Computer-aided detection of COVID-19 from CT scans using an ensemble of CNNs and KSVM classifier

Bejoy Abraham • Madhu S. Nair

Received: 20 November 2020 / Revised: 13 May 2021 / Accepted: 12 July 2021 / Published online: 16 August 2021
© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2021

Abstract
Corona Virus Disease-2019 (COVID-19) is a global pandemic which is spreading briskly across the globe. The gold standard for the diagnosis of COVID-19 is viral nucleic acid detection with real-time polymerase chain reaction (RT-PCR). However, the sensitivity of RT-PCR in the diagnosis of early-stage COVID-19 is less. Recent research works have shown that computed tomography (CT) scans of the chest are effective for the early diagnosis of COVID-19. Convolutional neural networks (CNNs) are proven successful for diagnosing various lung diseases from CT scans. CNNs are composed of multiple layers which represent a hierarchy of features at each level. CNNs require a big number of labeled instances for training from scratch. In medical imaging tasks like the detection of COVID-19 where there is a difficulty in acquiring a large number of labeled CT scans, pre-trained CNNs trained on a huge number of natural images can be employed for extracting features. Feature representation of each CNN varies and an ensemble of features generated from various pre-trained CNNs can increase the diagnosis capability significantly. In this paper, features extracted from an ensemble of 5 different CNNs (MobilenetV2, Shufflenet, Xception, Darknet53 and EfficientnetB0) in combination with kernel support vector machine is used for the diagnosis of COVID-19 from CT scans. The method was tested using a public dataset and it attained an area under the receiver operating characteristic curve of 0.963, accuracy of 0.916, kappa score of 0.8305, F-score of 0.91, sensitivity of 0.917 and positive predictive value of 0.904 in the prediction of COVID-19.

Keywords COVID-19 • Computed tomography • CNN • KSVM • Computer-aided diagnosis

1 Introduction
On March 2019, the World Health Organization (WHO) has declared the Corona Virus Disease-2019 (COVID-19) a pandemic [17]. Early diagnosis and rapid isolation of the patients are crucial for effective treatment and for preventing the spread of the disease. Viral nucleic acid detection with real-time polymerase chain reaction (RT-PCR) is the gold standard for the detection of COVID-19. The availability of RT-PCR test kits is insufficient in most of the places where the disease has spread. Moreover, RT-PCR shows less sensitivity in the early diagnosis of COVID-19 [10]. Due to the shortage of RT-PCR test kits and the likelihood of false-negative RT-PCR findings during the early diagnosis, National Health Commission of China has endorsed the diagnosis of COVID-19 using chest CT findings [4]. COVID-19 in CT scans is characterized by ground glass opacity (GGO). GGO represents an area of hazy increased lung opacity through which underlying vessels are visible [14]. Manual analysis of a large number of CT scans by the radiologists is tedious. A computer-aided diagnosis mechanism could assist radiologists in the diagnosis of COVID-19 from CT scans.

There exist a number of computer-aided studies for the diagnosis of COVID-19 from CT scans. He et al. [13] proposed a method that combines transfer learning with self-supervised learning. Anwar and Zakir [3] and Sakagianni et al. [21] also used the public dataset created by [13] for their study. Anwar and Zakir [3] used EfficientNet to diagnose COVID-19 using CT scans. The model used three different learning approaches, namely, reduce on plateau, cyclic
and constant learning rates. Sakagianni et al. [21] used auto Machine Learning Cloud Vision platform for the prediction of COVID-19. Abraham and Nair [2] proposed a multi-CNN architecture consisting of multiple pre-trained CNNs along with correlation-based feature selection (CFS) technique and Bayesnet classifier for the classification of COVID-19 and non-COVID-19 X-ray images. Pu et al. [19] developed three classifiers based on CNN for distinguishing COVID-19 and pneumonia. Mei et al. [15] used a combination of CNN, multi-layer perceptron (MLP) and clinical information to diagnose COVID-19 from CT scans. Xu et al. [28] used a local attention classification model for classifying the infected regions into COVID-19, influenza and irrelevant to infections. Wu et al. [27] used a deep learning- based multi-view fusion model for the detection of COVID-19. In addition to CT, few works have explored the use of X-rays [12] and cough audio recordings [9] for the computer-aided diagnosis of COVID-19.

All the above-mentioned methods which employed CT scans for the diagnosis of COVID-19 have used a single CNN for the diagnosis of COVID-19. Combination of features from multiple CNNs could produce better results than single CNNs. None of the state-of-the-art studies have explored the optimum combination of features from pre-trained CNNs along with an off-the-shelf classifier for the detection of COVID-19.

The contributions of the proposed ensemble method are listed below.

i. The proposed method explored the combination of various pre-trained CNNs to put forward a model that gives better performance than single pre-trained CNNs and transfer learning using end-to-end CNNs for the diagnosis of COVID-19 from CT scans. Based on empirical analysis, an ensemble of features generated using 5 pre-trained CNNs (MobilenetV2, Shufflenet, Xception, Darknet53 and EfficientnetB0) is proposed in this study for the detection of COVID-19 from CT scans. Most of the state-of-the-art methods employing CT images for the detection of COVID-19 have used features from a sole CNN.

ii. The proposed technique used an off-the-shelf classifier, kernel support vector machine (KSVM), for the classification of COVID-19 from CT scans. No existing techniques have employed the combination of the above-mentioned ensemble of CNNs with KSVM for the detection of COVID-19.

2 Materials and methods

Figure 1 displays the architecture of the proposed method. The functions of various blocks of the architecture are described in the following subsections.

2.1 Feature extraction using multiple CNNs

Convolutional neural networks produce powerful feature representations for computer vision tasks. Many research works like segmentation of liver tumor [30] and diagnosis of prostate cancer [1] have proven the effectiveness of CNN in medical applications. For most of the medical diagnostic tasks, training a network from scratch is difficult due to the non-availability of sufficiently large medical datasets. Off-the-shelf features extracted using pre-trained CNNs were found successful in medical diagnosis. Using a pre-trained network is much faster than training a network from scratch. Off-the-shelf features extracted using CNN could achieve superior results even without fine-tuning [23].

Combining features extracted from an ensemble of pre-trained networks could achieve better performance than features extracted using single CNNs. The proposed method combines the features generated from the last fully connected (FC) layer of an ensemble of various CNNs. The CNNs were pre-trained on Imagenet dataset [7] which contains millions of images. The pre-trained CNNs can classify objects into 1000 different categories. The number of layers, input size and number of parameters of the above-mentioned pre-trained networks are represented in Table 1.

As the dataset contains 746 images, a feature vector of dimension 746 × 1000 is generated by each CNN (Activations from the last convolution layer of Darknet53 and last FC layer of other CNNs). Feature vectors produced by each CNN are concatenated to get a feature vector of size 746 × 5000. Specimen features produced by the endmost convolutional layer of MobilenetV2, Shufflenet, Xception, Darknet53 and EfficientnetB0 are displayed in Fig 2.

The five pre-trained CNNs (MobilenetV2, Shufflenet, Xception, Darknet53 and EfficientnetB0) which achieved best performance in our study are described in the following sections.

2.1.1 MobilenetV2

The first layer of MobilenetV2 is a convolution layer that accepts input images of dimension 224 × 224 × 3 [22]. The model introduced 19 Inverted Residual with Linear Bottle-neck (IRB) layers which accepts a low-dimensional feature representation as input. Pointwise convolution operation is performed to increase the dimension of feature matrix to a high-dimensional one. A depth-wise convolution followed by a pointwise convolution is done subsequently.

2.1.2 Shufflenet

Shufflenet architecture introduced by Zhang et al. [31] consists of pointwise batch convolution operation and channel shuffle (CS) operation. Pointwise batch convolution reduces
**Fig. 1** Architecture of the proposed method. Abbreviations—C: Convolution, BN: Batch Normalization, ReLU: Rectified Linear Unit, IRB: Inverted Residual Block, PC: Pointwise Convolution, Avg Pool: Average Pooling, BC: Batch Convolution, CS: Channel Shuffle, DWC: Depthwise Convolution, Res: Residual

**Table 1** Depth, input size and number of parameters of pre-trained CNNs

| CNN            | Depth | Input size              | Number of parameters (In Millions) |
|----------------|-------|-------------------------|------------------------------------|
| MobilenetV2    | 53    | $224 \times 224 \times 3$ | 3.5                                |
| Shufflenet     | 50    | $224 \times 224 \times 3$ | 1.4                                |
| Xception       | 71    | $229 \times 229 \times 3$ | 22.9                               |
| Darknet53      | 53    | $256 \times 256 \times 3$ | 41                                 |
| EfficientnetB0 | 82    | $224 \times 224 \times 3$ | 5.3                                |

the computation complexity of dense $1 \times 1$ convolution with a cost of weakened representation. Weakened representation caused by pointwise batch convolution can be overcome by using channel shuffle operation which secures input from different batches by dividing the channels in each batch into various groups.

**2.1.3 Xception**

Xception [6] is a pre-trained CNN model based on Inception, where stacks of depthwise separable convolution layers with residual connections substitute inception modules. A depthwise separable convolution is composed of a depthwise convolution which is a spatial convolution operation done separately over each input channel, followed by a $1 \times 1$ convolution operation. The Xception architecture consists of 36 convolutional layers which are grouped into 14 modules, all of which except the first and last modules have linear residual connections among them.
2.1.4 Darknet53

The basic building block of Darknet53 is CBL (Convolution, Batch normalization, Leaky ReLU) layer [20]. Darknet53 is a pre-trained CNN mainly composed of a series of $1 \times 1$ and $3 \times 3$ convolutional layers. The network consists of 5 residual layers which are used to solve the gradient vanishing problem. The residual layers are indicated as Res $n$, where $n$ is the number of residual blocks in a residual layer. The different values of $n$ are 1, 2, 4 and 8.

2.1.5 EfficientnetB0

The main building block for EfficientNetB0 is the Mobile inverted Bottleneck MBC to which squeeze and excitation optimization is added [24]. MBC blocks consist of a layer that performs expansion followed by compression of the channels. Direct connections are used among bottlenecks which connect much lesser channels compared to expansion layers.

2.2 K SVM

Support vector machine (SVM) proposed by Vapnik et al. [25] is a classifier that determines the optimum hyperplane which separates the positive and negative instances of a training data. A training data of $n$ observations can be defined as

$$T = \{(x_1, y_1), (x_2, y_2), \ldots , (x_n, y_n)\}$$

(1)

where $x_i = x_1, x_2, \ldots , x_n$ is an instance of training data and $y_i \in \{\pm 1\}$ is the corresponding label.

A hyperplane in the feature space can be represented as

$$w^T x + b = 0$$

(2)

where $w$ represents the weight matrix and $b$ is the bias term. Distance of an instance $x_i$ from the hyperplane $(w, b)$ is defined as

$$d(w, b; x_i) = \frac{w . x_i + b}{\|w\|}$$

(3)

SVM model proposes a maximum margin separating $w . x_i + b = 1$ and $w . x_i + b = -1$, which can be obtained by solving the optimization problem

$$\min \frac{1}{2} \|w\|^2$$

(4)

conditional on

$$y_i (w^T . x_i + b) \geq 1, \ i = 1, 2, \ldots , n$$

(5)

By introducing a mapping function $\delta$, the vector $x$ in the original low-dimensional space can be transformed into $\delta(x)$ of high-dimensional space. A kernel function is described as

$$K(x_i, x_j) = \delta^T (x_i) \delta (x_j)$$

(6)

The optimization problem in Eq. 7 can be solved in dual form as given below.

$$\min \frac{1}{2} \theta^T M \theta - e^T \theta$$

(7)

conditional on

$$y_i (w^T \delta(x_i) + b) \geq 1 - \epsilon_i, \ \epsilon_i \geq 0, \ i = 1, 2, \ldots , n$$

where $C$ is a regularization parameter and $\epsilon$ is a slack variable used to solve overfitting.

3 Results and discussion

A publicly available dataset consisting of 349 CT scans diagnosed with COVID-19 and 397 CT scans confirmed as non-COVID was used for the study [33]. The area under the receiver operating characteristic curve (AUROC), accuracy, kappa score, positive predictive value (PPV), sensitivity, and F-score were used as the performance metrics. MATLAB 2020b with default parameter settings was used for feature extraction using the CNNs. The classification was performed using LibSVM package of Weka 3.6. Parameter settings of the classifier are given in Table 2.

A 10-fold cross-validation (CV) was performed on the dataset, and the average results of 10-fold were taken into account. Using the same training and test set will give more accurate results when the performance of different models is
Table 2 Parameter Settings

| Parameter      | Value           |
|----------------|-----------------|
| Cost           | 4.5             |
| Degree         | 3               |
| eps            | 0.001           |
| Kernel         | Radial Basis Function (RBF) |
| Loss           | 0.1             |

Fig. 3 Confusion matrix of COVID-19 vs. non-COVID-19 classification. The diagonal element at the bottom shown in green color corresponds to accuracy. Rightmost column values represent sensitivity and bottom-most row elements represent PPV.

Fig. 4 ROC curves corresponding to COVID-19 vs. non-COVID-19 classification.

Two more types of validation were performed to ensure the performance of the method further. Partitions were created using different seed values from 1 to 10 and repeated the experiments 10 times. Ten-fold CV was performed using the training sets and test sets created using each seed value. The average PPV, sensitivity, F-score, kappa score, AUROC and accuracy achieved on repeating the experiments 10 times were 0.9033, 0.9194, 0.9112, 0.8319, 0.9622 and 0.9155, respectively. We also performed validation by dividing the data into a 70% training set and a 30% test set. This type of validation achieved a PPV, sensitivity, F-score, kappa score, AUROC and accuracy of 0.933, 0.866, 0.898, 0.959 and 0.9018, respectively. The results achieved using the three types of validation were comparable.

Table 3 represents the results achieved using various combinations of pre-trained CNNs. Ensemble of 4 pre-trained CNNs consisting of Darknet53, Shufflenet, Xception and EfficientnetB0 also achieved an AUROC of 0.963. However, the proposed 5 CNNs achieved a slightly better accuracy, sensitivity, kappa score and F-score. The combinations of various 4 CNNs selected from the 5 networks of the proposed ensemble also achieved good performance, but slightly lower than the proposed architecture. The results show that ensemble of CNNs, selected based on empirical analysis, could perform better than single CNNs.

Pre-trained networks can be used in two different ways for the classification task. In the proposed method, the pre-trained CNNs serve as a feature extractor, where the activations of a layer are used as features to train an off-the-shelf classifier like KSVM. Pre-trained CNNs can also be used for transfer learning where the layers of the pre-trained network are fine-tuned using the data at hand. Table 4 illustrates the performance of transfer learning using MobilenetV2, Shufflenet, Xception, Darknet53 and EfficientnetB0 with the proposed method. Number of epochs, mini batch size and learning rate were set as 5, 8 and 0.0001, respectively. The proposed method achieved better accuracy and kappa score than transfer learning using pre-trained CNNs. The execution time in minutes of transfer learning using MobilenetV2, Shufflenet, Xception, Darknet53 and EfficientnetB0 was 146.52, 70.08, 312.78, 657.35 and 298.99, respectively, whereas the proposed method took only 10.79 minutes for 10-fold CV. The time required for feature extraction and classification of a single instance was 24.315 seconds.

We selected KSVM as the classifier empirically, after experimenting with other major classifiers. The results attained using various major classifiers along with the proposed ensemble of CNNs are shown in Table 5. Even though...
Table 3 Performance of various CNNs and ensemble of CNNs in combination with KSVM

| Network                               | PPV  | Sensitivity | F-Score | Kappa score | AUROC | Accuracy |
|---------------------------------------|------|-------------|---------|-------------|--------|----------|
| MobilenetV2+Shufflenet+Xception+     | 0.904| 0.917       | 0.91    | 0.8305      | 0.963  | 0.916    |
| Darknet53+EfficientnetB0             |      |             |         |             |        |          |
| MobilenetV2+Shufflenet+Xception+     | 0.892| 0.90        | 0.896   | 0.8036      | 0.958  | 0.902    |
| Darknet53+Alexnet                    |      |             |         |             |        |          |
| MobilenetV2+Shufflenet+Xception+     | 0.892| 0.897       | 0.894   | 0.8009      | 0.958  | 0.901    |
| Darknet53+Shufflenet+Xception+B0     |      |             |         |             |        |          |
| MobilenetV2+Darknet+EfficientnetB0   | 0.90  | 0.90        | 0.90    | 0.8116      | 0.96   | 0.906    |
| MobilenetV2+Shufflenet+Xception+B0   | 0.905| 0.903       | 0.904   | 0.8196      | 0.962  | 0.910    |
| MobilenetV2+Shufflenet+Xception+B0   | 0.892| 0.9         | 0.896   | 0.8036      | 0.959  | 0.902    |
| Xception+Darknet53                   |      |             |         |             |        |          |
| Alexnet+Googlenet+Resnet101          | 0.863| 0.868       | 0.866   | 0.747       | 0.943  | 0.874    |
| Efficientnet+Shufflenet+MobilenetV2  | 0.904| 0.891       | 0.898   | 0.8087      | 0.957  | 0.905    |
| Darknet53+Shufflenet+MobilenetV2     | 0.887| 0.897       | 0.892   | 0.7955      | 0.953  | 0.898    |
| Darknet53+Shufflenet+Xception        | 0.884| 0.897       | 0.89    | 0.7929      | 0.956  | 0.897    |
| MobilenetV2+Shufflenet+Xception      | 0.886| 0.871       | 0.879   | 0.773       | 0.956  | 0.887    |
| MobilenetV2+Darknet+Xception         | 0.881| 0.888       | 0.884   | 0.7821      | 0.954  | 0.891    |
| MobilenetV2+Shufflenet               | 0.87  | 0.883       | 0.876   | 0.766       | 0.883  | 0.883    |
| MobilenetV2+Darknet53                | 0.871| 0.888       | 0.879   | 0.7714      | 0.886  | 0.886    |
| Xception                             | 0.831| 0.848       | 0.84    | 0.6962      | 0.916  | 0.849    |
| Darknet53                            | 0.85  | 0.845       | 0.848   | 0.7145      | 0.93   | 0.857    |
| MobilenetV2                           | 0.84  | 0.857       | 0.848   | 0.7123      | 0.926  | 0.857    |
| Shufflenet                            | 0.839| 0.865       | 0.852   | 0.7148      | 0.926  | 0.858    |
| Alexnet                               | 0.826| 0.805       | 0.816   | 0.6576      | 0.909  | 0.830    |
| Googlenet                             | 0.813| 0.811       | 0.812   | 0.6473      | 0.904  | 0.824    |
| Resnet101                             | 0.844| 0.868       | 0.856   | 0.7259      | 0.928  | 0.863    |
| Densenet201                           | 0.845| 0.857       | 0.85    | 0.7176      | 0.93   | 0.859    |
| EfficientnetB0                        | 0.87  | 0.86        | 0.865   | 0.7468      | 0.948  | 0.873    |
| Resnet18                              | 0.843| 0.828       | 0.835   | 0.7339      | 0.925  | 0.847    |

Performance of the proposed method (Ensemble of CNNs+KSVM) is represented in bold

Table 4 Comparison of the performance of proposed method with transfer learning

| Network                               | PPV  | Sensitivity | F-Score | Kappa score | AUROC | Accuracy |
|---------------------------------------|------|-------------|---------|-------------|--------|----------|
| Proposed                              | 0.904| 0.917       | 0.91    | 0.8305      | 0.963  | 0.916    |
| Xception                              | 0.9398| 0.8241     | 0.8782  | 0.7571      | 0.9590 | 0.8780   |
| MobilenetV2                           | 0.889| 0.874       | 0.882   | 0.687       | 0.923  | 0.843    |
| Shufflenet                            | 0.796| 0.883       | 0.837   | 0.679       | 0.927  | 0.839    |
| Darknet53                             | 0.777| 0.937       | 0.849   | 0.692       | 0.943  | 0.845    |
| EfficientnetB0                        | 0.9427| 0.8246     | 0.8797  | 0.7598      | 0.9659 | 0.879    |

Performance of the proposed method is indicated in bold

all the classifiers produced statistically significant results at $p < 0.05$, their performance was considerably less compared to KSVM.

Majority of the state-of-the-art techniques have used the public dataset used by the proposed method. Few other methods have used private datasets acquired from various...
Table 5: Performance of different classifiers when used along with the proposed ensemble of CNNs

| Classifier | PPV   | Sensitivity | F-Score | Kappa score | AUROC | Accuracy |
|------------|-------|-------------|---------|-------------|-------|----------|
| KSVM       | 0.904 | 0.917       | 0.91    | 0.8305      | 0.963 | 0.916    |
| AdaBoostM1  | 0.691 | 0.691       | 0.691   | 0.4185      | 0.79  | 0.710    |
| Random Forest | 0.805 | 0.768       | 0.718   | 0.6059      | 0.89  | 0.840    |
| Naive Bayes | 0.729 | 0.639       | 0.681   | 0.433       | 0.801 | 0.7198   |
| KNN        | 0.71  | 0.756       | 0.732   | 0.4825      | 0.7776| 0.741    |
| Bayesnet   | 0.726 | 0.668       | 0.696   | 0.4481      | 0.808 | 0.727    |

Results achieved using KSVM are indicated in bold.

Table 6: Performance achieved by different methods

| Method           | Dataset             | PPV   | Sensitivity | F-Score | Kappa score | AUROC | Accuracy |
|------------------|---------------------|-------|-------------|---------|-------------|-------|----------|
| Proposed         | He et al. [33]      | 0.904 | 0.917       | 0.91    | 0.8305      | 0.963 | 0.916    |
| He et al. [13]   | He et al. [33]      | –     | –           | 0.85    | –           | 0.94  | 0.86     |
| Sakagianni et al. [21] | He et al. [33] | 0.8831 | 0.8831     | –       | –           | –     | –        |
| Anwar and Zakir [3] | He et al. [33] | –     | –           | –       | –           | 0.90  | 0.90     |
| Wu et al. [27]   | Private             | –     | 0.811       | –       | –           | 0.819 | 0.76     |
| Pu et al. [19]   | Private             | –     | –           | –       | –           | 0.70  | –        |
| Xu et al. [28]   | Private | 0.813  | 0.867       | 0.839   | –           | –     | 0.867    |
| Mei et al. [15]  | Private             | –     | 0.843       | –       | –           | 0.92  | –        |
| Polsinelli et al. [18] | He et al. [33] | 0.8501 | 0.8755      | 0.8620  | –           | –     | 0.8503   |
| Wang et al. [26] | Private             | –     | 0.974       | –       | –           | 0.991 | –        |
| Di et al. [8]    | Private             | 0.9065 | 0.9327      | –       | –           | –     | 0.8979   |
| Gao et al. [11]  | Private             | –     | 0.8914      | –       | –           | 0.9755| 0.9599   |
| Zhang et al. [32] | Private | –     | 0.9444      | –       | –           | –     | 0.9403   |
| Yener and Oktay [29] | Kaggle, He et al. [33] | 0.90  | 0.94       | 0.92    | –           | 0.91  | 0.91     |
| Ouyang et al. [16] | Private | –     | 0.869       | 0.820   | –           | 0.944 | 0.875    |

Performance attained by the proposed method (ensemble of CNNs+KSVM) is shown in bold.

4 Conclusion

The experimental analysis performed in this work shows the effectiveness of ensemble of features extracted using MobilenetV2, Shufflenet, Xception, Darknet53 and EfficientnetB0 in predicting COVID-19 from CT scans. Features extracted using the aforementioned CNNs in combination with KSVM classifier achieved significant performance in the classification of COVID-19 vs. non-COVID-19. There can be combinations of CNNs that could provide better results. We leave it to the readers to experiment with more combinations of pre-trained CNNs to come up with improved results. A multi-class classification of CT scans into COVID-19, normal, viral pneumonia and bacterial pneumonia using an ensemble of CNNs is proposed as future research work.

References

1. Abraham, B., Nair, M.S.: Computer-aided grading of prostate cancer from mri images using convolutional neural networks. J. Intell. Fuzzy Syst. 36(3), 2015–2024 (2019)
2. Abraham, B., Nair, M.S.: Computer-aided detection of Covid-19 from x-ray images using multi-cnn and bayesnet classifier. Biocybern. Biomed. Eng. 40(4), 1436–1445 (2020)
3. Anwar, T., Zakir, S.: Deep learning based diagnosis of Covid-19 using chest ct-scan images. TechRxiv (2020)
4. Bernheim, A., Mei, X., Huang, M., Yang, Y., Fayad, Z.A., Zhang, N., Diao, K., Lin, B., Zhu, X., Li, K., et al.: Chest ct findings
in coronavirus disease-19 (Covid-19): relationship to duration of infection. Radiology 200463 (2020)
5. Chang, C.C., Lin, C.J.: Libsvm: a library for support vector machines. ACM Trans. Intell. Syst. Technol. (TIST) 2(3), 1–27 (2011)
6. Chollet, F.: Xception: Deep learning with depthwise separable convolutions. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 1251–1258 (2017)
7. Deng, J., Dong, W., Socher, R., Li J.J., Li, K., Fei-Fei, L.: Imagenet: a large-scale hierarchical image database. In: 2009 IEEE conference on computer vision and pattern rec, 248–255 (2009)
8. Di, D., Shí, F., Yan, F., Xia, L., Mo, Z., Ding, Z., Shan, F., Song, B., Li, S., Wei, Y., et al.: Hypergraph learning for identification of Covid-19 with ct imaging. Med. Image Anal. 68(101), 910 (2020)
9. Fakhry, A., Jiang, X., Shi, M., Xia, Y., Yang, K., Chen, J., Li, H., Tu, Y., et al.: Dual-branch combination network for chest ct classification. (2021) arXiv preprint arXiv:2103.03055
10. Fang, Y., Zhang, H., Xie, J., Lin, M., Ying, L., Pang, P., Ji, W.: Sensitivity of chest ct for Covid-19: comparison to rt-PCR. Radiology 296, E115–E117 (2020)
11. Gao, K., Su, J., Jiang, Z., Zeng, L.L., Feng, Z., Shen, H., Rong, P., Xu, X., Qin, J., Yang, Y., et al.: Dual-branch combination network (dcn): towards accurate diagnosis and lesion segmentation of Covid-19 using ct images. Med. Image Anal. 67(101), 836 (2020)
12. Gazda, M., Gazda, J., Plavka, J., Drotar, P.: Self-supervised deep convolutional neural network for chest x-ray classification. (2021) arXiv preprint arXiv:2103.03055
13. He, X., Yang, X., Zhang, S., Zhao, J., Zhang, Y., Xing, E., Xie, P.: Sample-efficient deep learning for Covid-19 diagnosis based on ct scans. (2020) medRxiv
14. Infante, M., Lutman, R., Imparato, S., Di Rocco, M., Ceresoli, G., Torri, V., Morenghi, E., Minuti, F., Cavuto, S., Bottoni, E., et al.: Differential diagnosis and management of focal ground-glass opacities. Eur. Respir. J. 33(4), 821–827 (2009)
15. Mei, X., Lee, H.C., Diao, K.Y., Huang, M., Lin, B., Liu, C., Xie, Z., Ma, Y., Robson, PM., Chung, M., et al.: Artificial intelligence-enabled rapid diagnosis of patients with Covid-19. Nat. Med. 26(8), 1224–1228 (2020)
16. Ouyang, X., Hua, J., Xia, L., Shang, F., Liu, J., Mo, Z., Yan, F., Ding, Z., Yang, Q., Song, B., et al.: Dual-sampling attention network for diagnosis of Covid-19 from community acquired pneumonia. IEEE Transa. Med. Imaging 38, 2595–2605 (2020)
17. Perc, M., Goršiek Miksić, N., Slavinec, M., Stožer, A.: Forecasting Covid-19. Front. Phys. 8, 127 (2020)
18. Polisinielli, M., Cinque, L., Placidi, G.: A light cnn for detecting Covid-19 from ct scans of the chest. (2020) arXiv preprint arXiv:2004.12837
19. Pu, J., Leader, J., Bandos, A., Shi, J., Du, P., Yu, J., Yang, B., Ke, S., Guo, Y., Field, J.B., et al.: Any unique image biomarkers associated with Covid-19? Eur. Radiol. 30, 6221–6227 (2020)
20. Redmon, J.: Darknet: open source neural networks in c (2013)
21. Sakagianni, A., Feretziakis, G., Kalles, D., Koufopoulou, C., Kaldis, V.: Setting up an easy-to-use machine learning pipeline for medical decision support: case study for Covid-19 diagnosis based on deep learning with ct scans. Stud. Health Technol. Inf. 272, 13–16 (2020)
22. Sandler, M., Howard, A., Zhu, M., Zhmoginov, A., Chen, LC.: Mobilenetv2: inverted residuals and linear bottlenecks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 4510–4520 (2018)
23. Sharif Razavian, A., Azizpour, H., Sullivan, J., Carlsson, S.: Cnn features off-the-shelf: an astounding baseline for recognition. In: Proceedings of the IEEE conference on computer vision and pattern recognition workshops, 806–813 (2014)
24. Tan, M., Le, Q.: Efficientnet: rethinking model scaling for convolutional neural networks. In: International Conference on Machine Learning, PMLR, 6105–6114 (2019)
25. Vapnik, V., Guyon, I., Hastic, T.: Support vector machines. Mach. Learn. 20(3), 273–297 (1995)
26. Wang, B., Jin, S., Yan, Q., Xu, H., Luo, C., Wei, L., Zhao, W., Hou, X., Ma, W., Xu, Z., et al.: Ai-assisted ct imaging analysis for Covid-19 screening: building and deploying a medical ai system. Appl. Soft Comput. 106897 (2020)
27. Wu, X., Hui, H., Niu, M., Li, L., Wang, L., He, B., Yang, X., Li, L., Li, H., Tian, J., et al.: Deep learning-based multi-view fusion model for screening 2019 novel coronavirus pneumonia: a multicentre study. Eur. J. Radiol. 128, 109041 (2020)
28. Xu, X., Jiang, X., Ma, C., Du, P., Li, X., Lv, S., Yu, L., Ni, Q., Chen, Y., Su, J., et al.: A deep learning system to screen novel coronavirus disease 2019 pneumonia. Engineering 6, 1122–1129 (2020)
29. Yener, FM., Oktay, AB.: Diagnosis of Covid-19 with a deep learning approach on chest ct slices. In: 2020 Medical Technologies Congress (TIPTEKNO), IEEE, 1–4 (2020)
30. Zhang, J., Xie, Y., Zhang, P., Chen, H., Xia, Y., Shen, C.: Lightweight hybrid convolutional network for liver tumor segmentation. In: IJCAI, 4271–4277 (2019)
31. Zhang, X., Zhou, X., Lin, M., Sun, J.: ShuffleNet: an extremely efficient convolutional neural network for mobile devices. In: Proceedings of the IEEE conference on computer vision and pattern recognition, 6848–6856 (2018)
32. Zhang, Y.D., Satapathy, S.C., Zhu, L.Y., Górriz, J.M., Wang, S.H.: A seven-layer convolutional neural network for chest ct based Covid-19 diagnosis using stochastic pooling. IEEE Sens. J. (2020)
33. Zhao, J., Zhang, Y., Su, J., et al.: A dataset about COVID-19. (2020) arXiv preprint arXiv:2003.13865

Publisher’s Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.