Comparison of Different CNN Models in Tuberculosis Detecting

Jian Liu[1*] and Yidi Huang[1]
[1]School of Computer & Communication Engineering, University of Science and Technology Beijing, Beijing, 100083 - China
[e-mail: liujian@ustb.edu.cn]
[e-mail: yieldingh1996@gmail.com]
*Corresponding author: Jian Liu

Received March 19, 2020; revised June 22, 2020; revised July 20, 2020; revised July 29, 2020; accepted August 9, 2020; published August 31, 2020

Abstract

Tuberculosis is a chronic and delayed infection which is easily experienced by young people. According to the statistics of the World Health Organization (WHO), there are nearly ten million fell ill with tuberculosis and a total of 1.5 million people died from tuberculosis in 2018 (including 251000 people with HIV). Tuberculosis is the largest single infectious pathogen that leads to death. In order to help doctors with tuberculosis diagnosis, we compare the tuberculosis classification abilities of six popular convolutional neural network (CNN) models in the same data set to find the best model. Before training, we optimize three parts of CNN to achieve better results. We employ sigmoid function to replace the step function as the activation function. What's more, we use binary cross entropy function as the cost function to replace traditional quadratic cost function. Finally, we choose stochastic gradient descent (SGD) as gradient descent algorithm. From the results of our experiments, we find that Densenet121 is most suitable for tuberculosis diagnosis and achieve a highest accuracy of 0.835. The optimization and expansion depend on the increase of data set and the improvements of Densenet121.

Keywords: Tuberculosis, CNN, sigmoid, binary cross entropy, SGD.

This work is supported by 2018 Sugon Intelligent-Factory on Advanced Computing Devices (No. MIIT2018-265-137), Fundamental Research Funds for the Central Universities (No. FRF-DF-19-011).

1. Introduction

Tuberculosis is a highly morbid and dangerous respiratory disease. It is a chronic infectious disease caused by mycobacterium tuberculosis that can invade many organs.

In recent years, many countries neglect the tuberculosis and reduce financial investment in tuberculosis prevention. What's more, the rapid population growth and the increase of floating population accelerate the spread of tuberculosis. Tuberculosis has been a serious problem in many developing countries and some developed countries.

Tuberculosis is not only a public health issue, but also a major socio-economic issue. Humans have a long way to go in preventing the spread of tuberculosis. It is a disease that can be controlled and cured as long as we try to do more research. We must achieve scientific strategies on the control of tuberculosis and fight for a long time without interruption. To effectively control tuberculosis, we need to make an effective diagnosis of tuberculosis. With the development of artificial intelligence, we can use convolutional neural network to help doctors in diagnosis of tuberculosis now.

The neural network [1] algorithm originated in the 1940s. It is composed of many connected neurons with adjustable connection weights [2]. It has the characteristics of massive parallel processing, distributed information storage, and good self-learning ability [3]. In the middle of the twentieth century, people did not have powerful computing equipment. The number of parameters in the deep neural network (DNN) is very large so that DNN requires a powerful computing ability. People could just use shallow neural network instead of DNN due to the lack of computing ability. The shallow neural network was not good enough compared with Support Vector Machines (SVM) [4-6], so the neural network algorithm was not popular at that time. In recent years, neural network has received more and more attention with the rapid increase of computer's computing power, especially convolutional neural network (CNN) [7-8]. CNN is a class of feedforward neural networks with convolutional computation and deep structure [9]. It is one of the representative algorithms of deep learning [10]. It has good image classification performance so that it is used in various image classification tasks. CNN is widely used in the diagnosis task of tuberculosis too.

Bogomasov et al. [11] use image processing techniques for feature extraction from CT scans and use artificial neural networks (ANN) for predicting probabilities for different lung irregularities associated with pulmonary tuberculosis and tuberculosis severity assessment. However, they directly use U-net and VGG19 as classification networks without comparing different CNN models. They get an accuracy of 0.6923 and it is too low for clinical use.

Sheen et al. [12] train and evaluate CNN for automatic interpretation of MODS cultures digital images. Though they achieve an average 95.76% accuracy, they use MODS cultures digital images as input images, instead of chest X-ray images. It is hard for hospitals in remote areas to get patients' MODS images, so their method is not suitable for poor areas.

Pasa et al. [13] propose a simple CNN optimized for the problem which is faster and more efficient than previous models but preserves their accuracy. They use a CNN model which consists of 5 convolutional blocks, followed by a global average pooling layer and a fully-connected softmax layer with two outputs. Their CNN model is truly simple, but they get low accuracy. For TB diagnosis, higher accuracy is more important than simpler model structure.

Panicker et al. [14] present an automatic method for the detection of Tuberculosis (TB) bacilli from microscopic sputum smear images. The proposed method performs detection of TB, by image binarization and subsequent classification of detected regions using a CNN. However, different TB patients have different symptoms, someone may not cough and have no phlegm. What's more, they get 78.4% precision and it is not enough for TB detection.

Xiong et al. [15] build a CNN model, named tuberculosis AI (TB-AI), specifically to recognize TB bacillus. They achieve 97.94% sensitivity and 83.65% specificity. The classification object is TB bacillus, so this approach is used to help doctors to distinguish what they see in the microscope. In addition, the training set just contains 45 samples, including 30 positive cases and 15 negative cases. It seems that more training samples should be better.

Kant et al. [16] propose a new DNN for TB diagnosis methodology with recall and precision of 83.78% and 67.55% respectively for bacillus detection from microscopy images of sputum. They use an a 5-layered fully-convoluted neural network architecture and the classification object is TB bacillus too. However, they just achieve 67.55% accuracy.

No one has ever found an efficient CNN model which is able to classify chest X-ray images with high accuracy. In order to find an optimal CNN model to help doctors in diagnosis of tuberculosis, we verify the effect of different CNN models on tuberculosis detection. We test six CNN models which are popular now. During the test process, we also improve these models in different aspects. The tests of six models were performed on the same data set which is composed of chest X-ray images. The six CNN models we tested are DenseNet121 [17], Inception V3 [18], NASNet mobile [19], Resnet50 [20], Vgg16 [21], Xception [22].

The remainder of this paper is as follows. In Section 2, we introduce our data set, the hardware devices we used, the process of data preprocessing. We also simply introduce six convolutional neural network models which are tested by us. Then we will detailly introduce our optimizing options for these CNN models. In Section 3, we will present and analyze the experiment results. In Section 4, we summarize the original text and propose a vision.

2. Preparation and experiments

2.1Data Set

In this paper, we use a data set of chest X-ray images. The number of images in the data set is 800, including 394 tuberculosis images and 406 normal images.

All of the images are labeled by professional doctors from a Chinese hospital and an American hospital. The original sizes of these images are 4020×4892, 2689×3001, 2991×2982. The examples of chest X-ray images are shown in **Fig. 1**.

2.2 Condition

Due to the image sizes are different, we scale the size to 224×224 uniformly. We split 600 chest X-ray images as training set and set the remaining 200 images as test set. The training set includes 304 normal images and 296 tuberculosis images. The test set contains 102 normal images and 98 tuberculosis images.

We use pretrained models from ImageNet and then finetune on our own data set. All of the six models are trained on 4 NVIDIA TITAN Xp.

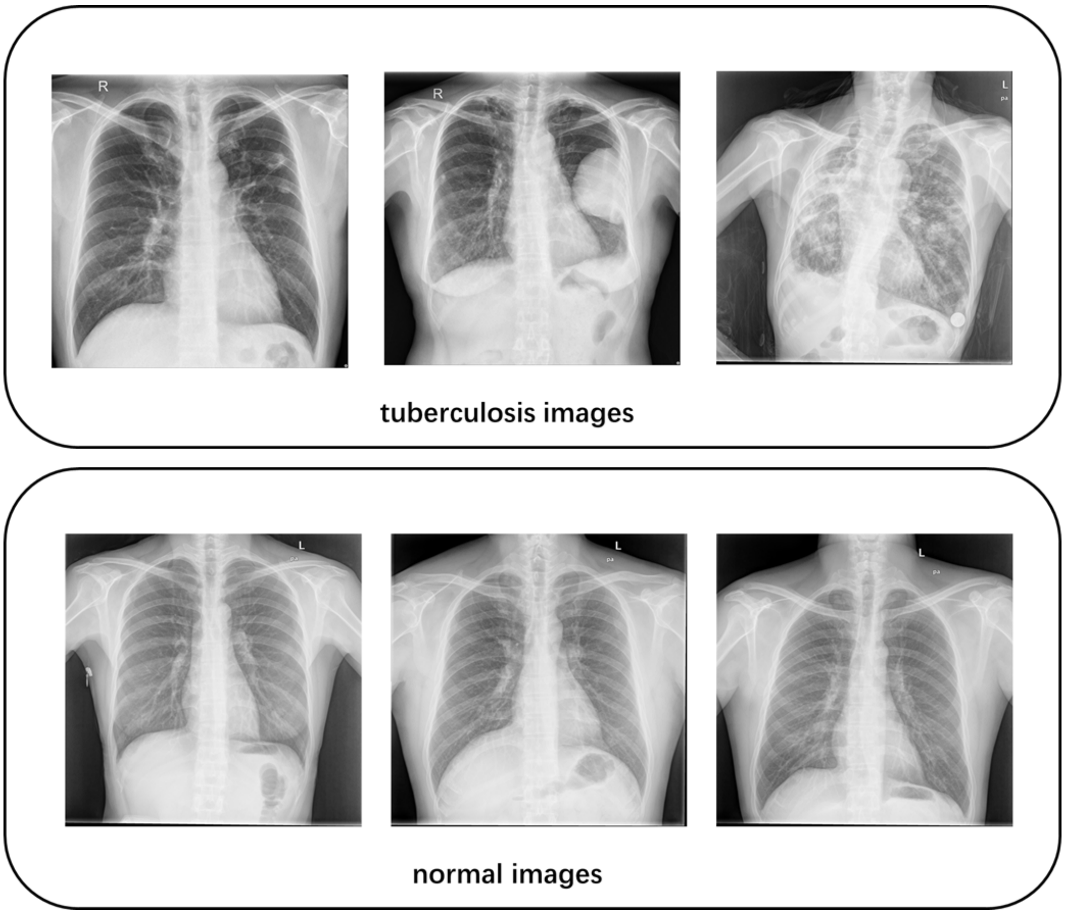

Fig. 1. Representative tuberculosis images (upside) and normal images (downside)

2.3 Introductions of Six Models

In this section, we will simply introduce six models and their improvements compared with traditional convolutional neural network. These models appeared between 2014 and 2018 and we will introduce these models in chronological order.

VGGNet is proposed in 2014 [21]. The researchers of VGGNet successfully construct convolutional neural network with 16~19 layers by repeatedly stacking 3×3 convolution layers and 2×2 maximum pooling layers. Researchers improve VGG's classification performance by continuously deepening the network structure, but the increase in the number of network layers does not lead to a large increase in the number of parameters. Because the number of parameters is mainly determined by the parameters of VGG's last three fully connected layers. At the same time, the researchers reduce the amount of VGGNet parameters by replacing the single large nuclear convolution layer with some small nuclear convolution layers in series. For example, the series of two 3×3 convolution layers are equivalent to one 5×5 convolution layer, and the series of three 3×3 convolution layers are equivalent to one 7×7 convolution layer. However, the number of parameters in three 3×3 convolutional layer is only about half of the number of parameters in one 7×7 convolutional layer. Researchers divide VGGNet into five parts [21], each of which includes the series of multiple 3×3 convolution layers and one maximum pooling layer. There are 3 fully connected layers and a softmax layer at the end of the network. **Fig. 2** shows the structure of VGG16.

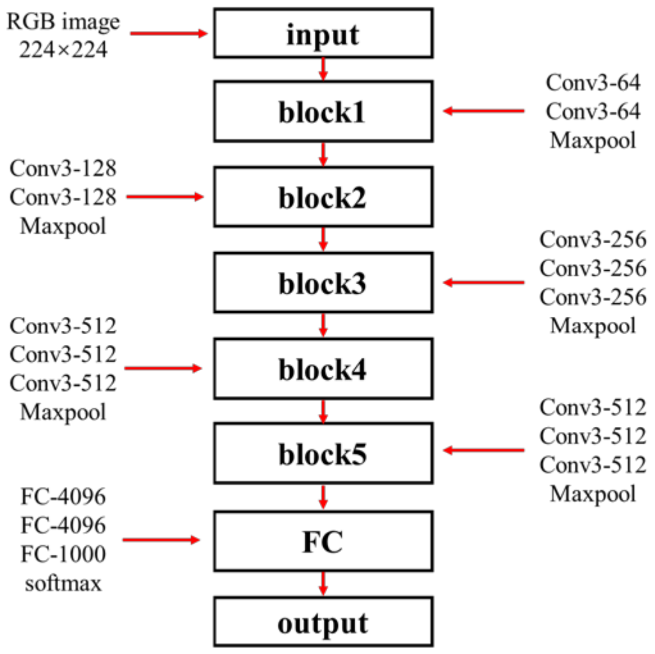

Fig. 2. The model structure of VGG16

Resnet [20] and Inception V3 [18] are proposed in 2016. The researchers of Resnet propose a deep residual learning framework [20]. **Fig. 3** shows an example of residual learning framework. Denote the underlying mapping as $H(x)$ and the output of the nonlinear stacking layer as $F(x) = H(x) - x. F(x)$ is called residual function. We only need to add the output of the nonlinear stacking layer to the input as $H(x) = x + F(x)$ to get the ideal mapping.

Resnet protects information integrity in the way of bypassing input information directly to the output and solves the problem that traditional convolutional networks or fully connected networks may loss information during the process of information transmitted.

The researchers of Inception V3 propose an idea that we can split a large two-dimensional convolution into two smaller one-dimensional convolutions [18]. By using this idea, we can reduce a lot of parameters, accelerate calculations and reduce over-fitting. Let's take the example of replacing a 5×5 convolution layer. From the introduction of VGGNet, we know that we can use two 3×3 convolution layers to replace one 5×5 convolution layer. By replacing, we can reduce the number of parameters and calculation. Now we can use a 1×3 convolution layer and a 3×1 convolution layer to replace a 3×3 convolution layer according to Inception V3. **Fig. 4** shows the replacing process. Actually, we can replace the n×n convolution with a series of 1×n and n×1 convolution, and the calculation can be reduced to the previous 1/n.

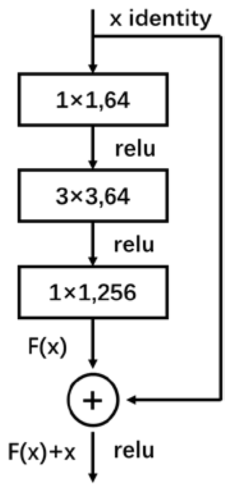

Fig. 3. An example of residual learning block

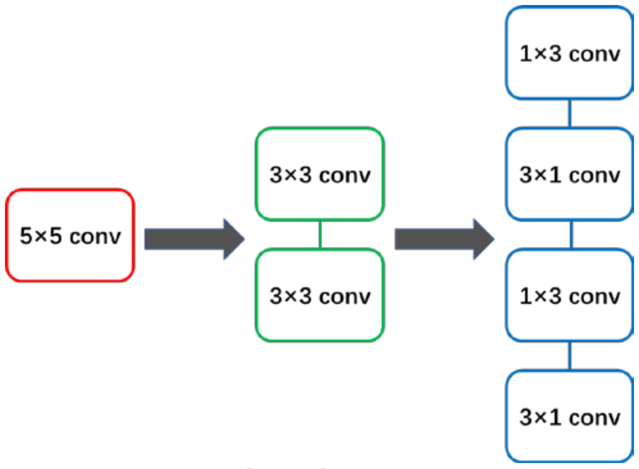

Fig. 4. The replacing process

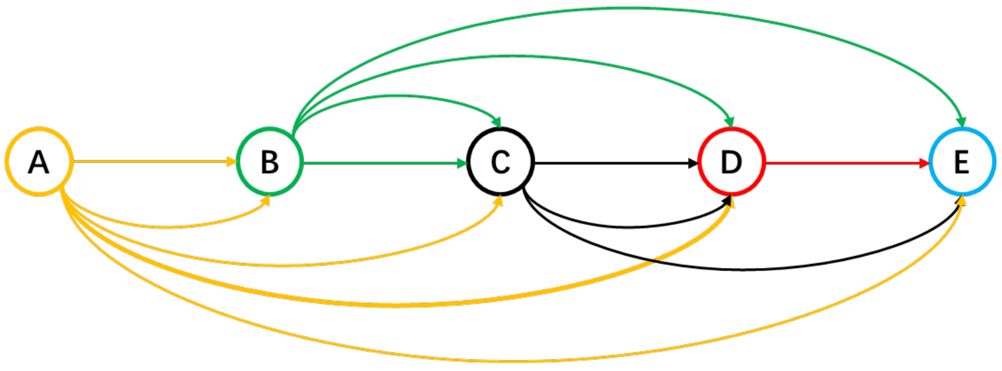

Fig. 5. An example of dense block

DenseNet [17] and Xception [22] are proposed in 2017. DenseNet has deviated from the fixed idea that increasing network layers and widening network structure can improve the classification performance of CNN. DenseNet significantly reduces the amount of network's parameters and solve the problem of the gradient vanishing. What's more, it ensures the maximum information flow between the network layers and encourage feature reuse. The structure of DenseNet is novel and advanced. **Fig. 5** shows an example of dense block. The researchers of DenseNet directly connect all layers to each other. Each layer gets extra input from all the previous layers to maintain the feed-forward characteristic. Because of its densely connected nature, researchers call it the Dense Convolutional Network (DenseNet) [17].

Xception is an improvement of Inception V3 proposed by google. It mainly uses depthwise separable convolutions to reduce the number of network's parameters and improve classification performance at the same time [22].

NASNet mobile is proposed in 2018 [19]. NASNet allows the computer to automatically design a CNN on a small data set [23], such as CIFAR-10. It utilizes migration learning technology [24] to enable the designed network to be migrated to a large dataset, such as ImageNet. The designed network can also be migrated to other tasks. HS Kim et al. [25] apply NASNet, which is an AutoML reinforced learning algorithm, to DeepU-Net neural network that modified U-Net to improve image semantic segmentation performance. Liu Jun et al. [26] aim to address the problem of target detection in collocated multiple-input multiple-output (MIMO) radar where the disturbance covariance matrix is unknown. They can use NASNet in the part of target detection. Cui Qi et al. [27] use a deep neural network to detect photographic images (PI) versus computer generated graphics (CG), and NASNet can also be used to generate a deep neural network.

2.4 Experiments

Before we use our data set to train these six models, we need to do some optimization for CNN.

Firstly, we choose sigmoid function as all of the six models' activation function instead of step function. The formula of the sigmoid function and the derivative of sigmoid function are shown as Equation 1~2. **Fig. 6** shows the sigmoid function and its derivative, we can see that the function curve of sigmoid function is smooth, easy to derive. However, step function does not have these advantages. So sigmoid function is often used as an activation function of neural network.

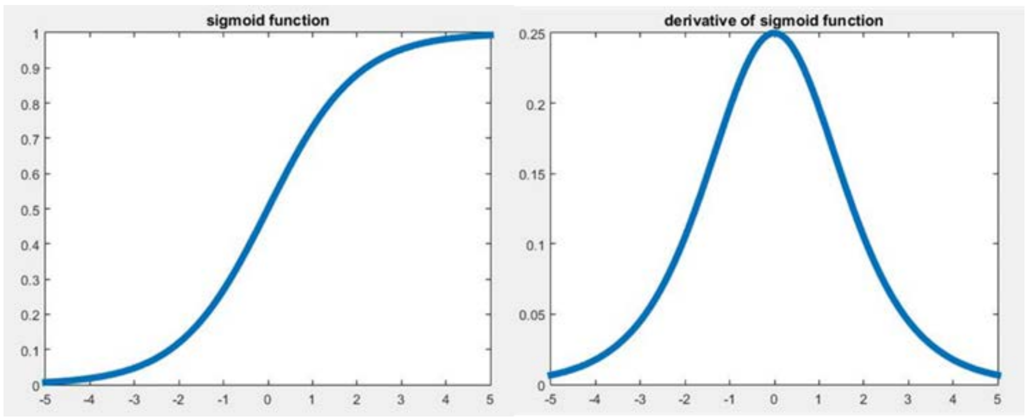

Fig. 6. The image of sigmoid function (left) and its derivative (right)

$$S(x) = \frac{1}{1+e^{-x}} \tag{1}$$

$$S'(x) = \frac{e^{-x}}{(1+e^{-x})^2} = S(x)(1 - S(x)) \tag{2}$$

Secondly, we choose binary cross entropy function as six models' cost function instead of using the traditional quadratic cost function. If we note traditional quadratic cost function as C_1, note single neuron's cost function as C_2, note single neuron's input value and output value as x and a, note weight and bias as w and b, note the real label of x as y, note the input value of sigmoid function as z and note sigmoid function as $\sigma(z)$, we will get these equations as Equation 3~8.

We can see from Equation 7~8 that if $(a - y) \neq 0$, the values of $\frac{\partial C}{\partial w}$ and $\frac{\partial C}{\partial b}$ are mainly decided by $\sigma'(z)$. But the value of $\sigma'(z)$ is always small as it is shown in the right picture of **Fig. 6** (The right picture shows that the value of $\sigma'(z)$ is always between 0 and 0.25). So neural network converges slowly during the training process. We can solve this problem by using binary cross entropy function as cost function. We define binary cross entropy function as C_3 and we can achieve Equation 9~12.

Substituting Equation 11~12 into Equation 10, we can get Equation 13. It is the formula for weights with new cost function, the formula for bias is similar and we record it as Equation 14.

$$C_1(w, b) = \frac{1}{2n} \sum_x (y - a)^2 \tag{3}$$

$$C_2(w, b) = \frac{(y-a)^2}{2} \tag{4}$$

$$z = wx + b \tag{5}$$

$$a = \sigma(z) \tag{6}$$

$$\frac{\partial C}{\partial w} = (a - y)\sigma'(z)x \tag{7}$$

$$\frac{\partial C}{\partial b} = (a - y)\sigma'(z) \tag{8}$$

$$C_3 = -\frac{1}{n} \sum_x [y \ln a + (1 - y) \ln(1 - a)] \tag{9}$$

$$\frac{\partial C}{\partial w_j} = -\frac{1}{n} \sum_x \left(\frac{y}{\sigma(z)} - \frac{(1-y)}{1-\sigma(z)} \right) \frac{\partial \sigma}{\partial w_j} \tag{10}$$

$$\frac{\partial \sigma}{\partial w_j} = \sigma'(z) x_j \tag{11}$$

$$\sigma(z) = 1/(1+e^{-z}) \tag{12}$$

$$\frac{\partial C}{\partial w_j} = \frac{1}{n}\sum_x x_j(\sigma(z)-y) \tag{13}$$

$$\frac{\partial C}{\partial b} = \frac{1}{n}\sum_x(\sigma(z)-y) \tag{14}$$

We can see from Equation 13~14 that the value of $\frac{\partial C}{\partial w_j}$ and $\frac{\partial C}{\partial b}$ are decided by $(\sigma(z)-y)$. It is a value that measures the difference between the output value of the neural network and the real label. Usually, we call it error. The value of error is always large at the beginning of training. It is getting smaller and smaller during training and it is always small at the end of training. When the value of error is large, $|\frac{\partial C}{\partial w_j}|$ and $|\frac{\partial C}{\partial b}|$ will be large too. That means fast refresh rates of bias and weights. On the contrary, it means slow refresh rates bias and weights. Generally speaking, we need fast refresh rates at the beginning of the training to speed up the training process. We need slow refresh rates to avoid missing the best values of weights and bias when the output value of the neural network is close to the value of real label. Achieving the best values of weights and bias means better classification performance.

As cost function, binary cross entropy function has special advantages. We can both speed up the training process and get better classification performance by using binary cross entropy. However, using the traditional quadratic cost function means a long training process and low classification accuracy. It is obvious that binary cross entropy function is more suitable to be CNN models' cost function.

Finally, we choose stochastic gradient descent (SGD) as gradient descent algorithm. If we note learning rate as η, note new weights and bias as w' and b', note gradient vector as ∇C, we will get the update equations of weights and bias as equation 15~17.

$$\nabla C = (\frac{\partial C}{\partial w}, \frac{\partial C}{\partial b})^T \tag{15}$$

$$w'_k = w_k - \eta\nabla C = w_k - \eta\frac{\partial C_x}{\partial w_k} \tag{16}$$

$$b'_l = b_l - \eta\nabla C = b_l - \eta\frac{\partial C_x}{\partial b_l} \tag{17}$$

$$\frac{\sum_{j=1}^{m}\nabla C_{x_j}}{m} \approx \frac{\sum_x \nabla C_x}{n} = \nabla C \tag{18}$$

$$w'_k = w_k - \frac{\eta}{m}\sum_j \frac{\partial C_{x_j}}{\partial w_k} \tag{19}$$

$$b'_l = b_l - \frac{\eta}{m}\sum_j \frac{\partial C_{x_j}}{\partial b_l} \tag{20}$$

We have to calculate the gradient vector for each instance x in the training set. If training set contains too many instances, training process will cost too much time. To solve this problem, we decide to use SGD. The core idea of the SGD algorithm is to take a small sample set from the training set and use it to estimate ∇C. We note the number of instances in training set and sample set as n, m. If the number of instances in the sample set is large enough, we will get the estimation formula of ∇C and new update equations of weights and bias as Equation 18~20.

We call the number of instances in the sample set mini-batch. After the neural network has learned all the instances in the current mini-batch, SGD will reselect a new mini-batch for training. When all training examples are used up, neural network finish one training epoch. Compared with traditional gradient descent (TGD), using SGD can save much time during the training process. Less training time does not mean lower classification accuracy. In fact, choosing TGD or SGD has little effect on classification accuracy. In a word, SGD is more suitable than TGD as gradient descent algorithm.

3. Results

First of all, we compare the performance of different combinations of cost function and gradient descent algorithm in all six CNN models. We note traditional quadratic cost function as TQCF, note binary cross entropy function as BCEF, note traditional gradient descent as TGD, note stochastic gradient descent as SGD. The results are shown in **Table 1~6**. We can see that BCEF+SGD gets the best performance in all six CNN models. BCEF+SGD achieves the highest classification accuracy with the least training epochs. In addition, BCEF+SGD costs the least training time for 500 epochs. So BCEF+SGD is the best combination for all six CNN models.

We now compare six CNN models. We can see that DenseNet121 gets the highest accuracy of 0.835 and Xception gets the lowest accuracy. Dense block and fully connected structure, which are DenseNet121's special advantages, are suitable for retaining key features in chest X-ray images. Using depthwise separable convolutions to reduce the number of network's parameters, which is used by Xception, does not work well in chest X-ray images. We can also see that Resnet50 takes the least time for 500 training epochs and use the fewest training epochs to achieve the highest accuracy. NASNet mobile takes the most time for 500 training epochs. Because the total number of convolutional layers and max pool layers of Resnet50 is 50, its model depth is shallowest. However, the model structure of NASNet mobile is too complex to take much time for each epoch of training.

Table 1. Classification performance of four combinations (DenseNet121)

combination	training epochs for highest accuracy	training time for 500 epochs (minutes)	accuracy
TQCF+TGD	684	163.6	0.820
BCEF+TGD	436	153.3	0.835
TQCF+SGD	703	73.2	0.815
BCEF+SGD	422	58.6	0.835

Table 2. Classification performance of four combinations (Inception V3)

combination	training epochs for highest accuracy	training time for 500 epochs (minutes)	accuracy
TQCF+TGD	655	149.3	0.800
BCEF+TGD	473	127.5	0.805
TQCF+SGD	694	81.7	0.785
BCEF+SGD	404	72.9	0.815

Table 3. Classification performance of four combinations (NASNet mobile)

combination	training epochs for highest accuracy	training time for 500 epochs (minutes)	accuracy
TQCF+TGD	557	486.3	0.780
BCEF+TGD	521	466.6	0.810
TQCF+SGD	589	243.8	0.765
BCEF+SGD	503	209.5	0.810

Table 4. Classification performance of four combinations (Resnet50)

combination	training epochs for highest accuracy	training time for 500 epochs (minutes)	accuracy
TQCF+TGD	596	154.3	0.770
BCEF+TGD	402	142.9	0.805
TQCF+SGD	557	67.8	0.785
BCEF+SGD	371	52.9	0.810

Table 5. Classification performance of four combinations (Vgg16)

combination	training epochs for highest accuracy	training time for 500 epochs (minutes)	accuracy
TQCF+TGD	886	267.4	0.725
BCEF+TGD	697	246.0	0.790
TQCF+SGD	863	127.5	0.755
BCEF+SGD	659	98.9	0.805

Table 6. Classification performance of four combinations (Xception)

combination	training epochs for highest accuracy	training time for 500 epochs (minutes)	accuracy
TQCF+TGD	736	186.3	0.720
BCEF+TGD	569	199.0	0.775
TQCF+SGD	734	104.5	0.735
BCEF+SGD	515	96.7	0.785

Table 7. The initialization of training.

Names	Initialization
Image size	224×224
Learning rate	0.0005
Batch size	10
Epochs for training	900
Cost function	binary cross entropy
Optimizer	SGD
Framework	Tensorflow and Keras
Hardware	NVIDIA GeForce TITAN XP

Now we need to compare the classification capabilities of six CNN models from multiple indicators. After the first experiment, we decide to set binary cross entropy as cost function and set SGD as optimizer for all CNN models. We also need to keep other training parameters consistent. We set other training parameters such as learning rate, batch size and so on. The initialization of training is setting as shown in **Table 7**.

We compare the classification performance of six CNN models. The classification performance is evaluated in terms of accuracy (ACC), precision (PPV), sensitivity (SEN) and specificity (SPE). We take the tuberculosis as a negative example and take the normal as a positive example. The respective definitions of these common metrics adopt true positive (TP), true negative (TN), false positive (FP), and false negative (FN).

Now we can achieve Equation 21~24. ACC refers to the ratio of the number of correctly predicted samples to the total number of predicted samples. ACC does not consider whether the predicted samples are positive or negative. PPV refers to the ratio of the number of positive samples correctly predicted to the number of all positive samples predicted. PPV only focuses on the part that is predicted to be a positive sample. SEN is also called true positive rate (TRP), it refers to the ratio of the number of correctly predicted positive samples to the total number of true positive samples. SEN reflects the ability of screening tests to find patients. SPE refers to the ratio of correctly predicted negative samples to the total number of true negative samples.

$$\mathrm{ACC} = (\mathrm{TP}+\mathrm{TN})/(\mathrm{TP}+\mathrm{TN}+\mathrm{FP}+\mathrm{FN}) \tag{21}$$

$$\mathrm{PPV} = \mathrm{TP}/(\mathrm{TP}+\mathrm{FP}) \tag{22}$$

$$\mathrm{SEN} = \mathrm{TP}/(\mathrm{TP}+\mathrm{FN}) \tag{23}$$

$$\mathrm{SPE} = \mathrm{TN}/(\mathrm{TN}+\mathrm{FP}) \tag{24}$$

The results of our experiments are shown in **Table 8**. From the results, we can see that the classification accuracy of six models is around 0.8. Densenet121 achieves the highest classification accuracy and the classification accuracy of Xception is the lowest.

The test set contains 102 normal images and 98 tuberculosis images. The confusion matrices are shown in **Fig. 7**. We can see that the number of FP is larger than the number of FN and TN is less than TP only for VGG16. Therefore, SEN of VGG16 is the highest in all 6 CNN models. For the other 5 CNN models, FN is more than FP and TP is less than TN. Densenet121 gets the most TP and TN. What's more, Densenet121 achieves the least FN and FP. Therefore, it gets the highest ACC, PPV and SPE. If we comprehensively analyze the values of ACC, PPV, SEN and SPE, we will find the performance of Densenet121 is the best in six models.

Table 8. Classification performance of six CNN models

Model	ACC	PPV	SEN	SPE
Densenet121	**0.835**	**0.863**	**0.822**	**0.849**
Inception v3	0.815	0.843	0.804	0.828
NASNet mobile	0.810	0.853	0.791	0.833
Resnet50	0.810	0.824	0.808	0.813
VGG16	0.805	0.775	0.832	0.781
Xception	0.785	0.794	0.786	0.784

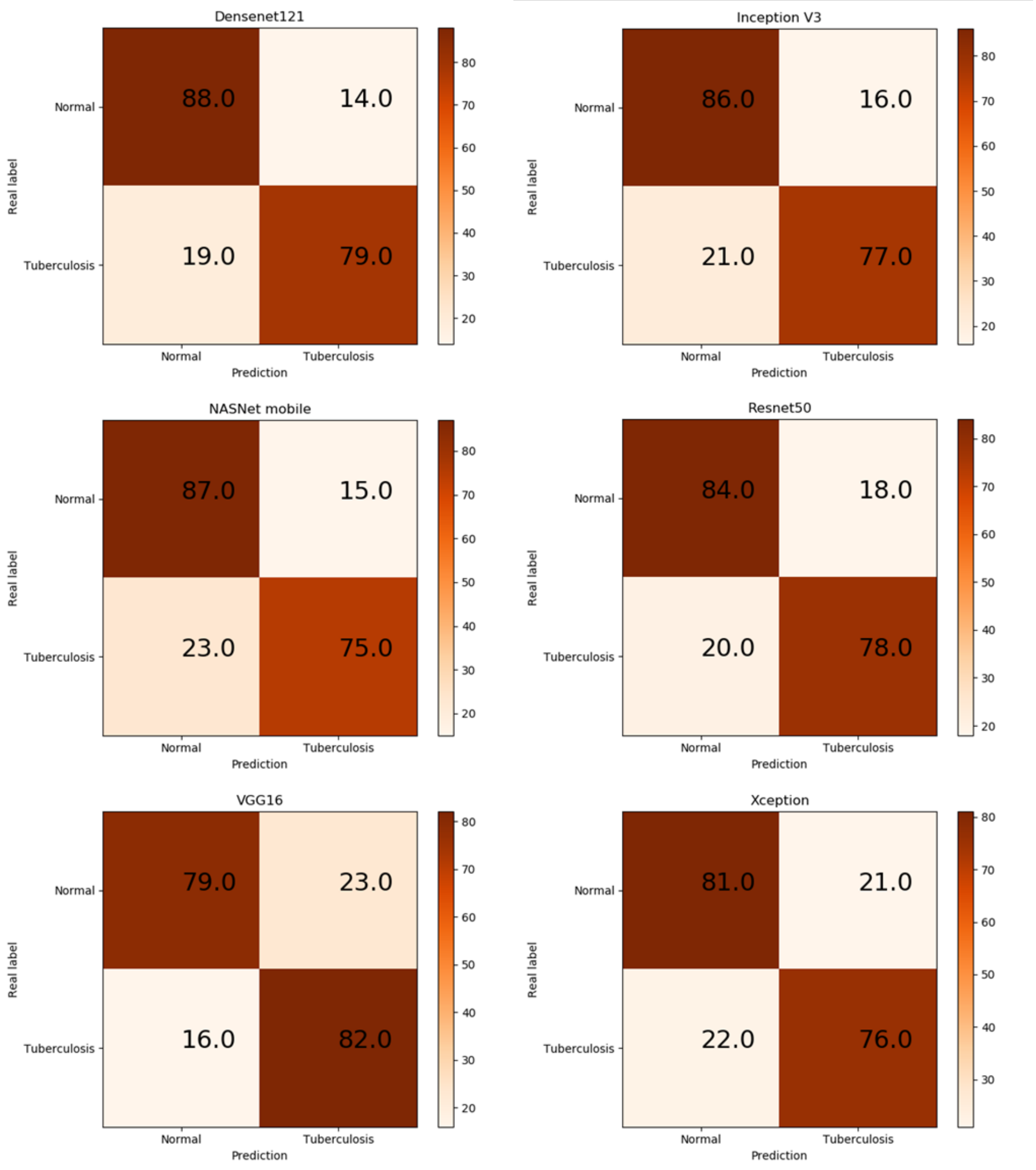

Fig. 7. Confusion matrix of six models

4. Conclusion

In this paper, we use six different convolutional neural network models to classify the same data set composed of chest X-ray images. Before training, we take three steps to optimize all the CNN models. Firstly, we use sigmoid function to replace the step function as the activation function, so that the activation function can be derived in the entire domain of definition. Secondly, we use binary cross entropy function as the cost function to replace traditional quadratic cost function in order to get the best values of CNN's weights and bias. Thirdly, we choose stochastic gradient descent (SGD) as gradient descent algorithm to speed up the training process. The results of our experiments show that the classification performance of Densenet121 is the best in six models. So Densenet121 is the most suitable CNN model for tuberculosis diagnosis. The research on the diagnosis of tuberculosis is very meaningful and important, there are still many shortcomings in our work. We will try to do more research on tuberculosis diagnosis and offer more effective measures to optimize CNN models in the future. We believe that the classification accuracy will be improved with a more powerful model and a larger data set.

References

[1] Zhang, Xiangyu, et al, "Shufflenet: An extremely efficient convolutional neural network for mobile devices," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2018. Article (CrossRef Link)

[2] Torlai, Giacomo, et al., "Neural-network quantum state tomography," *Nature Physics*, 14(5), 447-450, 2018. Article (CrossRef Link)

[3] Acharya, U. Rajendra, et al., "Deep convolutional neural network for the automated detection and diagnosis of seizure using EEG signals," *Computers in biology and medicine*, 100, 270-278, 2018. Article (CrossRef Link)

[4] Chen, Yuantao, et al., "The fire recognition algorithm using dynamic feature fusion and IV-SVM classifier," *Cluster Computing*, 22(3), 7665-7675, 2019. Article (CrossRef Link)

[5] Zhang, Jianming, et al., "Spatial and semantic convolutional features for robust visual object tracking," *Multimedia Tools and Applications*, 79, 15095-15115, 2020. Article (CrossRef Link)

[6] Gui, Yan, and Guang Zeng, "Joint learning of visual and spatial features for edit propagation from a single image," *The Visual Computer*, 36(3), 469-482, 2020. Article (CrossRef Link)

[7] Zeng, Daojian, et al., "Aspect based sentiment analysis by a linguistically regularized CNN with gated mechanism," *Journal of Intelligent & Fuzzy Systems*, 36(5), 3971-3980, 2019. Article (CrossRef Link)

[8] Liu, Jin, et al., "Attention-based BiGRU-CNN for Chinese question classification," *Journal of Ambient Intelligence and Humanized Computing*, 1-12, 2019. Article (CrossRef Link)

[9] Gu, Jiuxiang, et al., "Recent advances in convolutional neural networks," *Pattern Recognition*, 77, 354-377, 2018. Article (CrossRef Link)

[10] Levine, Sergey, et al., "Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection," *The International Journal of Robotics Research*, 37(4-5), 421-436, 2018. Article (CrossRef Link)

[11] Bogomasov, Kirill, et al., "Feature and Deep Learning Based Approaches for Automatic Report Generation and Severity Scoring of Lung Tuberculosis from CT Images," in *Proc. of CLEF2019 Working Notes*, 2380, 9-12, 2019. Article (CrossRef Link)

[12] Lopez-Garnier, Santiago, Patricia Sheen, and Mirko Zimic, "Automatic diagnostics of tuberculosis using convolutional neural networks analysis of MODS digital images," *PloS one*, 14(2), 2019. Article (CrossRef Link)

[13] Pasa, F., et al., "Efficient deep network architectures for fast chest x-ray tuberculosis screening and visualization," *Scientific reports*, 9(1), 1-9, 2019. Article (CrossRef Link)
[14] Panicker, Rani Oomman, et al., "Automatic detection of tuberculosis bacilli from microscopic sputum smear images using deep learning methods," *Biocybernetics and Biomedical Engineering*, 38(3), 691-699, 2018. Article (CrossRef Link)
[15] Xiong, Yan, et al., "Automatic detection of mycobacterium tuberculosis using artificial intelligence," *Journal of thoracic disease*, 10(3), 19-36, 2018. Article (CrossRef Link)
[16] Kant, Sonaal, and Muktabh Mayank Srivastava, "Towards automated tuberculosis detection using deep learning," in *Proc. of 2018 IEEE Symposium Series on Computational Intelligence (SSCI). IEEE*, 2018. Article (CrossRef Link)
[17] Huang, Gao, et al., "Densely connected convolutional networks," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2017. Article (CrossRef Link)
[18] Szegedy, Christian, et al., "Rethinking the inception architecture for computer vision," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2016. Article (CrossRef Link)
[19] Zoph, Barret, et al., "Learning transferable architectures for scalable image recognition," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2018. Article (CrossRef Link)
[20] He, Kaiming, et al., "Deep residual learning for image recognition," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2016. Article (CrossRef Link)
[21] Simonyan, Karen, and Andrew Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556*, 2014. Article (CrossRef Link)
[22] Chollet, François, "Xception: Deep learning with depthwise separable convolutions," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2017. Article (CrossRef Link)
[23] Zoph, Barret, and Quoc V. Le, "Neural architecture search with reinforcement learning," *arXiv preprint arXiv:1611.01578*, 2016. Article (CrossRef Link)
[24] Kornblith, Simon, Jonathon Shlens, and Quoc V. Le, "Do better imagenet models transfer better?," in *Proc. of the IEEE conference on computer vision and pattern recognition*, 2019. Article (CrossRef Link)
[25] Kim, H. S., Yoo, K. Y., & Kim, L. H., "Improved Performance of Image Semantic Segmentation using NASNet," *Korean Chemical Engineering Research*, 57(2), 274-282, 2019.
[26] Liu, Jun, et al., "Target detection exploiting covariance matrix structures in MIMO radar," *Signal Processing*, 154, 174-181, 2019. Article (CrossRef Link)
[27] Cui, Qi, Suzanne McIntosh, and Huiyu Sun, "Identifying materials of photographic images and photorealistic computer generated graphics based on deep CNNs," *Computers, Materials & Continua*, 55(2), 229-241, 2018. Article (CrossRef Link)

Jian Liu received his B.S. degree in Automatic Control Theory and Applications from Shandong University, China, in 2000, and the Ph.D. degree in School of Information Science and Engineering from Shandong University in 2008. He is currently an associate professor of University of Science and Technology (USTB), Beijing, China. His research interests include artificial intelligence, next generation Wireless Networks, cognitive radio networks, and mobile mesh networks. He is an IEEE member since 2009.

Yidi Huang received his B.S. degree in Communication Engineering from University of Science and Technology Beijing, China, in 2018. He is currently a pursuing master of University of Science and Technology (USTB), Beijing, China. His research interests include artificial intelligence, computer algorithm, computer vision and human-computer interaction.