CG Image Recognition Algorithm Based on Convolutional Neural Network

Chenyu Liu*,
University of Electronic Science and Technology of China, ChengDu, SiChuan, 610041, China
*Corresponding author’s e-mail: Lmomony@163.com

Abstract. Aiming at the problem that the existing computer generated images (CG) and real photograph (PG) have low recognition rates, this paper proposes an improved convolutional neural network method to achieve accurate recognition of the two photos. Firstly, a convolutional neural network classification model is established, then different models are established based on the VGG-19 network structure, and transfer learning is introduced to reduce training time and save computing resources. Finally, a softmax classifier is used for classification. The experimental results show that the accuracy of the proposed method for the recognition of PG images is about 93%. Compared with other methods, the method has the highest recognition accuracy, which indicates that the method is feasible and effective.

1. Introduction

With the development of computer graphics and other technologies, the fidelity of computer generated images (CG) has made it impossible for the naked eye to distinguish [1-2]. This is very beneficial to industries that are dedicated to virtual generation of real photograph, such as games and film and television industries, but it will also bring harm. For example, a filmmaker can modify an image to deceive everyone, which makes it a deceptive weapon. Therefore, it is of great significance to distinguish between a realistic generated image and a real photograph (PG). Some recent studies [3-4] have covered CG image and video detection. Among them, digital forensics methods have been proposed. Nevertheless, there is still much room for improvement in the technical capabilities of identifying CG images.

In this paper, an improved convolutional neural network method is proposed to realize the recognition of computer-generated images, and a transfer learning strategy is introduced during data training, which can reduce the feature dimension and improve the recognition accuracy. This paper models this problem of distinguishing between CG and PG images as a classification problem. For a given image, it is only necessary to predict a corresponding "CG" or "PG" label for this image.

2. Method of this paper

This paper uses the Caffe deep learning framework to implement the proposed method, and also performs data augmentation during training to improve the generalization ability of the model. Data augmentation includes two main operations: input image horizontal rotation and panning. Random translation includes horizontal and vertical dimensions.
2.1. Convolutional neural network framework

Inspired by the cat's visual cortical electrophysiology research, some researchers have proposed that the difference between convolutional neural networks (CNN) and ordinary neural networks is that convolutional neural networks include a feature extractor composed of a convolutional layer and a sub-sampling layer. A convolutional neural network consists of three parts: an input layer, a combination of \( n \) convolutional layers and pooling layers, and a fully-connected multilayer perceptron classifier [5-8].

CNN consists of an input layer, a convolutional layer, an activation function, a pooling layer, and a fully connected layer. The input image is mainly processed through the convolutional layer and the pooling layer. Its structure is shown in Figure 1.

![Convolutional neural network structure](image)

**Figure 1.** Convolutional neural network structure

The convolution layer is used to extract image features. In this layer, the convolutional neural network reduces the number of parameters in two unique ways: (1) local receptive field, (2) weight sharing. The more convolution layers, the stronger the ability to express features. The expression of the first convolution layer is

\[
x_j^l = f \left( \sum_{i \in M_j} x_{j-1}^i \times k_{ij}^l + b_j^l \right)
\]

Among them, \( x_j^l (j=1,2,\ldots,N) \) represents the feature map of the layer, \( M_j \) is the receptive field of the input layer, \( k_{ij}^l \) is the convolution kernel, \( b_j^l \) is the bias, and \( f(\cdot) \) is the activation function. The general activation functions are sigmoid function, tanh function, and RELU function. This article uses the RELU function as Activation function, the expression is

\[
h^{(i)} = \max(\omega^{(i)}x,0) = \begin{cases} \
\omega^{(i)}x, & x > 0 \\
0, & \text{else}
\end{cases}
\]

Among them, \( i \) represents the number of hidden units, \( \omega^{(i)} \) represents the weight.

The pooling layer is connected behind the convolution layer, and the object of the pooling layer is the local area of the feature map, which can make the feature have certain space invariance. The specific process is: for the adjacent feature map (FM) of the input image obtained from the convolution layer, the pooling technology is used to New features are obtained by FM down-sampling. In this step, parameters are reduced, feature dimensions are reduced, and features are space invariant. The most commonly used methods in pooling operation are stochastic pooling, max pooling and mean pooling.

Mean pooling is to average the feature points in the neighborhood; maxpooling is to maximize the feature points in the neighborhood; stochastical pooling is between the two. In this paper, the maximum pooling operation is adopted, in the form of

\[
x_j^l = f(\mu_j \max(x_{j-1}^i) + b_j^l)
\]

\[\text{(3)}\]
Where, $f(\cdot)$ is the activation function, $\mu^j_i$ is the weight coefficient, $P(\cdot)$ is the pooling operation, and $b^j_i$ is the bias. Finally, the output feature is used as the input of softmax regression in the output layer, and the CGS and PGs are identified by the loss function. For the convolution neural network architecture, this paper selects the vgg19 network architecture which has been successful in Imagenet as the basic network, and the overall model framework is shown in Figure 2. The main contribution of VGG19 network architecture is to adopt a very small 3 x 3 convolution core, and increase the depth of CNN network to 16~19 layers, greatly improving the existing performance.

2.2. Transfer learning

The convolutional neural network method needs a lot of training data, which leads to a long time for training, so migration learning has been concerned. This work shows that any deep neural network trained on natural images shows a strange phenomenon: the features learned from the first layer are similar to Gabor filters. In many types If the last layer is trained by a specific dataset, these features can be transformed from general to specific situations.

Migration learning is to use the knowledge learned before to help complete the learning tasks in the new environment. The reason why migration learning is introduced into convolutional neural network is based on the following reasons: at present, most successful models rely on a large number of labeled data, and many learning tasks are difficult to obtain a large number of labeled data For each task, training from scratch is very expensive. Migration learning definition: given the source domain and corresponding tasks, given the target domain and corresponding tasks. Migration learning is to use the knowledge in the given source domain and the given target domain to help learning the prediction function $f(\cdot)$ on the given target field when the given source domain is not equal to the given target domain or the task corresponding to the given source domain is not equal to the task corresponding to the given target domain.

For domain, it is defined as

$$D = \{x, P(x)\}$$

(4)
Where $\chi$ is the feature space and $P(X)$ is the marginal probability distribution, $X = (x_1, x_2, \cdots, x_n) \in \chi$. Learning task is defined as

$$T = \{y, f(\cdot)\} \quad (5)$$

Where $y$ is the label space, and $f(\cdot)$ means training through the training set $\{x_i, y_i\}$. When used to predict the labels of $X$, $f(\cdot)$ can be expressed as $P(y|X)$.

Transfer learning represents the possibility of transferring knowledge learned from one problem to another. Generally speaking, the transfer learning process includes the parameters of the (source) neural network. According to the specific task on a specific data set, this network is pre-trained into another (target) network. This target network has a different data set to solve. A different question. Transfer learning is simple and cost-effective. On the one hand, transfer learning saves a lot of training time and a lot of computing resources; on the other hand, if you start training from scratch, you need a lot of training images, but the data set images for some specific tasks are usually not Great. Transfer learning solves these two problems well. Therefore, this paper uses the pre-trained VGG-19 network for transfer learning.

The transfer learning strategy depends on many factors, but the two most important ones are the size of the data set and the similarity between the new data and the original data set. Keep in mind that the CNN features of the first few layers of the network are more generic, and more specific to the data set in the later layers. Fine-tune the weight of the pre-trained CNN through continuous backpropagation to achieve the best CNN performance.

### 2.3. Loss function

In this work, the problem of distinguishing between CGs and PGs is modeled as a classification problem. Softmax loss function is the most commonly used loss function in convolutional neural network-based methods. It is a generalization of binary logistic regression to multivariate. The form of Softmax loss function is shown in formula (6) or formula (7). Where $f_j$ represents the fraction vector of the element, and the maximum function is shown in formula (8).

$$L_q = -\log\left(\frac{e^{f_{y_q}}}{\sum_j e^{f_j}}\right) \quad (6)$$

$$L_q = -f_{y_i} + \log(\sum_j e^{f_j}) \quad (7)$$

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}} \quad (8)$$

The Softmax loss function is derived from the cross-entropy loss function and is used to evaluate the relationship between a distribution $p$ of a real class distribution $p$ and an estimated distribution $q$, as shown in formula (9).

$$H(p, q) = -\sum_x p(x) \log_q (x) \quad (9)$$

### 3. Experimental results and analysis

The method in this paper is based on experiments performed on two image databases. Database 1 is a self-built database. The database contains 2,000 computer-generated images and 2,000 real images. These images are compressed in the JPEG image format and the size ranges from 12kb to 1.8Mb. In the meantime, the image library content has a wide range of categories, including people, landscapes, animals, and architecture. 400 of the CG images come from the Columbia University CG library, and 1,600 from the CG websites at home and abroad; 1600 of the PG images come from the Columbia University PG library, and 400 from other PG image libraries. Data 2 is a DSTOK data set, including
4850 CG images and 4850 PG images. These images cover many fields such as automobiles, animals, and outdoor. They are compressed in the JPEG image format and the size is between 12kb and 1.8MB. Some images are shown in Figure 3. The images in the dataset are scaled to 224 × 224. The first line is a PG image, and the second line is a CG image.

Figure 3. Example of DSTOK data set

Use 50% cross-validation agreement to show the average accuracy of all experiments in this paper. In this paper, the initial learning rate is set to 0.01, and the attenuation coefficient is reduced by 0.1 times per 20,000 iterations. In this paper, the training batch size is set to 64 images per batch, and the weight attenuation coefficient is set to 0.0005. The optimization method is SGD (StochasticGradientDecent), and the momentum factor is set to 0.9.

The ratio of training samples to test samples in the experiment is 4:1. The last fully connected network layer will affect the training of the entire random initialization network. In order to reduce the impact, before the fine-tuning of the network, the last layer is pre-trained. The method of this paper is tested on the DSTOK data set. The experimental results are shown in Table 1.

| Fold | VGG19 (no migration) | VGG19 (migration) | VGG19 (migration + fine-tuning) |
|------|---------------------|-------------------|-------------------------------|
| 1    | 0.74                | 0.81              | 0.92                          |
| 2    | 0.67                | 0.77              | 0.91                          |
| 3    | 0.75                | 0.8               | 0.92                          |
| 4    | 0.69                | 0.72              | 0.93                          |
| 5    | 0.69                | 0.75              | 0.92                          |
| mean | 0.71                | 0.77              | 0.92                          |

From the data in Table 1, we know that the model trained using transfer learning and fine-tuning has achieved the best performance. The performance of the transfer learning model is also better than the model without the use of transfer learning, which can show that the transfer learning and fine-tuning strategies are effective and robust. Table 2 shows the comparison results between the model in this paper and the state-of-the-art methods on image database 1.
Table 2. Comparison of the results of this method and other methods

| Method | CG     | PG     | Average accuracy |
|--------|--------|--------|------------------|
| CON    | 0.902  | 0.854  | 0.878            |
| CUR    | 0.803  | 0.801  | 0.796            |
| GLC    | 0.637  | 0.626  | 0.612            |
| HOG    | 0.726  | 0.723  | 0.747            |
| HSC    | 0.804  | 0.793  | 0.731            |
| LBP    | 0.901  | 0.824  | 0.798            |
| Li     | 0.932  | 0.918  | 0.864            |
| LSB    | 0.656  | 0.624  | 0.925            |
| LYU    | 0.912  | 0.901  | 0.656            |
| POP    | 0.561  | 0.524  | 0.727            |
| SHE    | 0.734  | 0.683  | 0.572            |
| SOB    | 0.525  | 0.545  | 0.537            |
| VGG19  | 0.936  | 0.926  | 0.932            |

Table 2 shows that the model established in this paper is competitive among the most advanced models. On the self-built image data set, the average accuracy of this paper reaches 93%. These results prove that the transfer learning technology and fine-tuning strategy of this paper perform better than existing methods.

4. Conclusion
In order to solve the problem of low recognition rate between computer-generated images and real photos, this paper proposes a convolutional neural network method for fine-tuning transfer learning to implement computer-generated image detection. This method uses the VGG-19 network architecture and is trained on data. Introduce transfer learning strategies to save a lot of training time and a lot of computing resources, improve work efficiency and recognition accuracy. The experimental results show that compared with the existing methods on the self-built image database and DSTok data set, the model proposed in this paper has achieved better recognition accuracy, which shows the feasibility and effectiveness of the method in this paper.

References
[1] Wang, X.M., Gu, T.L., Luo, X.N., et al. (2019) A User Study on the Capability of Three Geo-Based Features in Analyzing and Locating Trajectories. IEEE Transactions on Intelligent Transportation Systems., 20: 3375–3385.
[2] Zhu, M.F., Chen, W., Xia, J.Z., et al. (2019) Location2vec: A Situation-Aware Representation for Visual Exploration of Urban Locations. IEEE Transactions on Intelligent Transportation Systems., 20: 3981–3990.
[3] Peng, F., Zhou, D.L., Long, M. (2017) Discriminating of Natural Images and Computer Generated Graphics Basing on Multi-Fractal and Regression Analysis. AEU-International Journal of Electronics and Communications., 71: 72–81.
[4] Tokuda, E., Pedrini, H., Rocha, A. (2013) Computer Generated Images vs. Digital Photographs: A Synergetic Feature and Classifier Combination Approach. Journal of Visual Commun and the Image Representation., 24: 1276–1292.
[5] Zhang, L., Liu, J., Zhang, B., et al. (2019) Deep Cascade Model-Based Face Recognition: When Deep-Layered Learning Meets Small Data. IEEE Transactions on Image Processing., 29: 1016–1029.
[6] Lu, J.W., Liong, V.E, Wang, G., et al. (2015) Joint Feature Learning for Face Recognition. IEEE Transactions on Information Forensics and Security., 10: 1317–1383.
[7] Gao, S.H., Zhang, Y.T., Jia, K., et al. (2015) Single Sample Face Recognition via Learning Deep Supervised Autoencoders. IEEE Transactions on Information Forensics and Security., 10: 2108–2118.
[8] Lu, J.W., Wang, G., Zhou, J. (2017) Simultaneous Feature and Dictionary Learning for Image Set Based Face Recognition. IEEE Transactions on Image Processing., 26: 4042–4054.