Journal of Combinatorial Mathematics
and Combinatorial Computing
www.combinatorialpress.com/jcmcc

Combinatorial Press

Multi-level information extraction based on convolutional neural network and its optimal training method

Xinwen Chen[1, ✉]

[1] *College of Information Engineering, Ezhou Vocational University, Ezhou, Hubei, 436000, China*

ABSTRACT

This paper focuses on the characteristics of multilevel information extraction, based on the convolutional neural network model (CNN), introduces the multi-scale feature fusion and multilevel feature fusion strategy to study the multilevel information extraction method, and proposes the full convolutional neural network based on the attention mechanism and residual connection to form the multilevel information extraction model. Aiming at the gradient disappearance and saddle point problem of convolutional neural network, an activation gradient (AG) algorithm is proposed to optimize its training, which is improved to a class of activation gradient convolutional neural network (AG-CNN). The practical application effect of the multilevel information extraction model in this paper is verified by the information extraction work of net-pen culture in river-type reservoirs. Compared with the classical models such as UNet and ResUNet, the intersection and integration ratio (IoU), recall rate, precision rate, and F1 score of this paper's model reach the highest 80.28%, 91.02%, 87.18%, and 89.03% among all the models, which possesses a stronger extraction capability. And in the multilevel information extraction experiments on Cifar100 and Caltech256 datasets, when the number of batch training data is greater than 100, the accuracy rate and performance of the experimental group basically remain stable.

Keywords: convolutional neural network, multi-scale feature fusion, multilevel feature fusion, activation gradient algorithm, multilevel information extraction

1. Introduction

With the popularization and rapid development of the Internet, media data in different modalities such as images, texts and videos are growing rapidly. In daily life, people often use WeChat, Weibo, Baidu and other applications for daily information access, and these applications store various modal information, in the face of the huge multimedia data, how to quickly and accurately retrieve the specific modal information they need is of great practical significance [1-4]. In the past, most of the information

✉ Corresponding author.
E-mail address: cxw0012024@163.com (X. Chen).
Received 15 September 2024; Revised 05 November 2024; Accepted 20 February 2025; Published Online 16 April 2025.
DOI: 10.61091/jcmcc127b-039

retrieval techniques are based on a single modality, such as keyword-based text retrieval, image retrieval, etc [5-6]. With the rapid development of deep learning in recent years, people have achieved great success in the fields of image understanding and text understanding, and accordingly, the development of cross-modal graphic retrieval has been greatly facilitated, and the use of deep neural networks for feature extraction and feature alignment between cross-modal data can result in a retrieval model that is far superior to the traditional methods [7-9].

Deep learning represented by Artificial Neural Networks (ANN) has been widely used in the fields of information retrieval, video understanding, machine vision, speech recognition, and natural language understanding. Among them, convolutional neural networks have achieved unprecedented success in the field of computer vision. The main advantage of neural networks lies in their ability to automatically learn hierarchical features from data according to the task purpose, which tend to be more discriminative and a more essential portrayal of the target compared to traditionally designed features [10-13]. In addition, neural network models have very excellent transferability, and a small amount of data can accomplish the migration of the model to meet the needs of a specific application [14-16]. Convolutional neural network is a “specialization” of neural network, which refers to a neural network that uses at least one layer of convolutional operations in the network, and it is a kind of feed-forward hierarchical neural network, which plays an important role in the task of multilevel information extraction [17-20].

In this paper, the idea of using convolutional neural network for information extraction is taken as the research direction, effectively integrating and fusing multi-level and multi-scale features, and proposing a full convolutional neural network based on residual connection and attention mechanism to construct a multi-level information extraction model. More residual connections are added to overcome the problems of degradation and different semantic levels, and the contribution of different scale features is measured by the category-specified attention mechanism, which generates the category-specified weights of each scale feature with the help of the flexible maximum function on the scale dimension. The proposed full convolutional neural network is trained with sample data. Facing the common training problems during convolutional neural network training, such as vanishing problem, gradient explosion problem and saddle point problem, the activation gradient (AG) algorithm is proposed to improve the convolutional neural network model. Restricting the gradient in training to a limited range improves the stability of training and effectively alleviates and deals with the gradient explosion problem in training. Increase the gradient close to 0 to deal with the gradient vanishing problem. Adjust the parameter names of the AG algorithm to improve the saddle point problem and obtain a high-performance model. Carry out multilevel information extraction experiments, apply this paper's multiple information extraction modeling method to the information extraction work of net-pen aquaculture in river-type reservoirs, and test its practical application effect. Taking the TOP1 accuracy rate, TOP5 accuracy rate and loss function value as the evaluation index surface, the optimization training effect of the convolutional neural network of this paper is analyzed in terms of parameter tuning and generalizability.

2. Basic principles of convolutional neural networks

Generally speaking convolutional neural network (CNN) constitutes factors including convolutional layer, pooling layer, activation function layer, fully connected layer and normalization layer, etc., through the combination of the corresponding structure to extract image information, and finally calculate the loss function to achieve the classification effect [21].

1) Convolutional layer. The size of the feature output map of the convolutional layer depends on the size of the convolutional kernel, and is also affected by the step size parameter and whether or not to use edge filling and other operational parameters. In image processing, the convolution operation

multiplies each pixel in the input feature map with the corresponding element in the convolution kernel and then sums to obtain the value of the corresponding pixel in the output feature map.

In order to take into account the edge information of the input feature map in the convolution operation, usually adding zero values around the feature map can realize the edge filling operation and ensure that the edge information is fully calculated. Taking y_{11} and y_{12} as an example, the calculation process is shown in equation (1):

$$\begin{aligned} y_{11} &= x_{11} \times v_{11} + x_{12} \times v_{12} + \cdots + x_{32} \times v_{32} + x_{33} \times v_{33} \\ y_{12} &= x_{12} \times v_{11} + x_{13} \times v_{12} + \cdots + x_{33} \times v_{32} + x_{34} \times v_{33} \end{aligned} \tag{1}$$

2) Pooling layer. Pooling layer in DNN is a kind of downsampling method and has the effect of keeping the features unchanged, expanding the sensory field, and reducing the redundancy of information. Pooling operations can be generally divided into maximum pooling, average pooling, random pooling and other types. In the example, the size of the input feature map is 4×4, and the size of the output feature map is 2×2. In the figure, different colors are used to indicate the position of the corresponding receptive field in each pooling result, in which the maximum pooling takes the maximum value of the receptive field, and the tie pooling takes the average value of the receptive field. It can be seen that the parameters in the pooling layer do not need to be learned by the network, and the feature map consumes less computational resources after pooling.

3) Activation function layer. The convolutional layer and pooling layer provide a guarantee for linear computation of DNN, however, the stacking of linear convolutional operations is difficult to form a complex expression space, and it is also not easy to extract the semantic information in the image, therefore, a nonlinear activation function layer is introduced to improve the nonlinear mapping learning ability of the network, so as to improve the expression ability of the neural network.

The activation functions used in the activation function layer mainly include Sigmoid function (logistic sigmoid function), Tanh function (hyperbolic tangent function), ReLU (modified linear unit), Leaky ReLU (linear unit with leakage correction).

The Sigmoid function, also known as Logistic function, maps the features to $(0,1)$ intervals and the function is monotonically increasing, the closer to the zero point the greater the slope of the value, the greater the input data the output is closer to 1, the smaller the input data the output is closer to 0. The formula of the Sigmoid function is shown in Equation (2) [22]:

$$f(z) = \frac{1}{1+e^{-z}} \tag{2}$$

The Tanh function is proposed to solve the problem that the output of Sigmoid function is positive, which is more common than the former. Tanh function can be regarded as approximately linear in a certain region near the zero point, in addition, because the mean value of the Tanh function is zero, so it makes up for the problem that the Sigmoid function is non-zero. The larger the input data of Tanh function, the output will be close to 1, and the smaller the input data, the closer to 0. The calculation formula of Tanh function is shown in equation (3). The formula of Tanh function is shown in equation (3):

$$f(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}} \tag{3}$$

The Tanh function still has the disadvantage of extremely small gradient and slow weight update when the input value tends to infinity. In order to solve this gradient vanishing phenomenon, Nair et al. proposed the ReLU function. The ReLU function uses a linear function combined with a constant function to solve part of the gradient vanishing problem while avoiding the saturation problem in the value domain, which is calculated as in equation (4):

$$f(z)=\begin{cases} z & z>0 \\ 0 & z\le 0 \end{cases} \tag{4}$$

In order to further optimize the performance of ReLU function when the input value is negative, Leaky ReLU is proposed on its basis, when the input value takes a negative value, the output still exists non-zero gradient. Leaky ReLU continues the advantages of ReLU calculation is simple, to avoid the overfitting, and the convergence speed is faster compared to Sigmoid function, Tanh function, and its formula is as in Equation (5):

$$f(z)=\begin{cases} z & z>0 \\ \alpha z & z\le 0 \end{cases} \tag{5}$$

4) Normalization layer. In general, the normalization operation in neural networks is mainly divided into three steps, firstly, the mean μ and variance η^2 of the input data are found, and then the input distribution is normalized by equation (6), where X represents the input data, $\overline{X}$ represents the data obtained after X normalization, and ε is an arbitrary positive real number in order to ensure that the denominator is not zero:

$$\overline{X}=\frac{X-\mu}{\sqrt{\eta^2+\varepsilon}} \tag{6}$$

The normalization operation of Eq. (6) can change the real distribution of the original input data, and in order to further characterize the feature distribution of the input data, Eq. (7) is used to approximate the restoration of the original feature distribution state of the input data, where ξ is the scale transformation parameter and ζ is the offset transformation parameter.

$$Y=\xi\overline{X}+\zeta \tag{7}$$

The normalization layer can be divided into batch normalization (BN), instance normalization (IN), layer normalization (, LN), group normalization (GN), etc.

5) Fully connected layer

The fully connected layer plays the role of a classifier in a neural network, which is generally located in the last layers of the network to synthesize all the information. The fully connected layer connects each node to all the nodes in the previous layer, synthesizing the features extracted from the previous layers in this way, so that the network finally obtains the global features rather than the local features of a certain part of the network, but also due to the nature of the fully connected layer, resulting in an increase in the number of parameters.

For multilayer fully connected, the middle layer serves as both the input of the front layer and the output of the back layer. Take the two-layer fully connected as an example, the input and output are 4 nodes, the neurons are fully connected in pairs with the neurons in the two layers before and after them, but there is no connection between the neurons within the same fully connected layer, S_{2i} is the input and S_{3i} is the output as shown in Eq. (8):

$$S_{3i}=W\times S_{2i}+\theta \tag{8}$$

where W is the connection weight and θ is the bias.

3. Multi-level information extraction model based on convolutional neural network

In this chapter, based on the full convolutional neural network for information extraction, multi-scale feature fusion and multi-level feature fusion strategies are introduced, and a full convolutional neural

network based on the attention mechanism as well as residual connectivity is proposed for multi-level information extraction.

3.1. Multi-level and multi-scale feature fusion for fully convolutional neural networks

3.1.1. Multi-level feature fusion. Comprehensive use of different levels of features is the main means of existing full convolutional neural networks to realize multilevel feature extraction. At present, there are two main types of multi-level feature fusion methods for fully convolutional neural networks.

The first one is the "encoding and decoding" full convolutional neural network based on transposed convolution or up-sampling. This type of fully convolutional neural network, in the "encoding" process, uses continuous convolution and pooling superposition operation to extract features layer by layer, and then in the "decoding" process, from the extracted high-level features to gradually fuse the low-level features with higher resolution in order to recover the localization information. However, in this kind of network, the fusion of different levels of features is usually done by direct addition or direct connection, which ignores the potential semantic differences between the different levels of features and may adversely affect the accuracy of the extraction results. On the basis of this method, a fine up-sampling module combined with residual concatenation is proposed to complete the feature map sampling from coarse to fine.

The second type is to fuse multilevel features by additional methods or data. These types of data are not easy to obtain, so these methods using additional data are not suitable for most cases. In addition, the second type of methods cannot form an end-to-end training, and the powerful feature extraction ability of full convolutional neural network is difficult to be fully utilized, and the parameters of these low-level feature extraction methods are also difficult to be determined. Therefore, the methods proposed in this paper belong to the "encoding and decoding" fully convolutional neural networks that can be trained end-to-end.

3.1.2. Multi-scale feature fusion. It is important to introduce multi-scale feature fusion in full convolutional neural networks. Currently there are two main types of methods for fusing multiscale features in full convolutional neural networks.

The first one is to use pyramid pooling or expansion convolution to form a multi-scale expression in the last layer of the full convolutional network. The use of expansion convolution for multi-scale feature extraction ensures that the number of parameters in the model will not increase, and more scale information can be extracted by expanding the receptive field.

The second method of fusing multi-scale features is to obtain multi-scale features by multi-scale input. In this way, the multi-scale representation of the image is firstly constructed, then each scale representation is input into the neural network to obtain the features of this scale, and finally these features are fused. Compared with the first method, this method is more direct to obtain multiscale features and can reduce the number of parameters through the strategy of weight sharing, but the computational efficiency of this method is lower than that of the first method due to the fact that this method has to perform one operation for each input.

3.2. Full Convolutional Neural Networks with Joint Attention Mechanism and Residual Connectivity

Based on the above ideas and analysis, under the idea of utilizing multiscale inputs to obtain multiscale features, this section proposes a full convolutional neural network based on residual connectivity and attention mechanism, which integrates multilevel and multiscale feature fusion and can be trained end-to-end. The network (later referred to as mcResNet-csAM) mainly consists of a multi-residual connected full convolutional neural network (mcResNet) for fusing multilevel features and a category-specific attention mechanism model (csAM) for multiscale feature fusion [23]. The network first inputs multiscale images, then fuses different levels of features and extracts different scale features by mcResNet, and finally fuses these multiscale features using csAM and obtains the final result.

3.2.1. Multi-Residual Connected Fully Convolutional Neural Networks for Multi-Level Feature Fusion. mcResNet is a typical "encoding-decoding" structure that can be trained end-to-end, and its "encoding" structure is based on a 101-layer residual network. In order to obtain a dense feature map, this section removes the last average pooling and fully connected layers, and uses the inflated convolution instead of the normal convolution in the last two residual combinations and sets the step size to 1. As a result of these steps, the encoding process results in a feature map with a resolution equivalent to 1/8 of the input image. In the "decoding" process, a transposed residual unit based on transposed convolution is designed in this section to upsample the low-resolution feature map to a higher resolution.

Inspired by residual networks, this section adds more residual connections to overcome the above two difficulties. Specifically, a unit consisting of multiple convolutional layers has a mapping that can be denoted by $\mathrm{F}^{l}(\cdot)$, while the desired target mapping can be denoted by $\mathrm{H}^{l}(\cdot)$. Unlike a regular unit that learns mapping $\mathrm{H}^{l}(\cdot)$ directly, a residual unit learns a mapping that is a residual mapping, which can be represented by equation (9), the

$$\mathrm{F}^{l}(\cdot) := \mathrm{H}^{l}(\cdot) - f^{l-1} \tag{9}$$

where f^{l-1} denotes the feature entered into the cell. Therefore, the target mapping $\mathrm{H}^{l}(\cdot)$ can be replaced by $\mathrm{F}^{l}(\cdot) + f^{l-1}$.

3.2.2. Scale Attention Mechanism Model for Category Designation. Unlike the approach of multiscale fusion through direct concatenation or summation, this chapter measures the magnitude of the contribution of features at different scales through the category-specified attentional mechanism by generating a scale weight for each category at each pixel.

The weighted feature map and the categorized feature map are denoted by $g_{i,c}^{s}$ and $f_{i,c}^{s}$, respectively, where s and c denote the scale and the category, respectively, while S and C denote the number of scales and categories, respectively, and i denotes the feature with position i in the feature map. The category-specified weights $\alpha_c^s(x_i)$ can be calculated by equation (10):

$$\alpha_c^s(x_i) = \frac{\exp\left(g_c^s(x_i)\right)}{\sum_{s=1}^{S}\exp\left(g_c^s(x_i)\right)} \tag{10}$$

Finally, the categorical feature map $f_c(x_i)$ can be computed by equation (11), where $f_c^s(x_i)$ is the categorical feature map of the scale branch s.

$$f_c(x_i) = \sum_{s=1}^{S}\alpha_c^s(x_i)\cdot f_c^s(x_i) \tag{11}$$

3.2.3. Training and Inference of Networks. For full convolutional neural network classification at the pixel level, each pixel x_i is usually classified into the category with the maximum a posteriori probability, which can be represented by equation (12):

$$c_i^* = \arg\max_{c\in\{1,\ldots,C\}} P_c(x_i;\theta) \tag{12}$$

where c_i^* denotes the category with the highest posterior probability of pixel x_i, $P_c(x_i;\theta)$ denotes the probability of category c at x_i, and θ denotes the parameters of the model. For the model presented in this chapter, $P_c(x_i;\theta)$ can be computed using the flexible maximum function of Eq. (13):

$$P_c\left(x_i;\theta\right)=\frac{\exp f_c\left(x_i\right)}{\sum_{c=1}^{C}\exp f_c\left(x_i\right)} \tag{13}$$

The above 2 equations establish the mathematical relationship from the model output to the classification result, so the training of the full convolutional neural network model proposed in this chapter by sample data is essentially a problem of solving the optimal parameters of the model θ.

The method proposed in this chapter has multiple scale branches, each of which can calculate the error between the true and predicted values, so the errors of these branches are combined in the loss function as shown in equation (14) [24]:

$$\mathrm{L}_T=\mathrm{L}\left(P_c(x;\theta),y\right)+\frac{1}{S}\sum_{s=1}^{S}\mathrm{L}\left(P_c\left(x^s;\theta\right),y^s\right) \tag{14}$$

where x and y denote the input image with the corresponding true classification result, respectively; x^s and y^s denote the scale branching s the input image with the corresponding true classification result of that scale, respectively; and L a negative logarithmic function is used, as shown in equation (15):

$$\mathrm{L}\left(P_c\left(x;\theta\right),y\right)=-\frac{1}{N}\sum_{i=1}^{N}\sum_{c=1}^{C}y_i^{(c)}\log P_c\left(x;\theta\right) \tag{15}$$

where N denotes the number of pixels in the input image; $y_i^{(c)}$ denotes when $y_i=c$, $y_i^{(c)}=1$; otherwise $y_i^{(c)}=0$. The loss function also remains consistent across different scale branches.

4. Activation gradient-based optimized training of convolutional neural networks

This chapter is to propose a new treatment for the common training problems during the training of convolutional neural networks. Deep convolutional neural networks often suffer from performance degradation or even training failure due to the gradient vanishing problem, gradient explosion problem and saddle point problem. This chapter proposes an activation gradient (AG) algorithm to deal with these problems and challenges, and accordingly improves the common convolutional neural network model into a class of activation gradient convolutional neural networks.

4.1. AG Algorithm

For the gradient vanishing problem, the gradient explosion problem and the saddle point problem, existing strategies are unable to mitigate and deal with them efficiently and simultaneously. Since these problems are directly related to the gradient, we try to design a strategy to control the gradient to make the training process more efficient and stable. Therefore, the AG algorithm acting on the gradient is designed and embedded into the optimizer for application to the training of convolutional neural networks. The AG algorithm is defined as follows:

$$AG\left(\sigma\right)=\alpha\cdot\arctan\left(\beta\cdot\sigma\right) \tag{16}$$

where $\sigma\in\square^{\breve{n}}$ is a $\breve{n}$-dimensional vector; α and β are two variable parameters of the AG; and $\arctan\left(\cdot\right)$ acts on each element of σ. Note that the values of α and β can be adjusted to improve performance depending on the problems encountered while training the model. Specifically, α primarily controls the range of the AG, while β primarily affects the slope of the curve near the 0 region.

4.2. Gradient Explosion and Gradient Vanishing

Before the formal discussion, we give some notational definitions and basic settings. First, the optimizer used for training in this chapter is stochastic gradient descent with momentum (SGDM). We define some basic notation here: the set of real numbers is denoted by $\mathbb{R}$; $\mathbb{R}^+$ denotes a positive real number; the weight matrix of the network during training is $\mathrm{W} \in \mathbb{R}^{m \times n}$. Assume that there are κ samples in a training batch of a neural network $\left\{x^{(1)}, \cdots, x^{(\kappa)}\right\}$. The corresponding labels of the above training samples are $y^{(1)}, \cdots, y^{(\kappa)}$. Defined in this chapter L is the loss function. $\mathrm{M}\left(x^{(i)}; w^{(i)}\right)$ is the model output at each iteration, where $x^{(i)}$ is the input to the model; and $w^{(i)}$ is the weight vector. Thus the gradient g of the optimization process of the convolutional neural network can be obtained as

$$g = \nabla_w \sum_{i=1}^{\kappa} \mathrm{L}\left(\mathrm{M}\left(x^{(i)}; w^{(i)}\right), y^{(i)}\right) / \kappa \tag{17}$$

The gradient during training, either too large or too small, can affect the performance of the model. To deal with these problems, the AG algorithm acting on the gradient is proposed in this chapter to target these problems. It will effectively mitigate and deal with the effects of these problems, which in turn will improve the performance and generalization of the trained model. The formula for AG acting on the gradient is

$$\dot{g} = AG(g) \tag{18}$$

where $g \in \mathbb{R}^n$. In general, for a given arbitrary scalar $\vartheta \in \mathbb{R}$, there is $\arctan(\vartheta) \in (-\pi/2, \pi/2)$. Thus, $\dot{g}_n \in \left(-\alpha\pi/2, \alpha\pi/2\right)$ can be obtained, where the subscript n denotes the n th element of the vector.

Gradient explosion problem is a neural network optimization process in which the chain derivation rule leads to successive products, which makes the gradient very large. In case of gradient explosion problem, it can easily lead to training failure. Even if the original gradient is very large, the AG algorithm can limit the gradient in training to a limited range, which can improve the stability of training. Therefore, AG algorithm can effectively mitigate and deal with the gradient explosion problem in training.

Regarding the gradient vanishing problem, a gradient close to 0 will cause the neural network model parameters to be almost never updated again, which leads to the failure of model training. It is not difficult to see $\exists \alpha, \beta \in \mathbb{R}^+$ that when the gradient g_n in training approaches 0, it makes the following equation hold:

$$AG\left(g_n\right) > g_n \tag{19}$$

α controls the upper and lower bounds on the output values of the AG algorithm and β controls the slope of the AG curve. Thus, if g_n is close to 0, when considering the gradient vanishing problem, we focus on the effects of α and β on AG and provide the following theorem.

Suppose $\alpha \cdot \beta > 1$, then $\exists g^* > 0$ which makes $\forall g_n \in \left\{\left(0, g^*\right) \cup \left(-g^*, 0\right)\right\}$, has $\left|AG\left(g_n\right)\right| > \left|g_n\right|$, where g denotes the gradient vector and g_n is the n th element in g.

Prove that, defining $f\left(g_n\right) = AG\left(g_n\right) - g_n = \alpha \cdot \arctan\left(\beta \cdot g_n\right) - g_n$, its first and second order derivatives can be found separately:

$$f'\left(g_n\right) = \frac{\alpha\beta}{1 + \left(\beta g_n\right)^2} - 1 \tag{20}$$

$$f''\left(g_n\right)=\frac{-2\alpha\beta^3 g_n}{\left[1+\left(\beta g_n\right)^2\right]^2} \tag{21}$$

It is easy to see that as $\alpha\beta>1$ and g_n approach 0, then $f'\left(g_n\right)>0$ must hold, which proves that $f\left(g_n\right)$ is a monotonically increasing function. For g_n approaching 0^+, the limit of $f\left(g_n\right)$ can be computed as:

$$\lim_{g_n\to 0^+} f\left(g_n\right)=\lim_{g_n\to 0^+}\alpha\cdot\arctan\left(\beta\cdot g_n\right)-g_n>f(0)=0 \tag{22}$$

For g_n converging to $+\infty$, the limit of $f\left(g_n\right)$ can be computed as:

$$\lim_{g_n\to +\infty} f\left(g_n\right)=\lim_{g_n\to +\infty}\alpha\cdot\arctan\left(\beta\cdot g_n\right)-g_n=-\infty<0 \tag{23}$$

At this point, it is known that there exists a point in $g^*\in(0,+\infty)$ such that $f\left(g^*\right)=0$ holds: for g_n, there is $g_n\in(0,+\infty)$, which implies that $-f\left(g_n\right)$ is a convex function. Due to $f\left(g^*\right)=0$ and $f(0)=0$, there is for $\tau\in(0,1)$, according to the proposed Jensen's inequality:

$$\begin{aligned}-f\left(\tau\cdot 0+(1-\tau)g^*\right)&=-f\left((1-\tau)g^*\right)\\&<-\tau f(0)-(1-\tau)f\left(g^*\right)=0\end{aligned} \tag{24}$$

holds. It is easy to conclude that when $f\left(g_n\right)>0$, there is $g_n\in\left(0,g^*\right)$, which shows that the following equation holds:

$$f\left(g_n\right)=\alpha\cdot\arctan\left(\beta\cdot g_n\right)-g_n>0 \tag{25}$$

Similarly, $f\left(g_n\right)=\alpha\cdot\arctan\left(\beta\cdot g_n\right)-g_n\in\left(-g^*,0\right)$ holds for $g_n<0$. So far, it can be concluded that $\left|AG\left(g_n\right)\right|>\left|g_n\right|$, $\forall g_n\in\left\{\left(0,g^*\right)\cup\left(-g^*,0\right)\right\}$. The proof is complete.

4.3. Saddle point problems

In the training of convolutional neural networks, the saddle point problem is also a shackle that limits the performance of the model. The AG algorithm enables the model to get rid of the saddle points to some extent quickly during the training process. Since the first-order derivatives of both local minima and saddle points are 0, the optimizer can easily confuse local minima and saddle points, which will cause the neural network to sometimes fall into saddle points. However, the saddle points have second order derivatives less than 0 in at least one dimension, which gives the optimizer the potential to continue optimizing. The Hessian matrix that sets the loss function is denoted as H. The eigenvalues of H are defined as $\lambda_1,\lambda_2,\cdots,\lambda_\xi$, where the symbol ξ refers to the total dimensionality of the model parameters. The probability that an eigenvalue is greater than 0 is defined as p_t, where p_t is not close to 1. We can easily compute the probability of $p(\mathrm{H})$, i.e., the probability that H is a positive definite matrix with all eigenvalues greater than 0 as follows:

$$p(\mathrm{H})=p\left(\lambda_1\right)\cdot p\left(\lambda_2\right)\cdot\cdots\cdot\left(\lambda_\xi\right)=\left(p_t\right)^\xi \tag{26}$$

Thus, a sufficiently large ξ (typically, deep learning models contain high dimensional parameters, and ξ will be large) will result in $p(\mathrm{H})$ being close to 0. This suggests that there is a high

probability that the Hessian matrix corresponding to point γ is not positive definite, i.e., there is a high probability that point γ is a saddle point. The symbols d_i and d_j represent two different dimensions, $i, j \in [1, \xi]$. Point P is a saddle point that crosses P along the two curves c_i and c_j in the d_i and d_j directions in the loss surface. P is locally minimal (in the d_j dimension) for curve c_j, but locally maximal (in the d_i dimension) for curve c_i. Neural networks often fall into this situation during optimization, especially optimizers of gradient decay series. The reality is that the loss surface is high dimensional, which complicates the solution of the saddle point problem. The effectiveness of the AG algorithm is verified by the curve c_i and its gradient-activated c_i^* in Fig. Based on the definition of the gradient, the following equation can be obtained:

$$g_{c_i} = \frac{\partial l}{\partial w} \tag{27}$$

The gradient $\tilde{g}_{c_i}$ after AG activation can be calculated as follows:

$$\tilde{g}_{c_i} = AG\left(g_{c_i}\right) = \alpha \cdot \arctan\left(\beta \cdot g_{c_i}\right) = \alpha \cdot \arctan\left(\beta \cdot \frac{\partial l}{\partial w}\right) \tag{28}$$

Therefore, the above equation is processed to obtain an expression for curve c_i^*:

$$c_i^* = \int \tilde{g}_{c_i} dw = \alpha \int \arctan\left(\beta \cdot \frac{\partial l}{\partial w}\right) dw \tag{29}$$

$\exists g^* > 0$ and $\beta > 1$ such that $\forall g_{ori}^i \in \left\{\left(0, g^*\right) \cup \left(-g^*, 0\right)\right\}$, the SGDM embedded in the AG algorithm escapes faster than the original, where $i \in [k+1, k+\delta]$. i.e., $\left|w_{ori}^{k+\delta} - w_{ori}^k\right| < \left|w_{AG}^{k+\delta} - w_{AG}^k\right|$, where the subscript ori denotes the original parameter and the subscript AG denotes the parameter activated by the AG. The parameter near the saddle point is denoted as w and is considered after step k for a total of δ steps.

Proof. Considering a total of δ steps after step k, the basic form of gradient descent in the case of the original and embedded AG algorithms are respectively:

$$w_{ori}^{k+\delta} = w_{ori}^k - \eta \sum_{i=k}^{\delta} \hat{g}_{ori}^i \tag{30}$$

and:

$$w_{AG}^{k+\delta} = w_{AG}^k - \eta \sum_{i=k}^{\delta} \hat{g}_{AG}^i \tag{31}$$

where η is the learning rate, i denotes the ird step length, and $\hat{g}$ is the gradient. Since the gradient near the saddle point is close to 0, by Theorems 1, $\exists g^* > 0$ and $\beta > 1$, such that $\forall g_{ori}^i \in \left\{\left(0, g^*\right) \cup \left(-g^*, 0\right)\right\}$, exists:

$$\left|g_{ori}^i\right| < \left|g_{AG}^i\right| \tag{32}$$

Further there:

$$\left|\eta \sum_{i=k}^{\delta} \hat{g}_{ori}^i\right| < \left|\eta \sum_{i=k}^{\delta} \hat{g}_{AG}^i\right| \tag{33}$$

So:

$$\left|w_{ori}^{k+\delta} - w_{ori}^{k}\right| < \left|w_{AG}^{k+\delta} - w_{AG}^{k}\right| \tag{34}$$

Proof complete.

5. Multi-level information extraction experiments

In the above paper, the full convolutional neural network, which integrates multilevel and multiscale features, is proposed as a multilevel information extraction method to realize the efficient extraction of information. In this chapter, the hierarchical information extraction method of this paper will be applied to the information extraction of net-pen aquaculture in riverine reservoirs, and a multilevel information extraction experiment will be conducted to test the practical application of this paper's method.

In this experiment, the Shuikou reservoir area in the Minjiang River Basin of China was selected as the study area to carry out the multilevel information extraction study of net-pen culture in reservoirs. The GF-2 remote sensing images and their corresponding labeled data covering the net box culture area in Shuikou reservoir area of Minjiang River Basin in China with different time phases were selected to produce the dataset.

5.1. Comparative analysis of precision

Based on the produced dataset, classical models such as UNet, ResUNet, Deep LabV3+, TransUNet, HRNet and the multilevel information extraction model proposed in this paper were utilized for training respectively. In order to compare the extraction effect of different models more intuitively, the accuracy of each model is evaluated, and the evaluation results are shown in Table 1. As shown in Table 1, the intersection ratio (IoU), recall rate, precision rate and F1 score of this paper's model are 80.28%, 91.02%, 87.18% and 89.03%, respectively, which are the highest compared with the precision evaluation results of the other five classical models, proving that this paper's model has a strong extraction capability.

Table 1. Model comparison

Model	IoU(%)	Recall(%)	Precision(%)	F1(%)
UNet	78.19	89.76	86.02	87.76
ResUNet	78.83	90.97	85.67	88.26
Deep LabV3+	74.56	86.97	83.87	85.4
TransUNet	77.57	88.34	86.5	87.44
HRNet	72.19	86.01	81.85	83.7
Model of this article	80.28	91.02	87.18	89.03

5.2. Application analysis

1) Extraction application of net-pen aquaculture area in inland waters. The model in this paper is applied to Cotton Beach Reservoir in Fujian Province, China, as an application of the model in the extraction of net-pen aquaculture in inland waters. The acquired 2022 Cotton Beach Reservoir 2-view GF-2 remote sensing images are used to produce the dataset, and the dataset is augmented so that the model can be further fitted during the training process to achieve the best training effect. In addition, in order to highlight the extraction ability of this model in Cotton Beach Reservoir, the UNet model and ResUNet model are also added for comparison, and the test results of the three models after training are shown in Table 2. The IoU, recall, precision, and F1 score of this paper's model reach 82.54%, 90.74%, 90.25%, and 90.48%, respectively, which are better than the other two models, further

proving that this paper's multilevel information extraction model has a certain degree of generalization ability.

Table 2. Information extraction performance of model

Model	IoU(%)	Recall(%)	Precision(%)	F1(%)
UNet	80.09	89.37	88.42	88.94
ResUNet	80.17	91	87.92	89.12
Model of this article	82.54	90.74	90.25	90.48

2) Extraction and application in marine net-pen aquaculture area. In order to further highlight the applicability of the model in this paper, its extension is applied to the sea net box aquaculture area. The selected net box aquaculture area is located in Sansha Bay sea area of Ningde City, Fujian Province, China. In this paper, the 1-view GF-2 remote sensing images of some of the net box aquaculture sea areas in Sansha Bay, Ningde City, China, in 2019 are selected as the data source and the corresponding labels are drawn for model training and test dataset production. At the same time, consistent with the inland waters application comparison analysis, still using the UNet model, ResUNet model and this paper's model for comparison, the test accuracy comparison results are shown in Table 3. As can be seen from the table, the model has improved extraction precision compared to the UNet model and ResUNet model, and the IoU, recall, precision, and F1 score all exceed the 90% level.

Table 3. The performance of information extraction test

Model	IoU(%)	Recall(%)	Precision(%)	F1(%)
UNet	93.2	98.46	94.6	96.56
ResUNet	93.63	98.53	94.94	96.67
Model of this article	93.9	98.59	95.28	96.86

6. Analysis of optimized training results for convolutional neural networks

Based on the convolutional neural network, this paper integrates multi-level and multi-scale features, combines attention mechanism and residual connection, proposes a method for multi-level information extraction of fully convolutional neural network, constructs a multi-level information extraction model, and proposes an optimization training method corresponding to the AG algorithm for the performance degradation or even training failure of convolutional neural network due to gradient vanishing problem and saddle point problem. This chapter will focus on the optimization effect of the convolutional neural network optimization training method proposed in this paper.

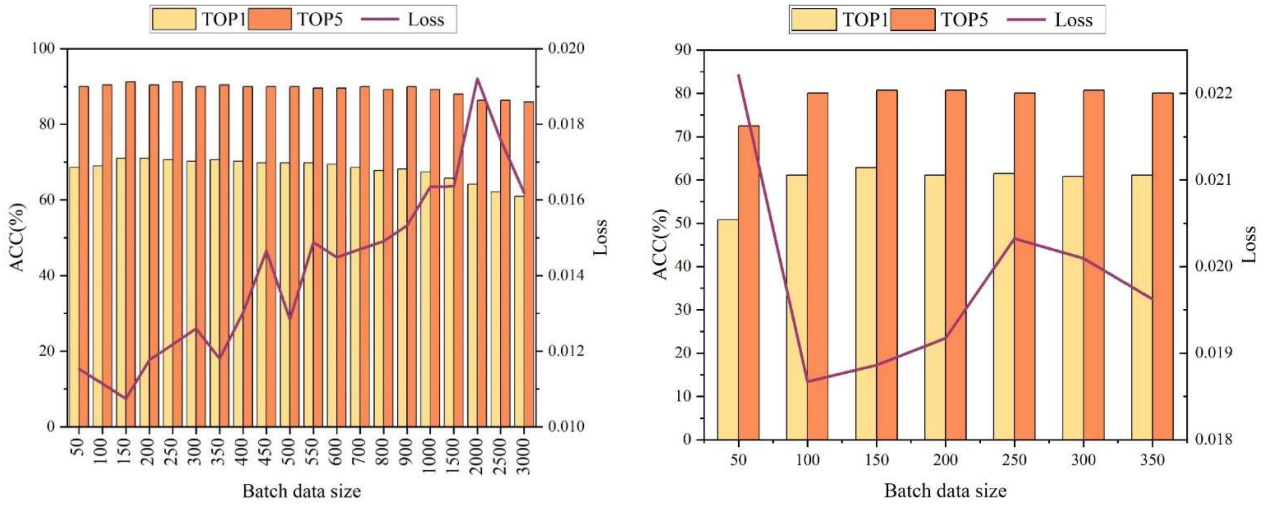
The experiments in this chapter use three metrics, TOP1 accuracy, TOP5 accuracy and loss function value, to evaluate the effectiveness of optimal training of convolutional neural networks.TOP1 is the accuracy at the maximum confidence level of the network's prediction, and then TOP5 is the accuracy at which the network predicts that the highest five confidence levels contain the correct category.

6.1. Parameter adjustment analysis

Different batch data volumes and data preprocessing steps may also have an impact on the final performance of the experimental groups.This section compares the experimental results of this paper's multilevel information extraction model trained using the optimizer embedded in the AG algorithm in the face of different batch data volumes and data preprocessing steps.

The multilevel information extraction experiments are conducted on Cifar100 dataset and Caltech256 dataset using the convolutional neural network of this paper. The effect of different batch data volume on the model of this paper is specifically shown in Fig. 1. Figures (a) and (b) show the

results on Cifar100 dataset and Caltech256 dataset, respectively. As can be seen from the figure, in the Cifar100 dataset, the accuracy of this paper's model performs poorly when the batch data volume is less than 100, and the performance of this paper's model remains stable as the batch data volume increases. When the batch data volume is greater than 1000, the performance of this paper's model is negatively correlated with the size of the batch data volume, and with the increase of the batch data volume, the TOP1, TOP5 accuracy of this paper's model gradually decreases. While the loss function value remains basically stable throughout the process. In the Caltech256 dataset, the performance of this paper's model is poor when the batch data volume is less than 100, and the performance is basically stable thereafter.



(a)Results of Cifar100 data set

(b)Results of Cifar256 data set

Fig. 1. Test result

The input data of the convolutional neural network has a significant impact on the final performance, so data preprocessing is an essential step before training. Experiments are conducted on Caltech256 dataset using the convolutional neural network proposed in this paper. At the very beginning, no data preprocessing step is used, only the input image size is unified as (224,224), and then random cropping, random horizontal flipping, random vertical flipping, and normalization preprocessing steps are added gradually. The impact of different data preprocessing processes is specifically shown in Table 4, which compares the performance of this paper's model in the face of a variety of different data preprocessing steps by constantly adding new data preprocessing steps. From the results in the table, it can be seen that different data preprocessing steps have different impacts on the performance of this paper's model, while the use of Beta constant learning rate significantly improves the performance of all this paper's model. Even in the face of the preprocessing steps that degrade the performance of the experimental group, the model in this paper still improves the performance.

Table 4. The influence of different data preprocessing process

Data enhancement	Constant learning rate			Beta constant learning rate		
	TOP1 (%)	TOP5 (%)	LOSS	TOP1 (%)	TOP5 (%)	LOSS
No data enhancement	49.6	75.34	0.0284	53.08	78.97	0.0294
+Random cutting	57.58	81.25	0.025	62.96	85.62	0.0208
+Random horizontal inversion	61.98	85.29	0.0219	67.93	89.82	0.0147
+Random vertical flip	59.56	84.46	0.0201	65.82	88.9	0.0138
+Normalization	59.97	84.99	0.0192	65.22	89.12	0.0122

6.2. Generalizability analysis

This section compares the experimental results of this paper's multilevel information extraction model trained using the optimizer embedded in the AG algorithm in the face of different batch data volumes and data preprocessing steps. This section experiments with different loss functions on TGS salt identification dataset. Here each pixel is divided into two categories by image segmentation with the goal of discriminating where in the geologic image salt deposits are contained. The training set in the dataset contains 5,000 images while the test set contains 20,000 images with a uniform size of (101,101). Experiments were conducted using convolutional neural networks with 300 rounds of training to compare the performance of this paper's model in three different loss functions, as shown in Table 5. Since it is only binary classification, i.e., whether individual pixels are predicted correctly or not, only accuracy (ACC) and loss (LOSS) are used as performance metrics in the table. The proposed algorithm achieves the maximum performance improvement in the experimental group with MSE loss function with an accuracy improvement of 7.4%, whereas the accuracy improvement is the least in BCE with Logits loss function with 2.89% and in Soft Margin with 3.37%. It can be seen that the model in this paper can be applied in a variety of loss functions, and the variation of the loss function also has an impact on the performance of the model in this paper.

Table 5. The performance of the model in different loss functions

Loss function	BCE with Logits		Soft Margin		MSE	
Learning rate algorithm	ACC (%)	LOSS	ACC (%)	LOSS	ACC (%)	LOSS
Constant learning rate	62.61	0.1137	62.9	0.1187	60.78	0.542
Beta constant learning rate	65.5	0.184	66.27	0.1704	68.18	0.8419

Overall, the multilevel information extraction model trained in this paper using the optimizer embedded in the AG algorithm, when optimized for the training of convolutional neural networks, performs better overall in deeper and more filtered network structures.

7. Conclusion

This paper utilizes convolutional neural network for information extraction, proposes a full convolutional neural network based on residual connection and attention mechanism, integrates multilevel and multiscale feature fusion, and constructs a multilevel information extraction model. The multilevel information extraction experiments are carried out, and the multilevel information extraction model of this paper is applied to the information extraction of net-pen aquaculture in riverine reservoirs to explore its practical utility. Comparing the classical models such as UNet, ResUNet, Deep LabV3+, TransUNet, HRNet, etc., the intersection and union ratio (IoU), recall, precision, and F1 score of this paper's model reaches 80.28%, 91.02%, 87.18%, and 89.03%, which are higher than those of other comparative models, and it has the highest precision of multilevel information extraction. In the specific extraction work of inland water net-pen aquaculture zones and marine net-pen aquaculture zones, the model in this paper also still maintains the highest IoU, recall, precision, and F1 scores.

Facing the phenomenon of convolutional neural network's performance degradation or even training failure due to gradient disappearance, gradient explosion and saddle point problem during the training process, this paper proposes activation gradient (AG) algorithm to deal with the problem and phenomenon, and improve the full convolutional neural network proposed in this paper into a class of convolutional neural network with activation gradient. The optimized training effect of the convolutional neural network in this paper is evaluated using three indexes: TOP1 accuracy, TOP5 accuracy and loss function value. In the face of parameter tuning, on the Cifar100 dataset and Caltech256 dataset, when the amount of batch data is less than 100, the TOP1, TOP5 accuracy of this

paper's convolutional neural network model performs not plus. While when the amount of batch data is greater than 100, the model performance is improved and basically remains stable. Compared with the performance of the proposed model in the three different loss functions of MSE, BCE with Logits and Soft Margin, the proposed model achieved the greatest performance improvement in the experimental group of MSE loss function, with an accuracy increase of 7.4%, and the accuracy of the experimental group of BCE with Logits and Soft Margin3 loss function increased by 2.89% and 3.37%, respectively.

Funding

This work was supported by Hubei Provincial Department of Education Class B Project (Project No: B2013003); Key Teaching and Research Project of Ezhou Vocational University (Project No: J2022ZD08).

References

[1] Li, W. H., Yang, S., Wang, Y., Song, D., & Li, X. Y. (2021). Multi-level similarity learning for image-text retrieval. Information Processing & Management, 58(1), 102432.

[2] Lan, H., & Zhang, P. (2021). Learning and integrating multi-level matching features for image-text retrieval. IEEE Signal Processing Letters, 29, 374-378.

[3] Bouchakwa, M., Ayadi, Y., & Amous, I. (2020). Multi-level diversification approach of semantic-based image retrieval results. Progress in Artificial Intelligence, 9(1), 1-30.

[4] Yang, L., Zhang, J., Guo, Y., & Wang, Q. (2017). Bayesian-based information extraction and aggregation approach for multilevel systems with multi-source data. Journal of Systems engineering and Electronics, 28(2), 385-400.

[5] Liu, Y., Wu, H., Huang, Z., Wang, H., Ma, J., Liu, Q., ... & Rui, K. (2020, November). Technical phrase extraction for patent mining: A multi-level approach. In 2020 IEEE International Conference on Data Mining (ICDM) (pp. 1142-1147). IEEE.

[6] Huddar, M. G., Sannakki, S. S., & Rajpurohit, V. S. (2020). Multi-level context extraction and attention-based contextual inter-modal fusion for multimodal sentiment analysis and emotion classification. International Journal of Multimedia Information Retrieval, 9(2), 103-112.

[7] Chathurani, N. W. U. D., Geva, S., Chandran, V., & Cynthujah, V. (2015, December). An effective content based image retrieval system based on global representation and multi-level searching. In 2015 IEEE 10th International Conference on Industrial and Information Systems (ICIIS) (pp. 158-163). IEEE.

[8] Barkalle, S. G., & Jain, P. (2018, May). An Effective Content Based Image Retrieval System Based on Global Representation and Multi-Level Searching. In 2018 2nd International Conference on Trends in Electronics and Informatics (ICOEI) (pp. 1329-1335). IEEE.

[9] Yang, Y., Wu, Z., Yang, Y., Lian, S., Guo, F., & Wang, Z. (2022). A survey of information extraction based on deep learning. Applied Sciences, 12(19), 9691.

[10] Liu, P., Zhang, H., Lian, W., & Zuo, W. (2019). Multi-level wavelet convolutional neural networks. IEEE Access, 7, 74973-74985.

[11] Chang, J., & Han, X. (2023). Multi-level context features extraction for named entity recognition. Computer Speech & Language, 77, 101412.

[12] Soleymani, S., Dabouei, A., Kazemi, H., Dawson, J., & Nasrabadi, N. M. (2018, August). Multi-level feature abstraction from convolutional neural networks for multimodal biometric identification. In 2018 24th International Conference on Pattern Recognition (ICPR) (pp. 3469-3476). IEEE.

[13] Zhang, H., Nie, R., Lin, M., Wu, R., Xian, G., Gong, X., ... & Luo, R. (2021). A deep learning based algorithm with multi-level feature extraction for automatic modulation recognition. Wireless Networks, 27(7), 4665-4676.

[14] Liu, J., Yang, Y., & He, H. (2020). Multi-level semantic representation enhancement network for relationship extraction. Neurocomputing, 403, 282-293.

[15] Xi, E. (2021, April). Image feature extraction and analysis algorithm based on multi-level neural network. In 2021 5th International Conference on Computing Methodologies and Communication (ICCMC) (pp. 1062-1065). IEEE.

[16] Rao, T., Li, X., Zhang, H., & Xu, M. (2019). Multi-level region-based convolutional neural network for image emotion classification. Neurocomputing, 333, 429-439.

[17] Nguyen, T. V., Liu, L., & Nguyen, K. (2016, November). Exploiting generic multi-level convolutional neural networks for scene understanding. In 2016 14th International Conference on Control, Automation, Robotics and Vision (ICARCV) (pp. 1-6). IEEE.

[18] Kumar, D., & Sharma, D. (2020, July). Multi-modal information extraction and fusion with convolutional neural networks. In 2020 international joint conference on neural networks (IJCNN) (pp. 1-9). IEEE.

[19] Wang, H., Qin, K., Zakari, R. Y., Lu, G., & Yin, J. (2022). Deep neural network-based relation extraction: an overview. Neural Computing and Applications, 1-21.

[20] He, Y., Li, Z., Yang, Q., Chen, Z., Liu, A., Zhao, L., & Zhou, X. (2020). End-to-end relation extraction based on bootstrapped multi-level distant supervision. World Wide Web, 23, 2933-2956.

[21] Tareq Jaouni,Francesco Di Colandrea,Lorenzo Amato,Filippo Cardano & Ebrahim Karimi. (2024). Process tomography of structured optical gates with convolutional neural networks. Machine Learning: Science and Technology(4),045071-045071.

[22] Li Peijun,Zha Yuanyuan,Zuo Bingxin & Zhang Yonggen. (2023). A Family of Soil Water Retention Models Based on Sigmoid Functions. Water Resources Research(3).

[23] Dang Lanxue,Pang Peidong & Lee Jay. (2020). Depth-Wise Separable Convolution Neural Network with Residual Connection for Hyperspectral Image Classification. Remote Sensing(20),3408-3408.

[24] Changsong Liu,Wei Zhang,Yanyan Liu,Yuming Li & Rudong Jing. (2024). Generating crisp boundaries using multi-scale features and mixed loss function.Applied Intelligence(prepublish),1-17.