been used. Various local features have been extracted by extracting features from these windows, and the classification ability has improved.

Our model has a multileveled feature extraction with ten layers. In each layer, variable-sized patches have been used. By using ten layers,

ten feature vectors have been created. In the feature selection layer, iterative neighborhood component analysis (INCA) has been used to choose the most meaningful features from the generated features have been selected. The classification accuracies of these ten selected features have been calculated classification performances. By merging top k features and applying INCA 11th feature vector has been created and classification performances of these 11 feature vectors have been calculated by deploying the k nearest neighbors (kNN) classifier. To get the best classification performance, iterative hard majority voting (IHMV) has been applied. Therefore, this model is named swin-textural, and the proposed swin-textural generates 20 results and selects the best results automatically since it uses the greedy algorithm. In this respect, our proposed swin-textural is a self-organized image classification model. To test this model, we have used a publicly available Covid19 dataset. Moreover, this dataset has been tested using popular CNN models. Therefore, it is very suitable to show the high classification performance of the presented swin-textural.

1.2. Theoretical background

Symptoms are in the local areas in the CT images. Therefore, a machine learning model should focus on this area to get higher classification results, but this process is a difficult challenge for machine learning models. Patch-based models have been proposed to solve this problem without using any segmentation or pretrained algorithm. In the last of 2020, a new patch-based deep learning model was presented, named ViT (vision transformer) [8]. ViT obtained higher classification performance than popular CNNs [53–56]. Swin transformer [11] is an improved version of the ViT and uses variable-sized patch division operations. By using this strategy, we have proposed a new feature engineering model. Textural feature extractors are good options for extracting distinctive features from the symptomatic areas since CT symptoms generally show textural attributes. Therefore, we have used swin architecture with ten layers, an iterative feature selector, feature combining-based (selecting the best accurate features) child feature generation, and majority voting to get high classification performance. Feature fusion, iterative feature selection, and iterative hard majority voting have been used to get the maximum classification performance.

1.3. Innovations and contributions

The innovations and contributions of the presented swin-textural are given below.

1.3.1. Innovations

- Textural feature extractors are effective but not powerful enough to compete with deep learning models. To increase the classification capability of these models, we have proposed a new feature engineering architecture named swin-textural. In this architecture, ten types of fixed-size patch division have been used.
- A new swin-based Covid19 detection model has been proposed. As stated in the literature, swin architectures (swin transformer) are very effective new-generation image classification models. Herein, we are the first team to use a swin architecture-based feature engineering model for lung disorder detection as far as we know.

1.3.2. Contributions

- We have proposed a new lightweight model (the computational complexity of the presented swin-textural is linear) to get high classification performance from a CT image dataset.
- Our used CT image dataset has results using variable deep-learning models. In this article, we aim to propose a feature engineering model that can be an alternative to deep learning, especially CNNs. The classification results showed swin-textural (98.71% classification accuracy) outperformed, and it is more effective than CNNs for this dataset.

2. Materials and methods

2.1. Dataset

The study CT image dataset was downloaded from Kaggle [33]. It comprised three classes of 2167, 1247, and 757 (total 4171) transverse chest CT images belonging to 80, 80, and 50 (total 210) subjects with COVID-19 (“COVID-19”), other non-COVID lung conditions (“Others”), and normal lung findings (“Healthy”), respectively.

2.2. Swin-textural image classification model

We proposed a self-organized handcrafted CT image classification model, Swin-textural (Fig. 1), that combined computer vision-inspired shifted windows (Swin)-based patch division [11] with two established textural feature extractors, local binary pattern (LBP) and local phase quantization (LPQ). In Swin transformers [11], the input image is resized to 224 × 224, and divided using 14 × 14, 28 × 28, 56 × 56, and 112 × 112 non-overlapping square patches. In our model, we resized the input image to 420 × 420 and designed ten schemes of patch division using 20 × 20, 30 × 30, 35 × 35, 42 × 42, 60 × 60, 70 × 70, 84 × 84, 105 × 105, 140 × 140, and 210 × 210 non-overlapping square patches, which yielded ten feature extraction layers, giving the model added optionality. At each layer, LBP and LPQ operations extracted textural features from individual patches and the undivided input image; all the features were then concatenated and fed to the iterative neighborhood component analysis (INCA) [57] feature selector. INCA selected the most informative set of features in each of the ten extracted feature vectors based on classification accuracies calculated using k-nearest neighbor (kNN) [58]. Among the resultant ten selected feature vectors, features from those that surpassed a preset threshold accuracy of 97.5% were concatenated, and INCA was again applied to create the 11th selected feature vector. All 11 selected feature vectors were input to kNN classifier to generate 11 prediction vectors. From the 11 prediction vectors, iterative hard majority voting (IHMV) [59] was used to generate another nine voted prediction vectors. Finally, the best result among the twenty generated vectors was determined using a greedy algorithm. Details of the individual steps–(1) preprocessing; (2) patch-based textural feature extraction; (3) feature selection; (4) classification; (5) majority voting; and (6) best vector selection–are explained in the following sections.

2.2.1. Preprocessing

First, the gray-level transformation was performed to facilitate the operations of downstream LBP and LPQ feature extraction functions. Next, the gray-leveled images were resized to 420 × 420 to enable the creation of variable-sized patches with pre-specified dimensions (Fig. 1). The steps of this phase are.

Step 0: Read each CT image from the study dataset.
Step 1: Convert the CT image into the gray-level image.
Step 2: Resize the gray-leveled image to 420 × 420

2.2.2. Feature extraction

This constitutes the most novel and important phase of Swin-textural. Patch division of the resized input images was performed as described above (Fig. 1) to create ten feature extraction layers. At each layer, LBP and LPQ operations extracted 59 and 256 features, respectively, from every patch as well as the input image. LBP is a popular textural feature generator that uses 3 × 3 sized overlapping matrices and a basic comparative function, signum, to transform the image to create a map, the histogram of which constitutes the feature vector. LPQ is an LBP-like feature generator that typically uses 5 × 5 sized overlapping

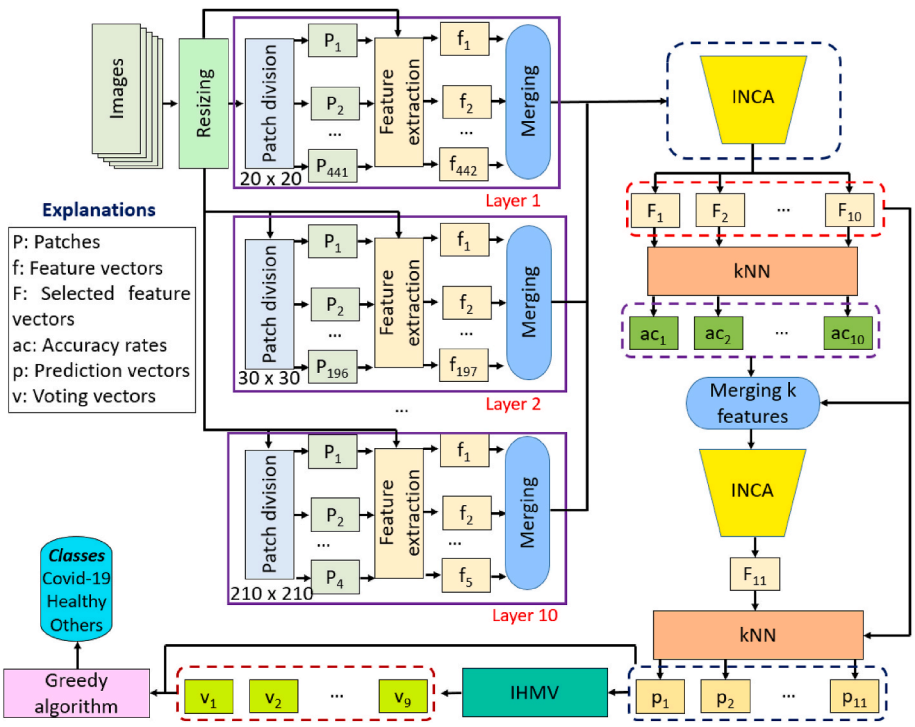

Fig. 1. Block diagram of the proposed Swin-textural model. Ten schemes of patch division produced ten feature extraction layers. For every resized 420 × 420 input image, as the square patch dimensions enlarged incrementally 20 × 20, 30 × 30, 35 × 35, 42 × 42, 60 × 60, 70 × 70, 84 × 84, 105 × 105, 140 × 140, and 210 × 210, the number of patches in successive layers decremented: 21 × 21, 14 × 14, 12 × 12, 10 × 10, 7 × 7, 6 × 6, 5 × 5, 4 × 4, 3 × 3, and 2 × 2 respectively. ac, vector accuracy; F, selected feature vector; f, extracted feature vector; IHMV, iterative hard majority voting; k, number of selected feature vectors with accuracy rates above preset threshold; kNN, k-nearest neighbor; P, patch; p, prediction vector; v, voted vector. Distance-based kNN was used as loss function (accuracy) calculator and classifier at the feature selection and classification phases, respectively.

matrixes to extract features; the size of the matrix is a parameter for LPQ. In LPQ, a short-time Fourier transform is used for blurring and generating imaginary and real components. By coding these components, a bit array with a length of 8 (four most significant bits belong to the imaginary components; and the four least significant bits real components). Using these 8 bits calculated for every overlapping matrix, the transformed image is generated, the histogram of which image is utilized as a feature vector, analogous to LBP. The feature extraction steps are given below.

Step 3: Divide the preprocessed image into patches.

$$p_t^k(a,b) = I(i+a-1, j+b-1), i \in \{1, array^k, \ldots, 420 - array^k + 1\},$$
$$j \in \{1, array^k, \ldots, 420 - array^k + 1\}, a \in \{1, 2, \ldots, array^k\}, b \in \{1, 2, \ldots, array^k\},$$
$$array \in \{20, 30, 35, 42, 60, 70, 84, 105, 140, 210\}, k \in \{1, 2, \ldots, 10\}, t \in \left\{1, 2, \ldots, \frac{420}{array^k}\right\} \quad (1)$$

where p_t^k represents the tth patch of kth type patch division; *array*, length of the used square patch; and I, the resized 420 × 420 input image. Equation (1) defines the multiple patch division function.

Step 4: Extract features from the created patches and input images using LBP and LPQ feature extractors.

$$f_1^k = \psi(\rho(I), \varphi(I)) \quad (2)$$

$$f_{t+1}^k = \psi\left(\rho\left(p_t^k\right), \varphi\left(p_t^k\right)\right) \quad (3)$$

where f_t^k represents the tth feature vector of the kth layer (the length of each feature vector is 315 = 59 + 256); $\psi(.)$, concatenation function; $\rho(.)$, LBP feature extraction function; and $\varphi(.)$, LPQ feature extraction function.

Step 5: Concatenate the generated feature vectors at each level.

$$cf^k = \psi\left(f_1^k, f_2^k, \ldots, f_{t+1}^k\right) \quad (4)$$

where cf^k represents the defined concatenated features of the kth level/layer. The length of the concatenated feature vector at each layer is the product of extracted feature vector length (315) and the number of feature vectors (Table 2).

2.2.3. Feature selection

INCA [57], an improved neighborhood component analysis (NCA),

Table 2
Lengths of concatenated feature vectors at the various feature extraction layers.

Layer	Patch size	Feature vectors, n	Merged feature vector length
1	20 × 20	442	139,320
2	30 × 30	197	62,055
3	35 × 35	145	45,675
4	42 × 42	101	31,815
5	60 × 60	50	15,750
6	70 × 70	37	11,615
7	84 × 84	26	8190
8	105 × 105	17	5355
9	140 × 140	10	3150
10	210 × 210	5	1575

was used to choose the most informative features from the extracted features. In INCA, qualified indexes are first calculated using NCA. Next, the iterative feature selection process is performed using the calculated indexes and a preset loop range, and a loss function calculator is used to calculate the classification accuracy of each selected feature vector. Finally, the feature vector with the maximum classification accuracy is selected. In our model, hyperparameters of the INCA function were set at: loop range, 100 to 1000; classifier, kNN classifier with ten-fold cross-validation. The detailed steps are given below.

Step 6: Apply NCA feature selector to calculate the qualified indexes.

$$id^k = \mathscr{n}\left(cf^k, y\right) \tag{5}$$

where id^k represents the qualified index of the kth combined feature vector, and y, the actual output.

Step 7: Select the feature vectors using the defined loop.

$$fv^k_{r-fl+1}(p,q) = cf^k\left(k, id^k(q)\right), p \in \{1,2,\ldots,D\}, \\ q \in \{1,2,\ldots,r\}, r \in \{fl, fl+1,\ldots,ll\} \tag{6}$$

where fv represents the selected feature vector; fl, the first loop value; ll, the last loop value; and D, number of images.

Step 8: Calculate the accuracies of the selected feature vectors.

$$acc^k_{r-fl+1} = \mathscr{K}\left(fv^k_{r-fl+1}, y\right) \tag{7}$$

where $\mathscr{K}(.)$ represents the kNN classification function.

Step 9: Compute indexes of the best accurate feature vectors.

$$ind^k = argmax\left(acc^k\right) \tag{8}$$

where ind^k represents the index of the kth most accurate vector.

Step 10: Select feature vectors using the calculated indexes in Step 9.

$$F_k = fv^k_{ind^k+fl-1} \tag{9}$$

where F^k represents kth' the selected feature vector.

In addition, we generated an 11th feature vector from the n most accurate feature vectors, where n was defined using a threshold value of classification accuracy pre-specified at 97.5%.

Step 11: Calculate the accuracies of the selected feature vectors.

$$acc_k = \mathscr{K}\left(F^k, y\right) \tag{10}$$

Table 3
Lengths of the selected feature vectors and the calculated accuracies.

Feature vector	Patch size/used features	Length of the created feature vectors by INCA	Accuracy (%)
F1	20 × 20	821	87.75
F2	30 × 30	989	96.48
F3	35 × 35	376	96.88
F4	42 × 42	719	**98.20**
F5	60 × 60	684	**97.65**
F6	70 × 70	900	**98.20**
F7	84 × 84	622	**97.72**
F8	105 × 105	311	97.43
F9	140 × 140	111	97.39
F10	210 × 210	181	96.12
F11	F4+F5+F6+F7	723	**98.47**

Step 12: Merge the extracted feature vectors with classification accuracies over 97.5%.

$$X(p, c+q) = F_k(p,q), if\ acc_k > 97.5\%, c = c + \mathscr{L}(F_k), \\ q \in \{1,2,\ldots,\mathscr{L}(F_k)\}, p \in \{1,2,\ldots,D\} \tag{11}$$

where X represents the merged top most accurate features; $\mathscr{L}(.)$, the length calculation function.

Step 13: Apply the INCA feature selector to calculate the 11th feature vector (see Steps 6–10).

$$F_{11} = \mathscr{I}(X, y) \tag{12}$$

where $\mathscr{I}(.)$ represents the INCA feature selection function.

The attributes of all the selected feature vectors are shown in Table 3. With the study dataset, n was 4, i.e., selected feature vectors from the fourth, fifth, sixth, and seventh layers (highlighted in bold type in Table 3) were used to calculate the 11th selected feature layer. The latter's merged length was 2925 (=719 + 684 + 900 + 622), which was reduced after INCA selection to 723.

2.2.4. Classification

In Swin-textural, kNN, a standard shallow machine learning classifier [58], was used at both feature selection and classification phases. For the latter, the hyperparameters were: k,1; distance, L1-norm; voting, none; validation, and ten-fold cross-validation. The details of the classification phase are given below.

Step 14: Classify 11 selected features by deploying kNN classifier.

$$Pr^u = \mathscr{K}(F^u, y), u \in \{1,2,\ldots,11\} \tag{13}$$

where Pr^u represents the calculated uth predicted vector. 11 prediction vectors were obtained.

2.2.5. Majority voting

IHMV [59], the mode operator and a loop to generate more than one voted vector, was used to obtain voted results. Our model's loop range was set from 3 to 11, generating nine voted vectors. The detailed steps are given below.

Step 15: Sort the generated predicted vectors in Step 14 by descending order of accuracy rates.

$$inx = \mathbb{S}(acc^u) \tag{14}$$

where inx represents the sorted indexes; and $\mathbb{S}(.)$, the sorting function.

Step 16: Apply the mode operator iteratively to create voted vectors.

$$v^{t-2}(i) = \omega\left(Pr^{inx(1)}(i), Pr^{inx(2)}(i), \ldots, Pr^{inx(t)}(i)\right), t \in \{3,4,\ldots,11\}, i \\ \in \{1,2,\ldots,D\} \tag{15}$$

where $\omega(.)$ represents the mode operator, D is the number of observations, and v defines the voted vectors.

2.2.6. Selection of the best result

A greedy algorithm was used to select the best result from among the 11 predictions and 9 voted vectors generated by kNN classifier and IHMV, respectively. By incorporating this final step, the Swin-textural model fully acquired the self-organized attribute. The steps are given below.

Step 16: Calculate the accuracy rates of the generated prediction vectors and voted vectors.

$$acc_u = \delta(Pr^u, y), u \in \{1,2,\ldots,11\} \tag{16}$$

$$acc_{u+h} = \delta\left(v^h, y\right), h \in \{1, 2, \ldots, 9\} \quad (17)$$

where $\delta(.)$ represents the accuracy calculation function, and twenty accuracy values were calculated.

Step 17. Apply the greedy algorithm to choose the most accurate result.

$$id = argmax(acc) \quad (18)$$

$$R = \begin{cases} Pr^{id}, id \leq 11 \\ v^{id-11}, id > 11 \end{cases} \quad (19)$$

where R represents the final predicted value and id, is the index of the most accurate predicted vector.

The given 17 steps have been defined the proposed model.

3. Results

The proposed swin-textural has been tested on the Kaggle CT image dataset, and the results are presented in this section.

3.1. Model construction

The swin-textural is a parametric feature engineering model. To implementation of this model, we have used a simple configured computer, and this model only needs a computer with ≥8 GB main memory, a processing unit with ≥2 GHz, and MATLAB (≥2015) programming environment. The first parameter of the swin-textural is the size of the image. Herein, we resized all images to 420 × 420 to create 10 different patch sizes. Then, 20^2, 30^2, 35^2, 42^2, 60^2, 70^2, 84^2, 105^2, and 210^2 sized patches have been used to create feature extraction layers. In the feature generation phase, two popular feature extraction models have been used: LBP and LPQ. INCA selected the most informative features. Parameters of the INCA are: loop range is from 100 to 1000 and the loss function: kNN with 10-fold CV. The lengths of the selected 10 feature vectors are 821 (F1), 989 (F2), 376 (F3), 719 (F4), 684 (F5), 622 (F6), 900 (F7), 111 (F8), 311 (F9) and 181 (F10) respectively. We want to create a child feature vector (11th feature vector) using these features. The best accurate feature vectors have been selected to create child feature vector (11th feature vector). A threshold point (>97.5% classification accuracy condition) has been used to select the best feature vectors, and the top feature vectors have been merged. These feature vectors are F4, F5, F6 and F7. The length of the 11th feature vector is calculated as 723 (by merging F4, F5, F6 and F7 and applying INCA) and the best accurate vector is the 11th feature vector (it reached 98.47% classification accuracy). In the last phase, IHMV was used. In the IHMV, a mode function and a loop (from 3 to 11) have been used and 9 voted prediction label vectors have been created. Our proposal selected the best resulted voted vector by using a greedy algorithm (it selects the maximum accurate vector as a result).

3.2. Performance metrics

To evaluate the proposed swin-textural, we have used the commonly used performance evaluation metrics. These metrics are recall – it is shown class-wise classification accuracy –precision – which is a very important performance to demonstrate detection rate –. F1-score – this performance metric is the harmonic average of the recall and precision – and classification accuracy. The mathematical expressions of these performance metrics are given below.

$$recall = \frac{tp}{fn + tp} \quad (20)$$

$$precision = \frac{tp}{fp + tp} \quad (21)$$

$$F1 = \frac{2tp}{fp + fn + 2tp} \quad (22)$$

$$accuracy = \frac{tn + tp}{fn + fp + tn + tp} \quad (23)$$

Herein, the critical parameters are fn, fp, tn, tp and these parameters represent false negatives, false positives, true negatives, and true positives, respectively.

3.3. Classification results

This section gives the classification performance of the presented swin-textural are. Our proposed swin-textural generates 20 classification results (10 feature-based results+ 1 child feature vector result + 9 voted results) and selects the best accurate result as the final result. The 18th and 20th results, the best-voted results (highlighted in bold in Table 4), were obtained by majority voting of the top 9 and all 11 prediction vectors, respectively. The former (18th result), which required fewer predictor vectors, was selected as the final result. The confusion matrix of the 18th result is also shown in Fig. 2.

By using the given confusion matrix, the comprehensive results have been calculated.

3.4. Time complexity

Processing time for grayscale conversion and image resizing in the preprocessing phase is dependent on size of the original image. The associated time complexity was $O(2n) \cong O(n)$, where n represents the size of the image. Processing time for patch division and textural feature extraction by the two image descriptors in the feature generation phase was $O(2pnlogn) \cong O(pnlogn)$, where p represents the number of patches, and the time complexities of used image descriptors are equal to $O(n)$. The time complexity of the INCA function in the feature selection phase was $O(N + IC)$, where N represents the complexity coefficient of the NCA; I, the number of iterations; and C, the time burden of the kNN classifier. For generating the 11th feature vector using the calculated accuracies of the ten selected feature vectors, the time burden was $O(10C + M) \cong O(C + M)$, where M represents the length of the features. In the kNN classification phase, the time burden was $O(K)$. For the

Table 4
Performance of the predicted and voted vectors in the Swin-textural model.

No	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)
1	87.75	86.81	84.90	85.77
2	96.48	95.95	95.68	95.81
3	96.88	96.16	96.28	96.21
4	98.20	97.78	97.87	97.82
5	97.65	97.23	97.22	97.22
6	97.72	97.35	97.39	97.36
7	98.20	97.70	97.88	97.79
8	97.39	96.62	97.26	96.91
9	97.43	96.93	97.09	97.00
10	96.12	94.89	95.60	95.22
11	98.47	98.26	98.18	98.22
12	98.59	98.31	98.30	98.30
13	98.44	98.21	98.08	98.13
14	98.63	98.44	98.36	98.40
15	98.44	98.30	98.08	98.18
16	98.61	98.43	98.37	98.40
17	98.59	98.41	98.28	98.34
18	**98.71**	**98.51**	98.42	**98.46**
19	98.56	98.35	98.26	98.29
20	**98.71**	98.46	**98.45**	98.44

The ten feature extraction layers generated the first ten results; the 11th result, by applying iterative neighborhood component analysis function to the four most discriminative feature vectors; and the 12th to 20th results were generated by iterative hard majority voting.

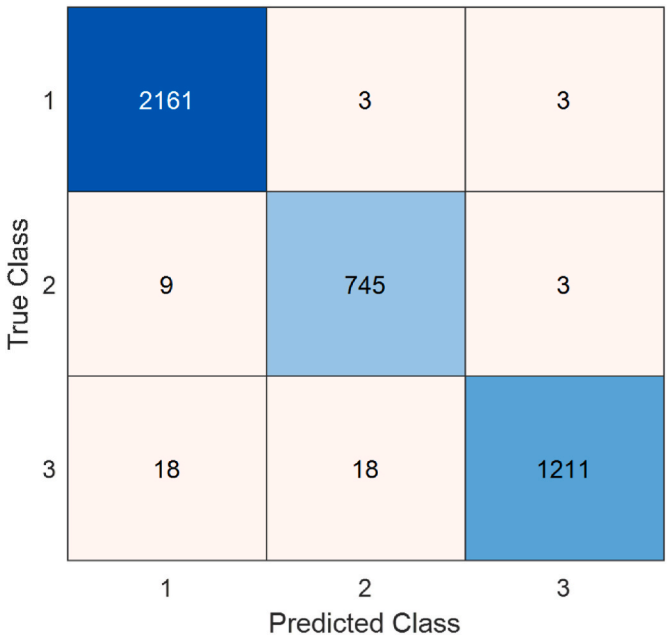

Fig. 2. Confusion matrix of the best calculated result. 1, 2, and 3 represent "COVID-19", "Healthy", and "Others" classes, respectively.

Table 5
Computational complexities of the Swin-textural model.

Phase	Time burden
Preprocessing	$O(n)$
Feature extraction	$O(pnlogn)$
Feature selection	$O(N + IC + C + M)$
Classification	$O(K)$
Majority voting	$O(iM)$
Selection the best result	$O(M)$
Total	$O(n + pnlogn + N + IC + C + M + K + iM + M)$

majority voting of voted vectors, the time complexity was $O(iM)$, where i represents the number of iterations of the IHMV. For the final selection of the best result, the time complexity was $O(M)$. The swin-textural method has a linear time burden as demonstrated by the total and component-time complexities shown in Table 5.

4. Discussion

Our novel self-organizing handcrafted swin-textural CT image classification model is computationally lightweight and attained high classification performance for detecting COVID-19 on a public three-class chest CT image dataset [33] that had been used in training several deep learning models [32,60]. In our model, we modified the swin transformer architecture, increasing the number of patch division schemes from four to ten, which produced ten feature extraction layers. In each layer, LBP and LPQ were used to extract textural features from the individual patches as well as the input image, which were all concatenated to form one merged feature vector, i.e., ten feature vectors per input image. INCA was used to select the most discriminative features from the merged features vectors. An 11th feature vector was created by applying INCA to the threshold-specified top selected feature vectors. All 11 selected feature vectors were fed to kNN classifier to generate 11 predicted vectors. Nine additional voted vectors were created using the IHMV algorithm, and the best result among the final 20 results determined using a greedy algorithm.

From Table 3, the fourth and sixth patch division schemes, with patch dimensions of 42 × 42 and 70 × 70, respectively, yielded equal

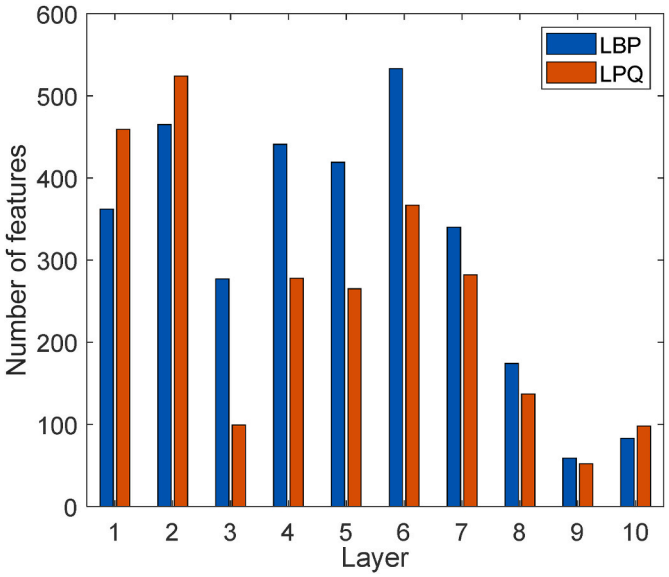

Fig. 3. Distribution of local binary pattern (LBP) and local phase quantization (LPQ) features per feature extraction layer. Except for layers 1, 2 and 10, LBP is the more effective feature extractor than LPQ.

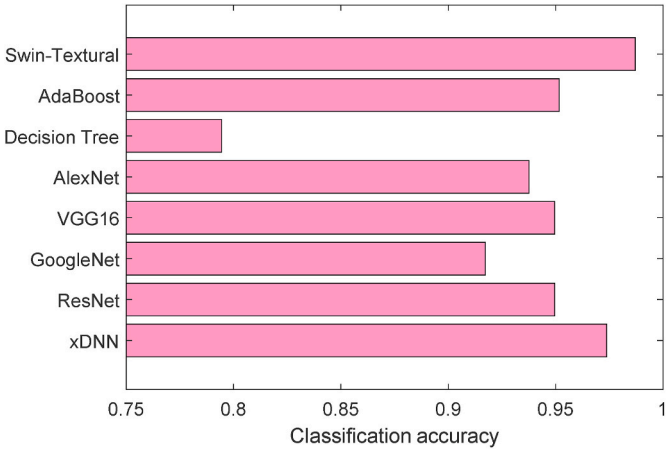

Fig. 4. Comparison of classification accuracies of deep learning models in Soares [32] and the Swin-textural model. The former and later reported two- and three-class accuracies for detection of COVID-19 on CT images.

highest 98.20% classification accuracy rates on our study dataset. Even though LBP and LPQ extracted 59 and 256 textural features per patch or undivided input image, respectively, by design, LBP-based features predominated among the features that were selected by INCA (see Fig. 3). Despite the disproportionate representation of LBP among selected features, it is important to note that the inclusion of both image descriptors contributed to the overall accuracy of the model, which outperformed both LPB- and LPQ-based ablation models as shown in Fig. 4 (Table 6).

4.1. Comparisons

Garg et al. [32] tested 20 deep learning models on the same dataset as ours. Swin-textural outperformed these deep learning models, achieving 2.13%, 5.19%, and 7.48% higher class-wise classification accuracies than the next best model, EfficientNetB5, for "COVID-19", "Healthy", and "Others" classes, respectively. The comparative results are listed in Table 6.

Additionally, there are two classes in the former version of this

Table 6
Comparison of classification accuracies of 20 deep learning models in Garg et al. [32] and the Swin-textural model.

No	Model	Accuracy (%)		
		COVID-19	Healthy	Others
1	DenseNet121	96.99	91.93	89.66
2	DenseNet169	97.19	91.95	89.71
3	DenseNet201	97.23	91.97	89.89
4	EfficientNetB0	96.33	92.01	89.80
5	EfficientNetB1	92.76	89.65	86.23
6	EfficientNetB2	95.30	91.97	89.31
7	EfficientNetB3	95.80	91.94	89.52
8	EfficientNetB4	96.34	92.20	90.09
9	EfficientNetB5	97.59	93.22	90.77
10	EfficientNetB6	95.97	91.77	89.63
11	EfficientNetB7	94.32	91.61	88.56
12	InceptionResNetv2	94.27	91.54	87.90
13	InceptionV3	95.49	91.30	88.24
14	ResNet101V2	93.64	90.35	87.17
15	ResNet152V2	93.80	90.93	87.66
16	ResNet50	96.25	91.74	89.33
17	ResNet50V2	93.08	90.81	86.68
18	VGG16	88.89	89.86	82.40
19	VGG19	85.58	87.49	78.40
20	Xception	94.70	91.79	88.54
21	**Swin-textural**	**99.72**	**98.41**	**97.11**

As highlighted in Table 6, the best model for this dataset is swin-textural. This model attained superior classification performances than other popular 20 deep learning models for three classes classification.

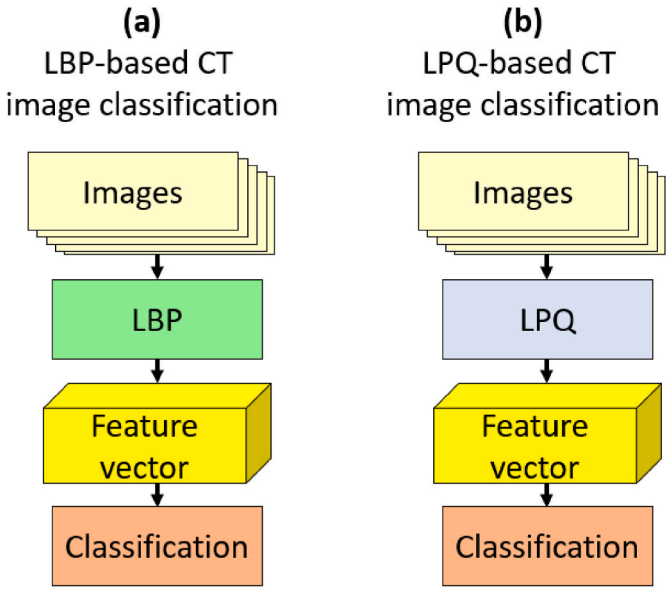

Fig. 5. Block diagrams of the local binary pattern (LBP) and local phase quantization (LPQ)-based image classification models for ablations.

dataset and these classes are COVID-19 and healthy. Other classes were added later. In his thesis investigating deep learning models on the same dataset, Soares [60] reported two-class "COVID-19" versus "Healthy" results. The best deep learning model, xDNN, attained 97.38% classification accuracy, which was still lower than the 98.71% overall accuracy achieved by Swin-textural for three-class classification on the dataset.

Finally, the swin-textural model performed favorably against state-of-the-art models for three- [32] and two-class [60] classification of COVID-19 on CT images.

4.2. Ablations

We performed ablation studies comparing the Swin-textural model with two classification models that combined our modified Swin architecture with either LBP and LPQ feature extraction functions separately (Fig. 5). The Swin-textural model outperformed both LBP-based and LPQ-based models by over 7% classification accuracy on the same dataset (Table 7).

Table 7
Overall classification performances of the Swin-textural model versus local binary pattern (LBP) and local phase quantization (LPQ)-based classification models.

Model	Accuracy (%)	Precision (%)	Recall (%)	F1 score (%)
LBP	91.56	88.84	89.74	89.27
LPQ	91.39	88.16	88.97	88.55
Swin-textural	98.71	98.51	98.42	98.46

As stated in Table 7, the classification performance of the LBP and LPQ is very close. We gave the number of features per the used layers in Fig. 3. According to the results, the best-resulted layers are 4th and 6th layers since these layers yielded over 98% classification accuracy. The common attribute of these layers is to use more LBP features than LPQ features.

In this work, we have used the INCA feature selector. INCA selects the best feature combination automatically. To choose the best feature selection, we have tested NCA (INCA uses NCA), mRMR (minimum redundancy maximum relevance), Chi2, and ReliefF feature selectors. The calculated classification accuracies for the 4th layer (this layer reached 98.20% classification accuracy by using INCA) have been given in Fig. 6.

Fig. 6 demonstrates that the best accurate feature selector is NCA. Therefore, we have used INCA as a feature selector in this model. Moreover, k values of the kNN have been ablated in this section. We have selected k values from 1 to 10 for ablation, and this test was applied on the 4th layer. The calculated classification accuracies per the used k values are depicted in Fig. 7.

Fig. 7 demonstrates that the best k value for the kNN is 1. Therefore, we have used 1NN as classifier.

4.3. Highlights

The proposed swin-textural model has the following advantages. First, it possesses linear time complexity and does not require heavy computing power or expensive hardware/software to run, which enhances the ease of implementation. For example, it can be used in

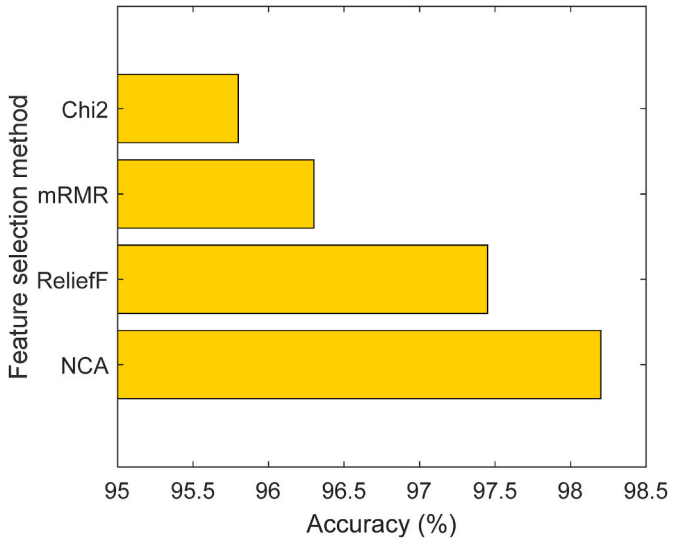

Fig. 6. Classification accuracies per the used basic feature selector for the 4th layer.

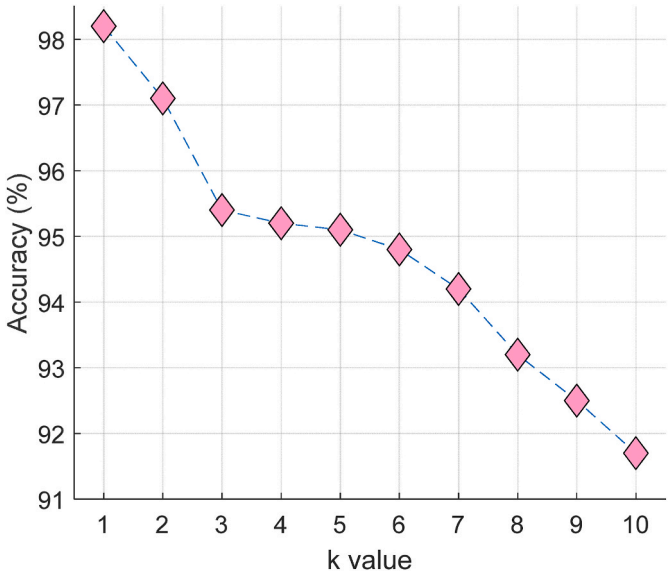

Fig. 7. Classification performances obtained for various k values.

hospitals for automated screening of CT images for COVID-19. Second, the model outperforms published deep learning models, achieving 98.71% overall classification accuracy despite using shallow hand-crafted methods with a shorter training time cost. It is possible that further improvement can be obtained by optimizing and fine-tuning the operation processes. Third, swin-textural is a parametric model, the parameters in the model being the number of layers, feature extractors, feature selection function, classifier, and majority voting technique. By modifying these parameters, new generation models can be developed.

Our limitations are given as follows. The recommended swin-textural attained higher classification accuracy for this dataset, but it can be tested on bigger datasets in the near future. Moreover, there is no fine-tuning operation not to increase computational complexity. On the other hand, by using fine-tuning, higher classification performances can be calculated.

As future work, we plan to incorporate transfer learning in the architecture for efficient deep feature engineering and explainable artificial intelligence to understand the results using our model [61–63]. In addition, such a deep feature engineering model may be re-purposed to solve other image classification problems.

5. Conclusion

A novel hand-modeled image classification model has been proposed in this research, and our proposal is named swin-textural. The recommended swin-textural was tested on a public CT image dataset. Using CT images and a machine learning technique for COVID-19 detection is one of the processes that will speed up COVID-19 detection. High-performance machine learning methods are seen as one of the most important solutions to solve the bottlenecks caused by PCR testing. Thus, many researchers have used deep learning models to propose an effective COVID-19 detection model. We presented a hand-modeled image classification architecture and we demonstrated the superiority of our presented swin-textural. The important points about our proposal are.

- The most suitable sizes of patches are 42 × 42 and 70 × 70 to solve the COVID-19 detection problem for this dataset. Using these patches, a classification accuracy of 98.2% was obtained.
- The worst sizes of patches are 20 × 20 and 210 × 210 since the worst two results were attained using these patches.
- The classification result of the 11th feature is the best (it reached 98.47% overall accuracy), and the 11th feature vector is created using the 4th, 5th, 6th, and 7th feature vectors.
- Swin-textural reached 98.71% overall classification accuracy.
- The best result is a voted result. To get the best result, there is no need to used the 1st and 10th feature vectors (there is no need to use 20 × 20 and 210 × 210 sized patch division).
- LBP is a more effective feature extraction function than LPQ to get high classification performance for this problem.

Swin-textural is ready to use in a medical center for COVID-19 detection since it has attained high classification performance. Soon, we will develop an automated COVID-19 detection assistant and COVID-19 detection duration will be decreased (using PCR tests, COVID-19 detection durations are from 6 h to 4 days), and this duration will be decreased from hours/days to minutes. Moreover, swin-textural is a parametric model, the parameters of this model are a number of layers, feature extractors, feature selection function, classifier and the used majority voting technique. By changing these parameters, new generation models can be proposed. In the future, we are planning to use transfer learning in this architecture and a deep feature engineering model will be proposed. By using this deep feature engineering model, other image classification problems will be solved.

Funding

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgement

None.

References

[1] Sanyaolu A, Okorie C, Marinkovic A, Haider N, Abbasi AF, Jaferi U, et al. The emerging SARS-CoV-2 variants of concern. Therapeutic advances in infectious disease 2021;8:20499361211024372.
[2] Sharma A, Tiwari S, Deb MK, Marty JL. Severe acute respiratory syndrome coronavirus-2 (SARS-CoV-2): a global pandemic and treatment strategies. Int J Antimicrob Agents 2020;56:106054.
[3] Dong D, Tang Z, Wang S, Hui H, Gong L, Lu Y, et al. The role of imaging in the detection and management of COVID-19: a review. IEEE reviews in biomedical engineering 2020;14:16–29.
[4] Hani C, Trieu NH, Saab I, Dangeard S, Bennani S, Chassagnon G, et al. COVID-19 pneumonia: a review of typical CT findings and differential diagnosis. Diagnostic and interventional imaging 2020;101:263–8.
[5] Li M, Lei P, Zeng B, Li Z, Yu P, Fan B, et al. Coronavirus disease (COVID-19): spectrum of CT findings and temporal progression of the disease. Acad Radiol 2020;27:603–8.
[6] Shiri I, Mostafaei S, Haddadi Avval A, Salimi Y, Sanaat A, Akhavanallaf A, et al. High-dimensional multinomial multiclass severity scoring of COVID-19 pneumonia using CT radiomics features and machine learning algorithms. Sci Rep 2022;12:1–12.
[7] Prokop M, Van Everdingen W, van Rees Vellinga T, Quarles van Ufford H, Stöger L, Beenen L, et al. CO-RADS: a categorical CT assessment scheme for patients suspected of having COVID-19—definition and evaluation. Radiology 2020;296:E97–104.
[8] Dosovitskiy A, Beyer L, Kolesnikov A, Weissenborn D, Zhai X, Unterthiner T, et al. An image is worth 16x16 words: transformers for image recognition at scale. 2020. arXiv preprint arXiv:201011929.
[9] Tolstikhin IO, Houlsby N, Kolesnikov A, Beyer L, Zhai X, Unterthiner T, et al. Mlp-mixer: an all-mlp architecture for vision. Adv Neural Inf Process Syst 2021;34.
[10] Trockman A, Kolter JZ. Patches are all you need?. 2022. arXiv preprint arXiv:220109792.

[11] Liu Z, Lin Y, Cao Y, Hu H, Wei Y, Zhang Z, et al. Swin transformer: hierarchical vision transformer using shifted windows. p. 10012-10022.
[12] Muhammad G, Hossain MS. COVID-19 and non-COVID-19 classification using multi-layers fusion from lung ultrasound images. Inf Fusion 2021;72:80–8.
[13] Born J, Wiedemann N, Cossio M, Buhre C, Brändle G, Leidermann K, et al. Accelerating detection of lung pathologies with explainable ultrasound image analysis. Appl Sci 2021;11:672.
[14] Wang S-H, Nayak DR, Guttery DS, Zhang X, Zhang Y-D. COVID-19 classification by CCSHNet with deep fusion using transfer learning and discriminant correlation analysis. Inf Fusion 2021;68:131–48.
[15] Keidar D, Yaron D, Goldstein E, Shachar Y, Blass A, Charbinsky L, et al. COVID-19 classification of X-ray images using deep neural networks. Eur Radiol 2021;31:9654–63.
[16] Kc K, Yin Z, Wu M, Wu Z. Evaluation of deep learning-based approaches for COVID-19 classification based on chest X-ray images. Signal, image and video processing 2021;15:959–66.
[17] Cohen JP, Morrison P, Dao L, Roth K, Duong TQ, Ghassemi M. Covid-19 image data collection: prospective predictions are the future. 2020. arXiv preprint arXiv:200611988.
[18] Kermany D, Zhang K, Goldbaum M. Labeled optical coherence tomography (OCT) and chest X-ray images for classification. Mendeley Data; 2018.
[19] Ravi V, Narasimhan H, Chakraborty C, Pham TD. Deep learning-based meta-classifier approach for COVID-19 classification using CT scan and chest X-ray images. Multimed Syst 2022;28:1401–15.
[20] El-Shafai W, Abd El-Samie FE. Extensive COVID-19 X-ray and CT chest images dataset. Mendeley Data; 2020.
[21] Barua PD, Muhammad Gowdh NF, Rahmat K, Ramli N, Ng WL, Chan WY, et al. Automatic COVID-19 detection using exemplar hybrid deep features with X-ray images. Int J Environ Res Publ Health 2021;18:8052.
[22] Ozturk T, Talo M, Yildirim EA, Baloglu UB, Yildirim O, Acharya UR. Automated detection of COVID-19 cases using deep neural networks with X-ray images. Comput Biol Med 2020;121:103792.
[23] Kermany DS, Goldbaum M, Cai W, Valentim CCS, Liang H, Baxter SL, et al. Identifying medical diagnoses and treatable diseases by image-based deep learning. Cell 2018;172:1122–31.
[24] Chowdhury MEH, Rahman T, Khandakar A, Mazhar R, Kadir MA, Mahbub ZB, et al. Can AI help in screening viral and COVID-19 pneumonia? IEEE Access 2020;8:132665–76.
[25] Rahman T, Khandakar A, Qiblawey Y, Tahir A, Kiranyaz S, Kashem SBA, et al. Exploring the effect of image enhancement techniques on COVID-19 detection using chest X-ray images. Comput Biol Med 2021;132:104319.
[26] Saad W, Shalaby WA, Shokair M, El-Samie FA, Dessouky M, Abdellatef E. COVID-19 classification using deep feature concatenation technique. J Ambient Intell Hum Comput 2022;13:2025–43.
[27] El-Shafai W, Abd El-Samie FE. Extensive and augmented COVID-19 X-ray and CT chest images dataset. Mendeley Data; 2020.
[28] De Sousa PM, Carneiro PC, Oliveira MM, Pereira GM, da Costa Junior CA, de Moura LV, et al. COVID-19 classification in X-ray chest images using a new convolutional neural network: CNN-COVID. Research on Biomedical Engineering 2022;38:87–97.
[29] Wang X, Peng Y, Lu L, Lu Z, Bagheri M, Summers RM. Chestx-ray8: hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. p. 2097-2106.
[30] Vayá MDLI, Saborit JM, Montell JA, Pertusa A, Bustos A, Cazorla M, et al. Bimcv covid-19+: a large annotated dataset of rx and ct images from covid-19 patients. 2020. arXiv preprint arXiv:200601174.
[31] Aslan N, Koca GO, Kobat MA, Dogan S. Multi-classification deep CNN model for diagnosing COVID-19 using iterative neighborhood component analysis and iterative ReliefF feature selection techniques with X-ray images. Chemometr Intell Lab Syst 2022;224:104539.
[32] Garg A, Salehi S, La Rocca M, Garner R, Duncan D. Efficient and visualizable convolutional neural networks for COVID-19 classification using Chest CT. Expert Syst Appl 2022;195:116540.
[33] Soares E, Angelov P, Biaso S, Froes MH, Abe DK. A COVID multiclass dataset of CT scans. https://www.kaggle.com/datasets/plameneduardo/a-covid-multiclass-dataset-of-ct-scans; 2020.
[34] Luz E, Silva P, Silva R, Silva L, Guimarães J, Miozzo G, et al. Towards an effective and efficient deep learning model for COVID-19 patterns detection in X-ray images. Research on Biomedical Engineering 2022;38:149–62.
[35] Wang L, Lin ZQ, Wong A. Covid-net: a tailored deep convolutional neural network design for detection of covid-19 cases from chest x-ray images. Sci Rep 2020;10:1–12.
[36] Bhattacharyya A, Bhaik D, Kumar S, Thakur P, Sharma R, Pachori RB. A deep learning based approach for automatic detection of COVID-19 cases using chest X-ray images. Biomed Signal Process Control 2022;71:103182.
[37] Agrawal T, Choudhary P. FocusCovid: automated COVID-19 detection using deep learning with chest X-ray images. Evolving Systems 2022;13:519–33.
[38] Haghanifar A, Majdabadi MM, Choi Y, Deivalakshmi S, Ko S. Covid-cxnet: detecting covid-19 in frontal chest x-ray images using deep learning. Multimed Tool Appl 2022:1–31.
[39] Zompatori M, Ciccarese F, Fasano L. Overview of current lung imaging in acute respiratory distress syndrome. Eur Respir Rev 2014;23:519–30.
[40] Ortiz A, Trivedi A, Desbiens J, Blazes M, Robinson C, Gupta S, et al. Effective deep learning approaches for predicting COVID-19 outcomes from chest computed tomography volumes. Sci Rep 2022;12:1–10.
[41] Zhang K, Liu X, Shen J, Li Z, Sang Y, Wu X, et al. Clinically applicable AI system for accurate diagnosis, quantitative measurements, and prognosis of COVID-19 pneumonia using computed tomography. Cell 2020;181:1423–33. e11.
[42] Polat H. A modified DeepLabV3+ based semantic segmentation of chest computed tomography images for COVID-19 lung infections. Int J Imag Syst Technol 2022;32:1481–95.
[43] COVID-19. Medical segmentation. http://medicalsegmentation.com/covid19/. [Accessed 10 October 2021].
[44] Wang Z, Dong J, Zhang J. Multi-model ensemble deep learning method to diagnose COVID-19 using chest computed tomography images. J Shanghai Jiaot Univ 2022;27:70–80.
[45] Padmapriya T, Kalaiselvi T, Priyadharshini V. Multimodal covid network: multimodal bespoke convolutional neural network architectures for COVID-19 detection from chest X-ray's and computerized tomography scans. Int J Imag Syst Technol 2022;32:704–16.
[46] Zhao J, Zhang Y, He X, Xie P. Covid-ct-dataset: a ct scan dataset about covid-19. 2020.
[47] Wang X, Peng Y, Lu L, Lu Z, Bagheri M, Summers RM. Chestx-ray8: hospital-scale chest x-ray database and benchmarks on weakly-supervised classification and localization of common thorax diseases. Proceedings of the IEEE conference on computer vision and pattern recognition 2017:2097–106.
[48] Xu X, Jiang X, Ma C, Du P, Li X, Lv S, et al. A deep learning system to screen novel coronavirus disease 2019 pneumonia. Engineering 2020;6:1122–9.
[49] Bouguettaya A, Zarzour H, Taberkit AM, Kechida A. A review on early wildfire detection from unmanned aerial vehicles using deep learning-based computer vision algorithms. Signal Process 2022;190:108309.
[50] Shafiq M, Gu Z. Deep residual learning for image recognition: a survey. Appl Sci 2022;12:8972.
[51] Abdou MA. Literature review: efficient deep neural networks techniques for medical image analysis. Neural Comput Appl 2022:1–22.
[52] Hertel R, Benlamri R. Deep learning techniques for covid-19 diagnosis and prognosis based on radiological imaging. ACM Computing Surveys; 2022.
[53] Baygin M, Yaman O, Barua PD, Dogan S, Tuncer T, Acharya UR. Exemplar Darknet19 feature generation technique for automated kidney stone detection with coronal CT images. Artif Intell Med 2022;127:102274.
[54] Dogan S, Barua PD, Kutlu H, Baygin M, Fujita H, Tuncer T, et al. Automated accurate fire detection system using ensemble pretrained residual network. Expert Syst Appl 2022;203:117407.
[55] Kobat MA, Kivrak T, Barua PD, Tuncer T, Dogan S, Tan R-S, et al. Automated COVID-19 and heart failure detection using DNA pattern technique with cough sounds. Diagnostics 2021;11:1962.
[56] Barua PD, Chan WY, Dogan S, Baygin M, Tuncer T, Ciaccio EJ, et al. Multilevel deep feature generation framework for automated detection of retinal abnormalities using OCT images. Entropy 2021;23:1651.
[57] Tuncer T, Dogan S, Özyurt F, Belhaouari SB, Bensmail H. Novel multi center and threshold ternary pattern based method for disease detection method using voice. IEEE Access 2020;8:84532–40.
[58] Peterson LE. K-nearest neighbor. Scholarpedia 2009;4:1883.
[59] Dogan A, Akay M, Barua PD, Baygin M, Dogan S, Tuncer T, et al. PrimePatNet87: prime pattern and tunable q-factor wavelet transform techniques for automated accurate EEG emotion recognition. Comput Biol Med 2021;138:104867.
[60] Almeida Soares E. Explainable-by-Design deep learning. 2022.
[61] Loh HW, Ooi CP, Seoni S, Barua PD, Molinari F, Acharya UR. Application of explainable artificial intelligence for healthcare: a systematic review of the last decade (2011–2022). Comput Methods Progr Biomed 2022:107161.
[62] Jahmunah V, Ng E, Tan R-S, Oh SL, Acharya UR. Uncertainty quantification in DenseNet model using myocardial infarction ECG signals. Comput Methods Progr Biomed 2022:107308.
[63] Jahmunah V, Ng E, Tan R-S, Oh SL, Acharya UR. Explainable detection of myocardial infarction using deep learning models with Grad-CAM technique on ECG signals. Comput Biol Med 2022;146:105550.