IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received June 23, 2020, accepted July 2, 2020, date of publication July 23, 2020, date of current version August 12, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3007599

OCT Image Recognition of Cardiovascular Vulnerable Plaque Based on CNN

JINGBO WANG
School of Life Science, Beijing Institute of Technology, Beijing 100081, China
College of Electrical Engineering, North China University of Science and Technology, Tangshan 063009, China
e-mail: allp413@ncst.edu.cn

ABSTRACT Optical coherence tomography (OCT) shows an important role in the diagnosis of cardiovascular diseases and the detection and intervention of vulnerable plaques. Clinical diagnosis of vulnerable plaques in cardiovascular system based on optical coherence tomography mainly relies on the manual analysis of vulnerable plaques in images by cardiovascular physicians. This analysis process is prone to subjective misjudgments and heavy workload. Studying the recognition technology of cardiovascular vulnerable plaque image will greatly improve the accuracy of diagnosis and reduce the workload of cardiovascular physicians, which is an effective way to achieve efficient diagnosis and treatment. In view of the recognition of cardiovascular vulnerable plaque medical images, a model based on convolution neural network (CNN) recognition is constructed. The CNN is used to learn the features of different levels from the original input OCT images. At the same time, several decision-making levels are designed. These decision-making levels can classify OCT images according to different feature maps, and finally make final recognition decisions according to the classification results. The experimental results on the clinical data set labeled by doctors show that the classification and recognition model of cardiovascular vulnerable plaque OCT image based on CNN has a high recognition rate. Making full use of the multilevel features of convolutional neural networks can effectively classify and recognize the OCT images of vulnerable cardiovascular plaques, provide support for the clinical diagnosis of cardiovascular diseases, and have great significance for the early intervention and prevention of cardiovascular diseases.

INDEX TERMS Convolutional neural network, vulnerable plaque, OCT image, image recognition.

I. INTRODUCTION

More than 20 million people worldwide experience acute heart disease such as acute coronary syndrome (ACS) and/or sudden cardiac death each year. ACS refers to acute myocardial ischemia syndrome, which eventually manifests as thrombosis in the coronary arteries. The pathology of ACS is based on vulnerable plaque rupture, erosion or calcified nodules. Among them, vulnerable plaque breakage is the most common and most important type in clinical practice. Vulnerable plaque (VP) refers to plaques that are unstable and have a tendency to form thrombi. These plaques are prone to rupture and cause acute coronary stenosis. Pathologically vulnerable plaque rupture is defined as a plaque containing a necrotic nucleus. Relevant data show that more than 70% of patients with coronary heart disease death are given rise to break of acute coronary plaque. So, early appraisal of vulnerable-plaque and active intervention to take effective measures is essential to reduce mortality in patients with coronary heart-disease.

Currently, intravascular-ultrasound (IVUS) is the most commonly used clinically to identify coronary plaques. It can evaluate the structure of the arterial wall and give us a deeper understanding of the physiology of coronary atherosclerosis, but the resolution of IVUS is not high enough to accurately identify the thickness of the fibrous membrane and the boundary between the intima and the media. In recent years, optical-coherence-tomography (OCT) has been increased as a new intracoronary imaging technique at home and abroad. OCT is a new imaging method based on light, which can be used as a detection method for coronary vulnerable plaque. Intracoronary OCT techniques use infrared light waves to image the structure of the coronary wall, providing a high-resolution cross-sectional image of the lumen, including

The associate editor coordinating the review of this manuscript and approving it for publication was Zhihan Lv.

lumen diameter, thrombus, plaque, dissection, tissue, and stent position. Intracoronary OCT is currently the highest spatial resolution technology. The minimum resolution of OCT is 10μm, which is 10 times that of IVUS. It is the highest resolution imaging technique currently known. The extremely high resolution of the OCT and its ability to differentiate the coronary lumen and vessel wall structures enable rapid and reliable cross-sectional images of coronary vessels. OCT can directly observe vulnerable plaque, detect the degree of stenosis of the coronary lumen, and distinguish different tissue-types such as fibrous tissue, necrotic tissue, lipid-rich and calcified tissue; OCT can also identify important characteristic lesions of vulnerable plaque, such as thin fibrous cap-atheroma, lipid-core, thin fibrous-cap thickness, plaque-calcification, massive macrophage infiltration, and thrombosis formation, and the sensitivity of identifying fiber caps and lipid cores is more than 90%. The ability of OCT to identify vulnerable plaques helps clinicians determine which types of plaques to intervene, which is important to prevent the occurrence of adverse cardiac events.

In the identification of vulnerable plaques of coronary arteries, the traditional way of identification is that the doctor visually discriminates the images to obtain a diagnosis conclusion. This method is not only wasting time and energy, subjective, and also may cause misdiagnosis, which is very unfavorable for early prevention and intervention of vulnerable plaque lesions [1] (CNN). This method has been triumphantly applied to handwritten character-recognition, face-recognition, human eye-detection, pedestrian-detection, robot navigation and has become a research hotspot. Generally, a convolutional layer, also known as a feature extraction layer, is connected to the local receptive field of the previous layer and extracts the local features.

The technique of image feature extraction is derived from the development of machine vision. The computer extracts the relevant pixel points or regions as pictures, and analyzes the pixel points or the area blocks, so that the whole process of conveniently identifying the pictures is the picture feature extraction. From the perspective of mapping or transformation, the feature extraction of the picture performs a series of transformation or mapping methods on the group measurement values, thereby achieving the purpose of highlighting the representative features of a certain mode. There are many image-based feature-extraction methods, like scale-invariant feature transform (SIFT) [2], Histogram Of Gradient (HOG) [3], convolutional neural network (CNN), wavelet transform, Local binary pattern (LBP) [4] and so on. The features extracted by each method have their own advantages and disadvantages, and the features extracted by each method of extracting features have obvious differences in the final recognition results of different images, especially some very simple feature extraction methods. These methods extract features with a single feature and are not representative. Although SIFT and CNN are universal, and the extracted features have good results for most pictures, if a feature extraction method is used alone, it is likely to over-fit. For a certain class, the image may have a very good classification effect, but the results for other images are not ideal.

II. PROPOSED METHOD

A. IMAGE FEATURE EXTRACTION

For a mature image recognition system, the main steps can be divided into four major steps: image acquisition and preprocessing, target image localization and segmentation, image feature extraction, feature recognition and classification. Image acquisition and preprocessing process, firstly through the camera or image acquisition device to capture images, and then pre-process the images to be processed to reduce or even eliminate the noise, thereby improving the separability between the image target and the background; Image localization and segmentation is to locate the recognition target area of the denoised photo and segment it with a specific algorithm; Image feature extraction is the extraction of abstract features for images segmented into regular sizes to facilitate computer coding and recognition; Feature recognition and classification is to use the extracted abstract image features to be sent to the classifier to calculate the probability size of a certain class, and finally select the tag corresponding to the maximum probability to classify and identify the image. So far, image features based on image recognition can be divided into the following four types:

(1) Image visual features: The physical characteristics of the image can be directly observed. For example: color, shape, texture, outline and other features.

(2) Manual features: The characteristics of certain statistical information of an image are obtained by calculation. These methods are widely used in some specific image recognition fields.

(3) Learning characteristics: Features learned through machine learning methods and some shallow artificial neural networks.

(4) Deep learning characteristics: feature extraction methods by stacking some shallow neural networks to form deep neural networks. For example: deep convolutional neural network [5], deep belief network [6], circulating neural network [7] and so on.

Among the four image features introduced above, visual features are difficult to express in a computer due to their own physical characteristics, making them inappropriate for image recognition. Manual features have been popular in past image recognition programs and are widely used in some specific recognition fields. However, these features are best designed for a single recognition problem, making the generalization of their features not very strong and not suitable for general image recognition programs. For example, histogram of oriented gradient is designed for the pedestrian detection and recognition problem. It is composed of the gradient-direction-histogram of the local region of the image and is used in the pedestrian detection and recognition system [8]; Local Binary Pattern is originally designed for the face recognition problem. However, these features are difficult to play their due role in the identification problem outside the design itself. For learning features, the corresponding features can be extracted for identifying differences in the problem and the data. For example, the PCA method and the LDA method map image data to a low-dimensional space by using the extracted feature vector, so that the projected features are easier to classify than the original data. Shallow neural networks use different connections to form different networks by simulating the bioelectrical signal transmission of the brain. The weight is updated and corrected as the gradient is reversed. However, due to the limited computer performance at the time, these machine learning algorithms are relatively simple in structure, which makes the expression ability limited and cannot extract special abstract and valuable features, so that the recognition results are not significantly improved compared to the recognition results using manual features.

With the introduction of various high-speed processors, especially the development of graphics processing unit (GPU) in the field of parallel processing, more complex and more abstract deep learning models can be implemented on computers. Deep convolutional neural networks share the characteristics of local receptive fields because of their unique characteristics, and its multi-level network structure is particularly conducive to the expression of abstract features, making this method widely concerned by more and more scholars at home-abroad. Especially in ImageNet, the world image-recognition contest held in recent days, the recognition-algorithm based on deep CNN has won five consecutive championships, namely AlexNet [9] in 2012, ZFNet [10] in 2013, and 2014. VGG [11] and GoogleNet [12] and ResNet [13] in 2015 prove that this method has great research and practical value.

Image feature extraction is to get the abstracted features from the preprocessed images, so that the extracted image features are suitable for the computer to identify, detect and other tasks as much as possible. Deep convolutional neural networks are widely used in various image recognition algorithms due to their superior image feature extraction capabilities.

B. CONVOLUTIONAL NEURAL NETWORK

A common CNN consists of input-layer, convolution-layer, activation-layer, pooling-layer, fully connected-layer, and final output-layer, and its structure is as shown in Figure 1.

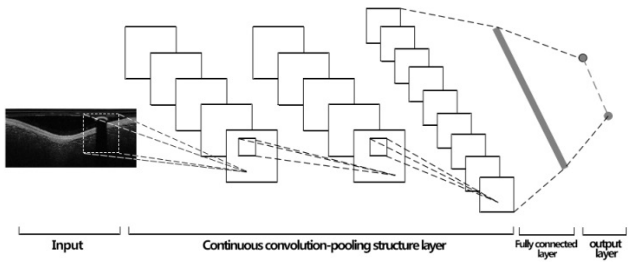

FIGURE 1. **Typical convolutional neural network model structure.**

In mathematics, convolution is a significant analytical-operation. It is operator that produces one third function through function-f and function-g, which describes the acreage of the overlap between the function-f and the flipped or translated function-g. The calculation-method is by Formula 1:

$$z(t) \overset{def}{=} f(t)^{*}g(t) = \sum_{r=-\infty}^{\infty} f(\tau)g(t-\tau) \tag{1}$$

The form of integration is:

$$\begin{aligned} z(t) = f(t)^{*}g(t) &= \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau \\ &= \int_{-\infty}^{\infty} f(t-\tau)g(\tau)d\tau \end{aligned} \tag{2}$$

Digital-image can be thought of as discrete-function of a two-dimensional-space, denoted like $f(x, y)$. Supposing a two-dimensional convolution-function $g(x, y)$, an output-image $z(x, y)$ is generated, which can be expressed by Formula 3:

$$z(x, y) = f(x, y)^{*}g(x, y) \tag{3}$$

Correspondingly, the kernel (in the convolutional neural network, called the "convolution kernel") is defined as a computational parameter in the learning theory algorithm. Then, when a two-dimensional image is used as an input, the corresponding convolution-operation can be shown by Formula 4:

$$z(x, y) = f(x, y)^{*}g(x, y) = \sum_{t}\sum_{h} f(t, h)g(x-t, y-h) \tag{4}$$

The form of integration is:

$$z(x, y) = (f^{*}g)(x, y) = \iint f(t, h)g(x-t, y-h)dtdh \tag{5}$$

Given a convolution-kernel of size m * n, there are:

$$z(x, y) = f(x, y)^{*}g(x, y) = \sum_{t=0}^{t=m}\sum_{h=0}^{h=n} f(t, h)g(x-t, y-h) \tag{6}$$

where f is the input image, g is the convolution kernel, m and n are the size of the kernel.

In the calculation, the convolution-kernel is multiplied by each n*n image area of the image, which is tantamount to extracting the n*n image area and representing it as a column-vector of length $n^{*}n$. In a zero padding, step-by-step sliding operation, a total of $(M-n+1)^{*}(M-n+1)$ calculation-results can get; while these image-regions are all represented as $n^{*}n$ column vectors, the original-image can be shown by $[n^{*}n^{*}(M-n+1)^{*}(M-n+1)]$. Without loss of generality, supposing that the number of convolution-kernels is K, the output got by the above convolution-operation is $k^{*}(M-n+1)^{*}(M-n+1)$, that is, the output is: the number of convolution-kernels * the image-width after convolution * the image-length after convolution. Its detailed illustration is shown in Figure 2.

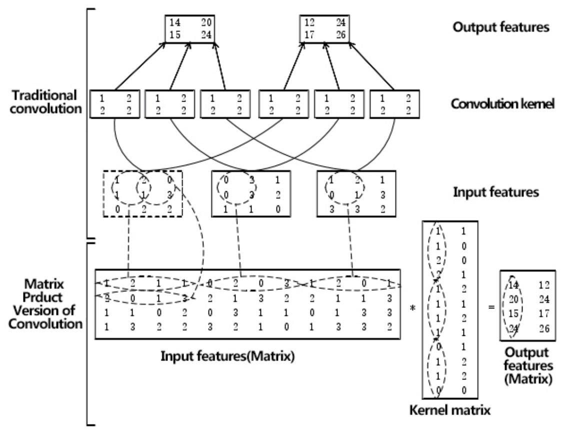

FIGURE 2. Schematic diagram of convolution operation.

1) CONVOLUTION-LAYER

The convolutional layer of CNN extracts different-features of the input through a convolution-operation, and the layer 1 convolutional layer extracts low-level-features such as edges and higher-level convolution-layers to get more advanced-features. To better know CNN, the one-dimensional-CNN (1DCNN) is taken as an example, and two-dimensional. Figure 3 shows the structure of the convolutional-layer and the pooled-layer of the one-dimensional-CNN.

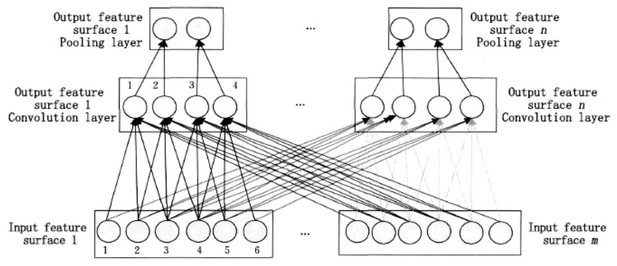
Output feature surface 1 Pooling layer
Output feature surface n Pooling layer
Output feature surface 1 Convolution layer
Output feature surface n Convolution layer
Input feature surface 1
Input feature surface m

FIGURE 3. Schematic-diagram of convolutional-layer and pooled-layer structure.

From Figure 3, the weight sharing occurs in the same color, and the different color weights are not shared. Weight sharing can reduce model complexity and make the network easier to train. Taking the output feature surface 1 of the convolutional layer in Figure 3 and the input characteristic surface 1 of its input layer as an example, $w_{1(1)1(1)} = w_{1(2)1(2)} = w_{1(3)1(3)} = w_{1(4)1(4)}$,and $w_{1(1)1(1)} \neq w_{1(2)1(1)} \neq w_{1(3)1(1)}$, where $w_{m(i)n(j)}$ represents the connection weight of the i-th neuron of the input-feature plane m and the j-th neuron of the output-feature plane n. In addition, the sliding-step size of the convolution-kernel, that is, the distance of each translation of the convolution kernel, is also an important parameter in the convolutional layer. In Figure 3, the convolution kernel is set to have a sliding step size of 1 in the upper layer and a convolution kernel size of 1 × 3. The size of each output-feature surface (ie, the number of neurons) oMapN of each convolutional layer in the CNN satisfies the following relationship:

$$oMapN = \left(\frac{(iMapN - CWindow)}{CInterval} + 1\right) \tag{7}$$

wherein iMapN represents the size of each input-feature surface; CWindow is the size of the convolution-kernel. In general, it is necessary to ensure that Formula (7) can be divisible, otherwise additional processing of the CNN network structure is required. The number of training parameters per convolutional layer CParams satisfies Formula (8).

$$CParams = (iMap \times CWindow + 1) \times oMap \tag{8}$$

wherein oMap is the number of output-feature faces for each convolutional-layer; iMap is the number of input-feature faces. 1 indicates the offset, and the offset is also shared in the same output-feature plane. Assume that the output-value of the k-th neuron of the output-feature surface n in the convolutional layer is x_{nk}^{out}, and x_{mh}^{in} represents the output value of the h-th neuron of the input feature surface m, as shown in Figure 3,then

$$x_{nk}^{out} = f_{cov}(x_{1h}^{in} \times w_{1(h)n(k)} + x_{1(h+1)}^{in} \times w_{1(h+1)n(k)} + x_{1(h+2)}^{in} \times w_{1(h+2)} \times w_{1(h+2)n(k)} + \cdots + b_n) \tag{9}$$

In the Formula (9), bn is an offset value of the output characteristic surface n. $f_{cov}(\cdot)$ is a nonlinear excitation-function. Jarrett *et al.* [14] explored different corrective nonlinearities (including max (0,x) nonlinear functions) in convolutional-networks, and found that they can increasingly improve the performance of convolutional-networks [15]. The calculation-formula for the ReLU function is as follows:

$$f_{cov}(x) = \max(0, x) \tag{10}$$

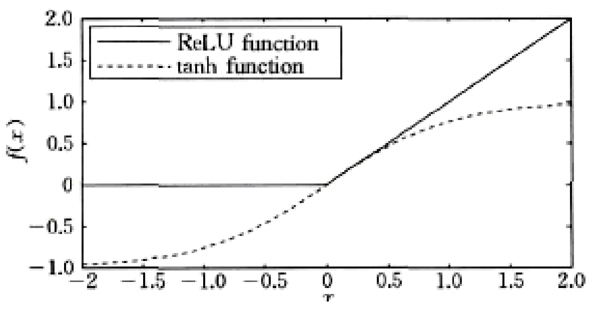

FIGURE 4. ReLU and tanh function graph.

In Figure 4, the solid line is the ReLU curve and the dotted line is the tanh curve. For ReLU, if the input is greater than 0, the output is equal to the input, otherwise the output is 0.

As can be seen from Figure 4, using the ReLU-function [16], the output doesn't tend to saturate as the input gradually increases. In his report, Chen analyzed three factors that affect CNN performance: number of layers, number of feature-faces, and network organization. The report uses nine structured CNNs for Chinese handwriting recognition experiments. Some statistical conclusions are obtained from statistical results of CNN structures with smaller convolution kernels: (1) Increasing the depth of the network can improve the accuracy; (2) the accuracy can be improved by increasing the number of element planes; (3) in terms of obtaining higher accuracy, the method of adding a convolutional layer is more effective than adding a fully connected layer. Bengio *et al.* [17] pointed out that: (1) The reuse of functions is better; (2) More features can be obtained from abstract expressions. Since more abstract concepts can be constructed from concepts that are weaker in abstraction, deep structures can capture more abstract expressions. For example, in CNN, this abstraction is established through pooling operations, and the more abstract concepts are usually invariant to most of the local changes in the input. Kai-Ming and Jian [18] explored how to balance the depth, number of features, and size of the convolution-kernel in the CNN network-structure in terms of computational complexity and time. This paper firstly studies the relationship between depth and convolution kernel size, replaces the larger convolution kernel with a smaller convolution kernel, and increases the network depth to increase the complexity. When the time complexity is approximately the same, a CNN structure with a smaller convolution-kernel and a deeper depth can obtain better experimental results than a CNN structure with a larger convolution-kernel and a shallower depth. Secondly, the paper also studies the relationship between the network-depth and the number of feature-planes. The CNN network structure is set to reduce the number of feature planes when increasing the network-depth, while the size of the convolution-kernel remains unchanged. The experimental results show that the deeper the depth, the better the performance of the network. However, as the depth increases, the network performance gradually becomes saturated. In addition, the paper also studies the relationship between the number of feature-planes and the size of the convolution-kernel through the fixed network-depth. Through experimental comparison, it is found that the number of feature-planes and the size of the convolution-kernel are similar, and the role of the network is not large.

2) POOLING LAYER

The pooling-method has the largest pooling, that is, the point with the largest value in the local accepting-domain, and the mean-pooling, that is, the mean-value of all the values in the local accepting-domain, and the random pooling. Boureau *et al.* [19] gave a detailed theoretical-analysis of the maximum-pooling and mean-pooling, and obtained the following predictions through analysis: (1) Pooling-max is quite good for separating very sparse-features; (2) It's good to perform pooling operations using all of the sampling-points in the local-area. For example, the mean pooling utilizes all sampling points in the locally accepted domain.The pooling method ensures that the neurons in the feature plane that are not the maximum excitation can also be utilized. In addition, there are pooling methods such as hybrid pooling, space pyramid pooling, and spectrum pooling. The so-called overlapping pooling method is that there is an overlapping area between adjacent pooling windows. Krizhevsky *et al.* [20] used the overlapping pooling framework to reduce the error rates of top-1 and top-5 by 0.4% and 0.3%, respectively. Compared with the non-overlapping pooling-framework, its generalization ability is stronger and it is more difficult to produce over-fitting. Assume that the output value of the l-th neuron on the n-th output feature-surface in the pooling-layer is t_{nl}^{out}. Similarly, in Figure 3, there are:

$$t_{nl}^{out} = f_{sub}(t_{nq}^{in}, t_{n(q+1)}^{in}) \tag{11}$$

wherein t_{nq}^{in} represents the output-value of the q-th neuron of the n-th input feature plane of the pooling layer; $f_{sub}(\cdot)$ can be a maximum-function, an average function, etc. The window in which the pooling-layer slides on the upper layer is also called the pooling core.The size of every output feature-surface (the number of neurons) of each pooled layer in CNN is DoMapN:

$$DoMapN = \left(\frac{oMapN}{DWindow}\right) \tag{12}$$

The size of the pooled core is DWindow and DWindow = 2. The pooling layer reduces the amount of computation of the network-model by reducing the number of connections between the convolutional-layers, that is, reducing the number of neurons by pooling-operations.

3) FULLY CONNECTED-LAYER

The fully connected-layer appears at the end of the network-structure and is a traditional multilayer-perceptron network. In Figure 5, each neuron of the fully connected-layer is fully connected to every neuron in the previous layer, which is also the source of the fully connected-layer name. The Softmax regression classification model is often used as the last layer

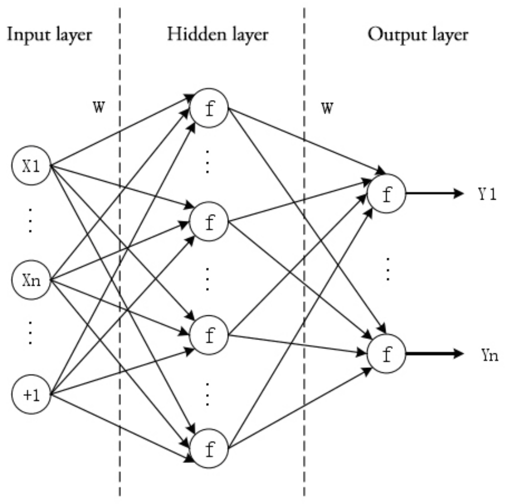

FIGURE 5. Conventional fully connected network structure.

of the fully connected-layer, with an output value of probability for each category between 0-1.

In the computer, the fully connected layer is equivalent to the inner product operation between the neural nodes, mainly involving the forward calculation and the backward calculation. The forward calculation uses Formula (13) to calculate the output-value of every neuron, while the backward calculation uses Formula (14) to calculate the error-term for every neuron.

$$y = W^T x + b \tag{13}$$

$$\frac{\partial \ell}{\partial x} = W^* \frac{\partial \ell}{\partial y}, \frac{\partial \ell}{\partial w} = x^* (\frac{\partial \ell}{\partial y})^T \tag{14}$$

where $y \in R^{m\times 1}$ represents the output of the neuron, $x \in R^{n\times 1}$ represents the input of the neuron, $W \in R^{n\times m}$ represents the weight of the neuron, b is the bias term, and ℓ is the neuron of the layer.

C. CONVOLUTIONAL NEURAL NETWORK TRAINING

The neural network can be trained by the gradient descent method through a sample training set {(x1,y1),…,(xm,ym)} containing m samples. For a single sample (x, y), its training cost-function is shown in Formula (15):

$$J(W, b; x, y) = \frac{1}{2} \parallel h_{W.b}(x) - y \parallel^2 \tag{15}$$

For a picture training set containing m samples, the overall cost-function is as shown in Formula (16):

$$\begin{aligned} J(W, b) &= \left[\frac{1}{2} \sum_{i=1}^{m} J(W, b; x^l, y^l) + \frac{\lambda}{2} \sum_{l=1}^{nl-1} \sum_{i=1}^{sl} \sum_{j=1}^{sl+1} (W_j^l)^2 \right] \\ &= \left[\frac{1}{2} \sum_{i=1}^{m} (\frac{1}{2} \parallel h_{w.b}(x^l) - y^l \parallel^2) \right] + \frac{\lambda}{2} \sum_{l=1}^{nl-1} \sum_{i=1}^{sl} \sum_{j=1}^{sl+1} (W_{ji}^{(l)})^2 \end{aligned} \tag{16}$$

The first one of the above formula $J(W, b)$ is the mean square error term, and the second one is the regularization term, which exists to reduce the weight magnitude and prevent the result from over-fitting. The role of λ is to assign the weight of these two items.

The goal of training is to minimize the value of the loss-function $J(W, b)$. During training, generally W and b randomly generate relatively small values as initial values. The gradient function is then used to find the optimal-solution for the objective function. Each iteration of the gradient-descent method updates the values of W and b as shown in Formula (17).

$$\begin{aligned} W_{ij}^l &= W_{ij}^l - \alpha \frac{\partial}{\partial W_{ij}^l} J(W, b) \\ b_i^l &= b_i^l - \alpha \frac{\partial}{\partial b_i^l} J(W, b) \end{aligned} \tag{17}$$

In the formula, α represents the speed of learning. The calculation method of the partial derivative in the formula is more complicated, and the backpropagation algorithm can calculate the partial derivative more effectively. The back propagation algorithm is analyzed below.

For the trained sample (x, y), the forward excitation is used to calculate the network excitation and the expected value of the output $h_{W,b}(x)$. Then for each node i of the first layer, its residual term δ_i^l is calculated, which represents how much the node contributes to the final output residual. The residual of the output node can be represented by the difference between the output-value of the network and the real value, and the intermediate node needs to be obtained by backpropagation.

For each output unit of i-th layer, the calculation method of the residual term is as shown in Formula (18).

$$\delta_i^{nl} = \frac{\partial}{\partial z_i^{nl}} \frac{1}{2} \parallel h_{W,b}(x) - y \parallel^2 = -(y_i - a_i^{nl}) \cdot f'(z_i^{nl}) \tag{18}$$

For the first layer, the calculation method of each node residual term is as shown in Formula (19).

$$\delta_i^l = (\sum_{j=1}^{sl+1} W_{ji}^l \delta_j^{l+1}) f'(z_i^l) \tag{19}$$

From the above, the partial derivative calculation method of each item of the intermediate layer is as shown in the Formula (20).

$$\frac{\partial}{\partial W_{ij}^l} J(W, b; x, y) a_j^l \delta_i^{l+1} \quad \frac{\partial}{\partial b_i^l} J(W, b; x, y) = \delta_i^{l+1} \tag{20}$$

Through the above steps, it is possible to iterate through the gradient descent analysis, gradually reduce the value of $J(W, b)$, and then train the convolutional neural network.

D. TRAINING OPTIMIZATION ALGORITHM

After defining the entire network structure, the back-propagation algorithm is used to train the entire network parameters, and the convolutional neural network obtains the best prediction model by training the sample set. In order to measure the prediction effect, the objective function C is usually defined as the prediction result:

$$C = \frac{1}{2n} \sum_x ||y(x) - \alpha^L(x)||^2 \tag{21}$$

where y(x) is the label of the sample and $\alpha^L(x)$ is the resulting value of the output. When C is smaller, the training effect is better. In this process, the gradient-descent algorithm is often used to find the minimum parameter of the objective function C. The core idea of the gradient descent algorithm is to derive the partial derivative for each parameter, let the parameter change in the direction that makes C smaller, and iterate repeatedly until C reaches a minimum value. Assuming v is a vector of all the parameters in the network, the derivative of the objective function C is:

$$\nabla C = (\frac{\partial C}{\partial v1}, \frac{\partial C}{\partial v2}, \cdots, \frac{\partial C}{\partial vm}) \tag{22}$$

The parameter update method is:

$$v \to v' = v - \eta \nabla C \tag{23}$$

Among above, η is the learning rate. The objective function of the single sample x output is:

$$C_X = \frac{1}{2}||y(x) - \alpha^L(x)||^2 \tag{24}$$

$$C = \frac{\sum C_x}{n} \tag{25}$$

So,

$$\nabla C = \frac{1}{n}\sum \nabla C_x \tag{26}$$

Because each time the parameters are updated, it is necessary to train the entire training data set all at once, which is slow. Therefore, Stochastic Gradient Descent (SGD) algorithm can be used, taking every m data as a training batch (mini-batch), then:

$$\nabla C = \frac{1}{n}\sum \nabla C_x \approx \frac{1}{m}\sum_{j=1}^{m} \nabla C_{xj} \tag{27}$$

Then, the parameters are updated once according to the calculation result of each training batch, thereby obtaining:

$$W_k \rightarrow W_k' = W_k - \frac{\eta}{m}\sum_j \frac{\partial C_{xj}}{\partial w_k} \tag{28}$$

$$b_\ell \rightarrow b_\ell' = b_k - \frac{n}{m}\sum_j \frac{\partial C_{jx}}{\partial b_\ell} \tag{29}$$

The stochastic gradient-descent algorithm is widely used in the training process of neural networks, but it is often difficult to choose a suitable learning rate in practical problems, and it is easy to converge to local optimum. In some cases, it may be trapped at the saddle point, or even unable to converge, resulting in oscillations. In view of these problems, the following types of gradient descent optimization algorithms have emerged in the field of deep learning, which have better optimization effects in practice: (1) SGD-based gradient parameter optimization algorithm. In the algorithm, Qian [21] proposed adding momentum parameters to the stochastic gradient descent algorithm. The momentum is accumulated when the gradient points in the same direction, and decreases when the gradient direction changes, which can speed up network convergence and reduce shock. Nesterov accelerated gradient (NAG), based on the momentum method, calculates the gradient based on the future approximate position of the parameter, and provides the momentum method pre-update capability. (2) Adaptive learning rate update algorithm. In view of the difficulty in manually selecting the learning rate by the SGD algorithm, Duchi *et al.* [22] proposed an Adagrad gradient optimization algorithm based on parameter adaptive update learning rate, which updates the infrequently updated parameters and frequently changes. The parameters make minor adjustments and are suitable for processing sparse data. Zeiler *et al.* [23] proposed the Adadelta algorithm, which is an extension of the Adagrad algorithm. The efficiency of the Adagrad algorithm is improved by replacing the sum of all gradient squares in the Adagrad algorithm to limit the sum of past gradients to a fixed value. The RMSprop algorithm is an unpublished adaptive learning rate method mentioned by Geoffrey E. Hinton in the Coursera course. It uses an average of the exponential delay gradient squared to divide by the learning rate, solving the problem of the disappearance of the learning rate in the Adagrad algorithm. The Adaptive Moment Estimation (Adam) algorithm proposed by Kingma and Ba [24] is another adaptive update learning rate algorithm for calculating various parameters. It retains some first-order moments and second-order moments of past gradients similar to momentum, and also has good effects in practice. The Adamax method comes from the Adam algorithm and is a variant of the Adam algorithm based on infinite norms. Dozat [25] proposed the Adam algorithm in a report, which is essentially a combination of Adam and RMSprop algorithms with Nesterov momentum, and also has a good convergence effect in practice. (3) Optimization strategy based on SGD training process.to optimize the training process of the stochastic gradient descent algorithm and increase the application performance of the SGD algorithm, Ioffe and Szegedy [26] proposed the Batch Normalization method, which uses zero mean and variance pair parameters in each mini-batch iterative process. Normalization processing speeds up the learning of the model. Neelakantan *et al.* [27] proposed that adding gradient noise based on Gaussian distribution for each gradient update can make the network with poor initialization more robust and help to train complex deep networks. In addition, the Early Stopping strategy mentioned by Geoffrey E. Hinton in the speech suggests monitoring the error changes on the verification data set during the training-process. If the verification error remains the same or there is an increase, the training is stopped immediately. In practice, this method also saves the time cost of the training model to some extent.

III. EXPERIMENTS

This experiment is all done in MatlabR2014b environment, Windows7 version, 64-bit operating system, Intel Core CPU, clocked at 3.40GHz, and memory is 16GB.

Before using the convolutional-neural network, we must preprocess the image. The OCT images used in the experiments in this paper were obtained from clinical data manually labeled by doctors. A cardiovascular OCT image is shown in Figure 6(a). The yellow text and arrows in the figure indicate the imaging-catheter and guidewire during cardiovascular OCT-imaging, the white text and arrows indicate the vessel wall, and the red sector is the range of unstable plaques manually-labeled by the doctor. The original OCT image is transferred from the original Cartesian to the polar coordinate system, as shown in Figure 6(b), and the area where the unstable plaque is located is represented by the column number of the image. The images used for training and testing are polar maps with an image size of 720*352 (width*height) and a single-channel 8-bit grayscale
image.

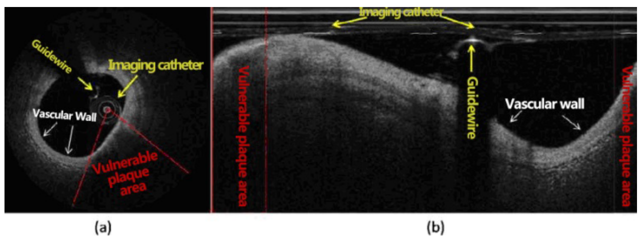

FIGURE 6. (a) Cardiovascular OCT-images in a Cartesian-coordinate system. (b) Cardiovascular OCT-images in polar-coordinates system.

In order to further extract the information part of the image, in the preprocessing, the pixel is cut out according to the boundary to a thickness of 100 pixels, the image size is 720*100 (width*height), and the part of the single channel 8-bit grayscale image is used as an input of the algorithm. In this experiment, 2000 pre-processed OCT images were selected, including 1000 positive samples containing vulnerable plaques and 1000 negative samples containing no vulnerable plaques. As shown in Figure 7:

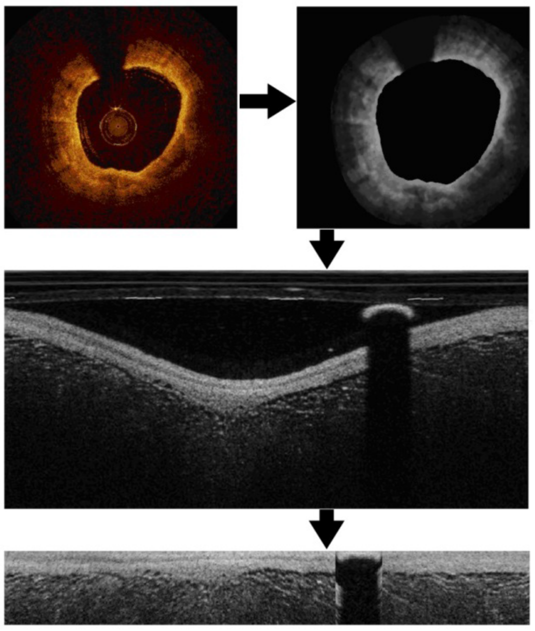

FIGURE 7. OCT image preprocessing and region extraction.

Divide the image into equally sized texture images, mark them, and manually sort the texture images. The number 0 indicates a non-vulnerable plaque texture image, and the number 1 indicates a vulnerable plaque texture image. A texture image with a sample size of 2000 training set image is selected as the training sample, wherein 1000 texture images are vulnerable plaques and 1000 texture images are non-vulnerable plaques. Then 200 test texture images were selected as test samples, of which 140 were vulnerable plaques and 60 were non-vulnerable plaques. The training samples and test samples are input into a convolutional neural network for training and testing.

IV. DISCUSSION

A. COMPARISON OF EXPERIMENTAL RESULTS BETWEEN OPTIMIZATION MODEL AND BASIC MODEL

Through the test model to verify the recognition of the picture set, the accuracy of the model is obtained. On the one hand, compared with the recognition accuracy of the basic model, the model is optimized or not. For the optimized test model, we set its weight learning rate base_lr to 0.001, the number of iterations to 1000, randomly select 200 images for verification, and divide into 5 groups of 40, and these pictures are different from the pictures used for training and testing. The trained models are used to classify and identify them, and the accuracy (Test-A) of the model is obtained by calculation. The accuracy of our definition of the model is expressed in terms of the number of sheets with the correct classification result/total number of verifications. The experimental results are shown in Table 1 below.

Observing the data in Table 1 can be known. After the measures of image data set expansion, network hierarchy adjustment and model parameter change, the test accuracy of the model is significantly improved compared with the basic model. The error loss function of the model converges to a very small value, and the network parameters of the model obtain the optimal solution, which indicates that the model can realize the recognition and classification of abnormal images. The results show that the accuracy of the model for image classification and recognition can reach about 90%, which is much higher than the experimental results of the basic model, and the model is optimized.

B. THE EFFECT OF DIFFERENT TRAINING OPTIMIZATION ALGORITHMS ON THE ACCURACY OF MODEL RECOGNITION

In the experiment, the correlation function is not changed, like the activation-function ReLU of the convolutional network, the size of the convolution-kernel is 3*3, and the dropout rate of 0.1, and so on. Only the training optimization algorithms in the network are set to: Adadelta, Adagrad, Adam, Nadam, Adamax and RMSprop, the experimental-results are shown in Table 2.

The experimental results of different optimization algorithms show that the convergence speed of other self-learning optimization algorithms is relatively fast except for the stochastic gradient descent SGD optimization algorithm. Comparing the accuracy, it can be seen that the RMSprop algorithm uses the ratio of the mean of the exponential delay gradient to the learning rate to update the parameters. The recognition accuracy of the vulnerable plaque recognition task with relatively simple sample data features is relatively low, indicating that a certain degree of over-fitting problem has appeared in the training of RMSprop algorithm. Comparing the over-fitting ratios, it can be seen that these self-learning gradient updating algorithms increase the gradient update parameters to achieve the learning rate self-learning purpose. In the vulnerable plaque recognition

TABLE 1. Comparison of experimental results between optimized CNN model and basic CNN model.

	Test	Loss
optimized CNN model	96.65%	0.02
basic CNN model	85.34%	0.76

TABLE 2. Experimental-results of different training optimization algorithms.

Training optimization algorithm	Average accuracy（%）	OverRadio	Average training time (seconds / trip)
Adadelta	96.23	1.0098	126
Adagrad	96.43	1.0048	131
Adam	96.15	1.0117	135
Nadam	96.46	1.0159	143
Adamax	96.47	1.0252	124
RMSprop	95.86	1.1280	121
SGD	96.65	1.0041	119

TABLE 3. Identification results.

	The number of vulnerable parts identified by the doctor's naked eye	Recognition rate	The number of vulnerable parts identified by the automatic identification eye	Recognition rate
Sample	116	87.89%	132	96.65%

task, the training parameters are added and the over-fitting problem has certain optimization control ability. However, the loss on the training-set and the test-set is larger than the SGD algorithm, and the robustness is relatively poor. In addition, due to the addition of the Nesterov momentum parameter, the Madam algorithm takes a lot of time to train. In general, the experimental results of these training optimization algorithms are not much different in the vulnerable plaque recognition task. In practice, different optimization algorithms can be selected according to different data types.

C. COMPARISON OF DOCTOR'S NAKED EYE RECOGNITION AND ALGORITHM AUTOMATIC RECOGNITION

The 200 OCT texture images to be tested are input into the constructed CNN network. According to the corresponding mark of the output, the texture image recognition result is obtained, and the identified vulnerable plaque is compared with the experienced doctor's naked eye recognition. Some results are shown in Table 3 below. The sample is randomly selected from 200 texture images to be tested.

After calculation, the automatic recognition rate of 200 images to be tested is 96.65%, while the visual recognition rate of experienced doctors is 87.89%. Experimental research can replace the doctor's manual discrimination of plaque with the naked eye, improving work efficiency.

V. CONCLUSION

At present, due to people's life, work styles and eating habits, more and more patients suffer from heart disease. Therefore, effective technical means are needed to screen for heart disease in persons who may be affected, and to diagnose and treat patients in time. With the widespread use of OCT technology in cardiovascular imaging, doctors diagnose cardiovascular disease based on the internal tissue structure of the coronary lumen reflected by the coronary vascular cross-sectional OCT image. Moreover, the automatic identification of the computer can quickly give objective judgments, make up for the many deficiencies of the doctor's manual identification, and has great significance for the diagnosis of the disease, the pathological analysis and the selection of the treatment plan.

The vulnerable plaque texture map and non-fragile plaque texture image data are used as training and test images. In this paper, the CNN theory is deeply explored, and the work is as follows: (1) The theory of convolutional-neural network is studied, and the backpropagation-algorithm is optimized. The algorithm involves a large number of mathematical-calculations and formula derivation, which provides a theoretical-basis for the subsequent research and design of network structure and the analysis of influencing factors. (2) By comparing the average accuracy and training time of several optimization algorithms, the influence of image database and network layer on network structure and model is analyzed. (3) Design an OCT coronary artery plaque image recognition network model based on convolutional-neural network, using the test accuracy of the model as the standard for evaluating the pros and cons of the model. Through experiment comparison, the influence of network parameters on model accuracy and error loss function is explored. (4) Optimize the CNN model to obtain an optimized CNN model for OCT vulnerable plaque image recognition to improve its test accuracy.

Here, the method of CNN is applied to the classification and recognition of OCT coronary lesion plaque images, which is of great significance for the automatic recognition and classification of images. However, due to my limited level and limited experimental conditions, there are still many shortcomings in the research work, such as: (1) The image database constructed is limited by its own energy and environment, and the number of data sets is relatively small compared to the current research needs. (2) For the design of the network structure, it is also necessary to adjust as the data set changes. The adjustment of the network parameters requires many experiments, and the test accuracy of the model is improved through comparative studies.

REFERENCES

[1] C. Nebauer, "Evaluation of convolutional neural networks for visual recognition," *IEEE Trans. Neural Netw.*, vol. 9, no. 4, pp. 685–696, Jul. 1998.

[2] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[3] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, Jun. 2005, pp. 886–893.

[4] Z. Guo, L. Zhang, and D. Zhang, "A completed modeling of local binary pattern operator for texture classification," *IEEE Trans. Image Process.*, vol. 19, no. 6, pp. 1657–1663, Jun. 2010.

[5] Y. B. Y. Lecun, "Convolutional networks for images, speech, and time series," in *The Handbook of Brain Theory and Neural Networks*. Cambridge, MA, USA: MIT Press, 1998.

[6] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313, no. 5786, p. 504, 2006.

[7] H. Sak, A. W. Senior, and F. Beaufays, "Long short-term memory recurrent neural network architectures for Large scale acoustic modeling," in *Proc. Comput. Sci.*, 2014, pp. 338–342.

[8] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 4, pp. 743–761, Apr. 2012.

[9] J. Schmidhuber, "Deep Learning in neural networks: An overview," *Neural Netw.*, vol. 61, pp. 85–117, Jan. 2015.

[10] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," in *Proc. Eur. Conf. Comput. Vis.*, in Lecture Notes in Computer Science, vol. 8689. Berlin, Germany: Springer, 2014, pp. 818–833.

[11] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," 2014, *arXiv:1409.1556*. [Online]. Available: https://arxiv.org/abs/1409.1556

[12] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2015, pp. 1–9.

[13] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2016, pp. 770–778.

[14] K. Jarrett, K. Kavukcuoglu, M. A. Ranzato, and Y. LeCun, "What is the best multi-stage architecture for object recognition?" in *Proc. IEEE 12th Int. Conf. Comput. Vis.*, Kyoto, Japan, Sep. 2009, pp. 2146–2153.

[15] V. Nair, G. E. Hinton, and C. Farabet, "Rectified linear units improve restricted Boltzmann machines," in *Proc. 27th Int. Conf. Mach. Learn.*, Haifa, Israel, 2010, pp. 807–814.

[16] K. O'Shea and R. Nash, "An introduction to convolutional neural networks," 2015, *arXiv:1511.08458*. [Online]. Available: https://arxiv.org/abs/1511.08458

[17] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 35, no. 8, pp. 1798–1828, Aug. 2013.

[18] K. He and J. Sun, "Convolutional neural networks at constrained time cost," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Boston, MA, USA, Jun. 2015, pp. 5353–5360.

[19] Y.-L. J. Y. Boureau, "A theoretical analysis offeature pooling in visual recognition," in *Proc. Int. Conf. Mach. Learn.*, 2010, vol. 32, no. 4, pp. 111–118.

[20] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1097–1105.

[21] N. Qian, "On the momentum term in gradient descent learning algorithms," *Neural Netw.*, vol. 12, no. 1, pp. 145–151, Jan. 1999.

[22] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *J. Mach. Learn. Res.*, vol. 12, no. 7, 2010, Art. no. 21212159.

[23] M. D. Zeiler, "ADADELTA: An adaptive learning rate method," 2012, *arXiv:1212.5701*. [Online]. Available: http://arxiv.org/abs/1212.5701

[24] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," 2014, *arXiv:1412.6980*. [Online]. Available: https://arxiv.org/abs/1412.6980

[25] T. Dozat, "Incorporating Nesterov momentum into adam," Stanford Univ., Stanford, CA, USA, Tech. Rep., Feb. 2016.

[26] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proc. 32rd Int. Conf. Mach. Learn.*, Lille, France, 2015, p. 37.

[27] A. Neelakantan, L. Vilnis, Q. V. Le, I. Sutskever, L. Kaiser, K. Kurach, and J. Martens, "Adding gradient noise improves learning for very deep networks," 2015, *arXiv:1511.06807*. [Online]. Available: https://arxiv.org/abs/1511.06807