# An efficient network based on double constrained loss for fabric image retrieval

**Jiangsheng Gui** ( jsgui@zstu.edu.cn )
Zhejiang Sci-Tech University
**Dongwei Wu**
Zhejiang Sci-Tech University

**Research Article**

**Keywords:** Fabric image, Similarity loss, Image retrieval, MobileNet, Deep hashing

**Posted Date:** June 19th, 2023

**DOI:** https://doi.org/10.21203/rs.3.rs-3062181/v1



**Additional Declarations:** No competing interests reported.

**Version of Record:** A version of this preprint was published at Signal, Image and Video Processing on September 5th, 2023. See the published version at https://doi.org/10.1007/s11760-023-02749-y.

# An efficient network based on double constrained loss for fabric image retrieval

Gui Jiangsheng[1*] and Wu Dongwei[1†]

[1*]School of Computer Science and Technology, Zhejiang Sci-Tech University, Hangzhou, 310018, China, Country.

*Corresponding author(s). E-mail(s): jsgui@zstu.edu.cn;
Contributing authors: 1296754497@qq.com;
†These authors contributed equally to this work.

**Abstract**

In order to efficiently retrieve the same or similar fabric samples, a fabric image retrieval model based on deep hashing is proposed. The model is improved on the basis of MobileNetV1, and the performance of the model is improved by combining the h-swish activation function and attention mechanism module. The hashing method is used to solve the problem of low feature matching speed caused by high dimensional feature output. By combining the label information and similarity information of the images, a new loss function is constructed, which solves the problem of low model accuracy caused by the large feature difference between similar samples and the small feature difference between heterogeneous samples in fabric images. The experimental results on self-built fabric image dataset showed that the feature extraction time of the proposed algorithm was 0.25ms, and the MAP reached 93.2%, which can take into account the fabric retrieval speed and improve retrieval accuracy at the same time, and has certain application prospects.

**Keywords:** Fabric image, Similarity loss, Image retrieval, MobileNet, Deep hashing

## 1 Introduction

With the development of the industry and the improvement of people's living standards, consumers are no longer limited to the actual demand of commodities, but pursuing more attractive and diversified products. In order to attract more buyers, the textile industry usually prints various patterns on fabrics, so "multi-variety" is increasingly becoming a new production mode of textile industry. The enterprises have accumulated a large amount of historical production data under this production mode. However, in the process of imitation of new fabrics, it is usually necessary to manually analyze the technological parameters of the fabrics and find the same or similar fabrics from the warehouse or historical production records. Therefore, how to find the required fabrics from this huge historical data is a great concern of enterprises[1].

At present, the fabric retrieval method used in the industry is based on text (TBIR[2]). This method uses manual annotation and query, which is not only labor-consuming and time-consuming, but also has strong subjectivity, and it is difficult to meet the requirements of textile industry for the retrieval accuracy and speed of fabrics. Therefore, the content-based image retrieval (CBIR) method

has emerged. In the existing research, SHEN et al.[3] built a private dataset containing 972 kinds of fabric images, and used various deep convolution neural network models such as VGG to search, finally achieved an accuracy of over 90%. Abdul Haris Rangkuti et al.[4] used VGG19 as feature extractor, and used Manhattan, Euclid, Chebyshev and other distance measurement models to study 56 traditional styles, and the accuracy reached over 90% on the self-made fabric dataset. Silvester Tena et al.[5] built a private fabric image dataset containing a large number of complex patterns, and conducted experiments with deep convolution neural networks such as ResNet101 and InceptionV3 and finally the recall of top5 reached 84.08%. It is effective to use CNN for fabric retrieval, but the high-dimensional output characteristics of networks such as ResNet101 slow down the retrieval speed. Xiang J et al.[6] introduced soft similarity into small fabric datasets, and a CNN network is designed for fabric image representation. Zhang N et al.[7] used the approximate nearest neighbor search Annoy method to measure the similarity, and the aggregation convolution of different layers is used to prove the fabric image, by combining the color and texture features, good results have achieved on the self-made dataset. However, there are many kinds of fabric image datasets, and the difference between heterogeneous samples is small, so it is easy to cause low accuracy due to too small feature difference.

In order to solve the above problems, this paper combines CNN with hash coding, and uses the low-dimensional hash code as the feature representation of the image, so that the model can obtain the high-order semantic features of the image while having a low output dimension. At the same time, starting with the method of using tag information, a new loss function is proposed to narrow the feature distance between each pair of similar samples and expand the feature distance between each pair of different samples, and the retrieval accuracy is further improved by generating high-quality hash codes.

# 2 Related work

## 2.1 Overview of deep hashing

Deep hashing combines feature extraction with hash expression, so that the two processes can be carried out at the same time, thus effectively retaining the original feature information when the features are converted into hash codes. There are many kinds of deep hashing network models, such as the hash network model based on single input sample (mainly applying classification loss), the hash network model based on paired sample input (mainly applying similarity loss) and the hash network model based on triple (mainly applying triple loss).

(1) Hash network model based on single input sample: The deep hashing method of this model is improved directly on the image multi-classification model, such as adding a hash layer between the last two layers. A hash layer usually includes FC layer, activation function and thresholding layer. The input of the hash layer is the image features extracted from the previous layer, and its output is the input of the next fully connected layer. In the process of model training using classification loss such as cross entropy, the hash layer can also learn the mapping relationship from image features to hash codes, and the structure of the hash layer is shown in Fig.1.

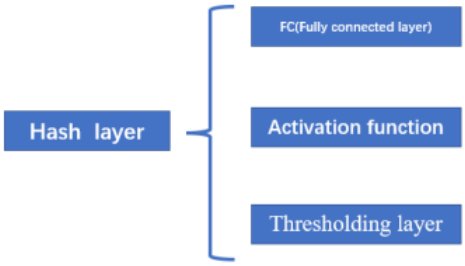


**Fig. 1** Hash layer structure

The thresholding layer can be constructed from this: when a sample enters the hash layer through CNN, the expected characteristic value should basically tend to 0 or 1, for example, the last input vector should be similar to the distribution of [0.1,0.9,0.8,0.005,0.995], then the vector can be simply converted into hash code at this time, for example, in the following two ways: 1. The average value is used as the threshold value. Firstly, the method accumulates all the elements in the output vector, and then takes the average value as the threshold. 2. A certain number as the threshold value. Since the values are distributed at both ends of 0 and 1, a number such as 0.5 can

be directly set as the threshold. In the thresholding layer, when the vector element value is greater than the threshold value, it is 1, otherwise it is 0, so that the input features are transformed into hash codes consisting of 0 and 1.

(2) Hash network model based on paired sample input: Different from single-input hash network model, this model is often constructed with two-way network structure, and the weights of two-way networks are shared during model training. The model trains two image samples as input at the same time, and then puts the obtained hash code into the loss function, and calculates the pairwise similarity loss according to whether the labels are the same or not.

(3) The hash network model based on triples: This model is similar to the hash network model input by paired samples, except that the two paths are changed into three paths, and the hash mapping relationship between the same class and different classes is learned at the same time. However, this method is limited by the selection of triples, therefore, there is no related research in this paper.

## 2.2 Network structure

In MobileNet series, MobileNetV1 proposed deep separable convolution, which greatly reduced the number of parameters of the model. MobileNetV2 proposed the inverse residual block based on V1 to improve the performance of model, while MobileNetV3 proposed h-swish[8] based on V2, which reduced the cost of calculating Sigmoid in swish and applied some SE (Squeeze-and-Exclusion)[9] modules. From the results in this paper, the performance of V3 is better than that of V1 and V2.

Compared with the structure of MobileNetV1, NMV (New MobileNetV1) proposed in this paper readjusts some structures, and adds h-swish activation function and SE mechanism module. The structure diagram of MobileNetV1 model is shown in Fig.2, and the structure diagram of NMV proposed in this paper is shown in Fig.3:

It can be found from Fig.3 that the improvement of NMV mainly includes the following points:

(1) Inspired by the innovative thought of Inception[10], since two 3×3 convolution kernels can replace one 5×5 convolution kernel in order to

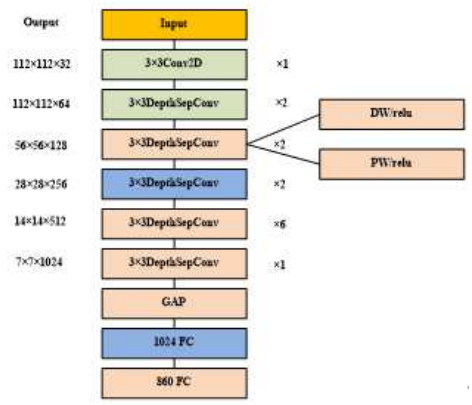


**Fig. 2** The structure diagram of MobileNetV1

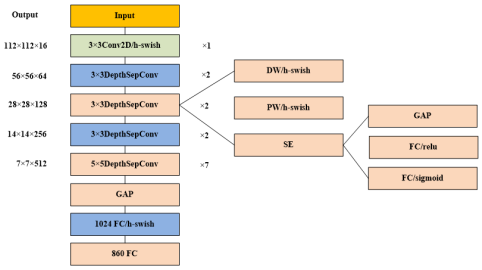


**Fig. 3** The structure diagram of NMV

reduce the parameters, one 5×5 convolution kernel can also replace two 3×3 convolution kernels in order to reduce the number of convolution layers. The reason why MobileNetV2 proposes the inverse residual block is that the gradient will disappear when V1 simply overlaps convolution layers, and if the convolution layers are too few, the feature extraction will not be sufficient. Therefore, a 5×5 convolution kernel is used in this paper.

(2) In order to get better results, h-swish proposed in MobileNetV3 is used instead of Relu activation function, and SE mechanism module is applied. The effectiveness of the two methods has been confirmed in related papers.

In order to promote implementation, this paper gives a detailed NMV model structure, as shown in Table 1:

In the NMV structure of Table 1, Conv2D uses standard 3×3 convolution. DepthSeqConv represents deep separable convolution, in which h-swish is applied after using deep convolution and point convolution. Use means that after the end of the current part, the SE mechanism module is added,

**Table 1** The structure of NMV

| Layer | Kernel Size | stride | Input | Output | SE |
|---|---|---|---|---|---|
| Conv2D | 3 × 3 | 2 | 224×224×3 | 112×112×16 | |
| DepthSepConv | 3 × 3 | 1 | 112×112×16 | 112×112×16 | |
| DepthSepConv | 3 × 3 | 2 | 112×112×16 | 56×56×64 | |
| DepthSepConv | 3 × 3 | 1 | 56×56×64 | 56×56×64 | |
| DepthSepConv | 3 × 3 | 2 | 56×56×64 | 28×28×128 | |
| DepthSepConv | 3 × 3 | 1 | 28×28×128 | 28×18×128 | |
| DepthSepConv | 3 × 3 | 2 | 28×28×128 | 14×14×256 | |
| 5×DepthSepConv | 5 × 5 | 1 | 14×14×256 | 14×14×256 | |
| DepthSepConv | 5 × 5 | 2 | 14×14×256 | 7×7×512 | Use |
| DepthSepConv | 5 × 5 | 1 | 7×7×512 | 7×7×512 | Use |
| GAP | 7 × 7 | 1 | 7×7×512 | 1×1×512 | |
| 1024FC/h-swish | 1 × 1 | 1 | 1×1×512 | 1×1×1024 | |
| 860FC/h-swish | | | 1×1×512 | 1×1×860 | |

and GAP means global average pooling. For the convenience of viewing, all layers above GAP in the network are aggregated and represented by Features layer, in which the structure of NMV is shown in Table 2.

**Table 2** The structure of NMV network

| Layer | Input | Output |
|---|---|---|
| Features | 3*224*224 | 512*7*7 |
| GAP | 512*7*7 | 512*1*1 |
| 1024FC/h-swish | 512*1*1 | 1024*1*1 |
| 860FC/h-swish | 1024*1*1 | 860*1*1 |

NMV - DH (New MobileNetV1-Deep Hashing) is a model that inserts the hash layer in a -DH way based on NMV, and all the models that insert the hash layer in this way in the experiment end with-DH. The improved NVM-DH model structure is shown in Table 3.

**Table 3** The structure of NMV network

| Layer | Input | Output |
|---|---|---|
| Features | 3*224*224 | 512*7*7 |
| GAP | 512*7*7 | 512*1*1 |
| 1024FC/h-swish | 512*1*1 | 1024*1*1 |
| 860FC/h-swish | 1024*1*1 | 48/64/128*1*1 |

In the improved NVM-DH model, the Hash1 layer uses the classification loss function to directly calculate the loss after the output of this layer, while the Hash2 layer uses the output of paired images to calculate the similarity loss. In the structure of NMV-DH network, the loss function used mainly includes classification loss and similarity loss.

# 3 Algorithm description

## 3.1 Construction of Classification loss function

A labeled image sample has class label information. The common method of using class label information is to regress the labeled feature vector, and directly regress its code to a matrix in the form of one-hot label, so as to directly calculate the regression classification loss, such as the NMV-DH structure above. At this time, the cross entropy loss is shown in Formula 1:

$$L_0 = -\sum_{i=1}^{N} \{q_i ln c_i + (1 - q_i) ln(1 - c_i)\} \quad (1)$$

In Formula 1, q_i and c_i are obtained by the following steps:

(1) Firstly, construct a 1024 orthogonal matrix Q=[q_1,. . . ,q_n ] (equivalent to one-hot matrix);

(2) At this time, the eigenvector matrix C=[c_1,. . . ,c_n ] of N samples has been obtained from the Hash1 layer, and the dimension of each c_i is 1024, where the label of each c_i corresponding sample is y_i;

(3) Then choose a q_i for each different y_i, and the q_i corresponding to the same y_i is the same. Therefore, every c_i labeled y_i can be matched to its corresponding q_i.

There is a problem with the constructed cross entropy loss function, that is, when the output coding features return to a matrix in the form of one-hot labels, the Hamming distance between the real labels corresponding to any two vectors is only 2. For example, vector A and vector B regress to [0,0,0,0,0,0,1,0] and [1, 0, 0, 0, 0, 0] respectively, which means that when they regress perfectly, their Hamming distance is only 2. It can easily cause the Hamming distance between the eigenvectors of classes to be too small, and it is difficult to normalize the eigenvectors of classes.

There is a problem with the constructed cross entropy loss function, that is, when the output coding features return to a matrix in the form of one-hot labels, the Hamming distance between the real labels corresponding to any two vectors is only 2. For example, vector A and vector B regress to [0,0,0,0,0,0,1,0] and [1, 0, 0, 0, 0, 0] respectively, which means that when they regress perfectly, their Hamming distance is only 2. It can easily cause the Hamming distance between the eigenvectors of classes to be too small, and it is difficult to normalize the eigenvectors of classes.

In order to solve the above problems, this paper proposes a loss function which uses Hadamard[11] matrix to expand the Hamming distance between classes, and calls it HL (Hadamard Loss), and applies it to Hash1 layer.

Hadamard matrix is composed of +1 and -1 elements, which is an orthogonal square matrix. The so-called orthogonal square matrix means that any two rows (or columns) are orthogonal, and the sum of squares of all elements in any row (column) is equal to the order of the square matrix, which is mainly composed of multi-order +1 and -1 elements, Hadamard matrix of order 2 can be expressed by formula 2: if it needs to be extended to a matrix of order 4, it is expressed by formula 3, and the general form is shown by formula 4, where A and K represent the order:

$$M_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad (2)$$

$$M_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \quad (3)$$

$$M_K = \begin{bmatrix} M_A & M_A \\ M_A & -M_A \end{bmatrix} \quad (4)$$

According to the properties of Hadamard matrix, it can be obtained that the inner product of any two column vectors (such as column vectors v(0)(1,1,1,1) and v(1)(1,-1,1,-1) in M_4) is 0, then the Hamming distance D(v(a),v(b)) of the two vectors (v(a),v(b)) is 1/2*(K-0)=K/2, K represents the matrix order, which is set to 1024 in the experiment. The following properties can be obtained:

Property 1: The Hamming distance of any two columns of the above matrix M_K is the same, and it is K/2.

Property 2: It can also be extended that the matrix [M_K,-M_K] also satisfies property 1.

The HL steps constructed in this paper are as follows:

(1) Firstly, get the eigenvectors c=[c_1,. . . ,c_n ] of N samples from the Hash1 layer, where the dimension of each c_i is K, and the label of each c_i corresponding sample is y_i;

(2) After that, construct Hadamard matrix with dimension K, get a matrix with [M_K,-M_K ]=[m_1,...,m_2K], and change all -1 to 0 , and choose a m_i for each different y_i, then the m_i corresponding to the same class is the same. m_i is used as the label information and c_i is used as the predicted feature, the corresponding content in Formula 1 is replaced, and the HL classification loss function L_0 is obtained as shown in Formula 5:

$$L_0 = -\sum_{i=1}^{N} \{m_i lnc_i + (1 - q_i) ln(1 - c_i)\} \quad (5)$$

## 3.2 Construction of similarity loss function

The similarity loss can be constructed for pairs of input images to ensure the similarity relationship between sample labels. A typical similarity loss or pairwise loss function can make similar images have similar hash codes (small Hamming distance), while different images have different hash codes (large Hamming distance). This paper mainly introduces two similarity loss functions.

(1) DSH[12]: This is a particularly classic similarity loss construction method. This method uses image pairs and corresponding similarity labels to train CNN, learns the class binary image representation that keeps similarity by carefully designing the loss function, and then quantifies the output of CNN to generate binary codes for new images. Its loss function design is as follows:

For a pair of images (I_(i,1),I_(i,2)), and the corresponding binary network outputs _(i,1),b_(i,2), if they are similar, define y=0, otherwise, define y=1. The loss definition of this pair of images is shown in Formula 6:

$$L = \frac{1}{2}(1-y)D(b_{i,1}-b_{i,2}) + \frac{1}{2}ymax(m - D(b_{i,1}-b_{i,2}), 0) \quad (6)$$

In formula 6,$D(\bullet)$represents the distance between b_(i,1) and b_(i,2), and m is the threshold parameter of the distance. Only when the distance is within the threshold, the diffenrent pairs will be considered as contributing to the loss function. In the references, its values are set to 1, 2, 3 and 6, and this paper uses the highest precision 2 as the parameter.

Euclidean distance is used when defining the distance. If the binary constraint is completely ignored, the difference between Euclidean space and Hamming space will lead to suboptimal binary code. The common scheme is to use Sigmoid or Tanh activation function to make the output approach the required value. However, using this nonlinear function will inevitably slow down or even limit the convergence of the network. In order to overcome this limitation, L1 norm is used to regularize the network output, and make it close to the discrete value (-1,1).

Then, N training pairs are selected from image pairs during the training, and L2 normal form is used for Euclidean distance. Combined with Formula 6, the overall loss DSH can be defined as follows, as shown by Formula 7:

$$\begin{aligned} L_1 = \sum_{i=1}^{N} \Big\{ & \frac{1}{2}(1-y) \left\| \mathbf{b}_{i,1} - \mathbf{b}_{i,2} \right\|_2^2 \\ & + \frac{1}{2} y_i \max\left( m - \left\| \mathbf{b}_{i,1} - \mathbf{b}_{i,2} \right\|_2^2, 0 \right) \\ & + \alpha \left( \left\| |\mathbf{b}_{i,1}| - \mathbf{1} \right\|_1 + \left\| |\mathbf{b}_{i,2}| - \mathbf{1} \right\|_h \right) \Big\} \end{aligned} \quad (7)$$

In formula 7, $\|\bullet\|_2$,$\|\bullet\|_1$,$|\bullet|$ are L2 norm of vector, L1 norm of vector, and absolute value operation of elements, respectively, and $\alpha$ is a weighting parameter to control regularization intensity. In references, its values are set to 0, 0.1, 0.01 and 0.001, in this paper, 0.1, the highest precision in the literature, is used as the parameter. In the selection of sample pair, the sample pair consists of a single sample and other samples because the total number of training samples is not large.

(2) CL (Cauchy Loss): It is also a loss function designed by using the similarity of image pairs. In this paper, another similarity loss function CL (Cauchy Loss) which uses Cauchy distribution to improve performance is proposed, and it is applied to Hash2 layer. The design is as follows:

First, given a set of trained image pairs, which are represented by $\{(x_i, x_j, s_{ij} : s_{ij} \in S\}$, then the generated hash code $H = [h1, h1, ..., h_N]$ for N samples in a set of images, its maximum posterior estimation can be expressed as $logP(H \mid S)$ .$logP(H \mid S)$ is proportional to $logP(S \mid H)P(H)$ by using Bayesian learning framework, where the likelihood function $logP(S \mid H)$ can be expressed as Formula 8, and Formula 9 can be obtained from

it.

$$P(S \mid H) = \prod_{sij \in S} [P(S_{ij} \mid h_i, h_j)]^{w_{ij}} \quad (8)$$

$$= \sum_{s_{ij} \in S}^{logP(S|H)P(H)} w_{ij} logP(S_{ij} \mid h_i, h_j) + \sum_{i=1}^{N} logP(h_i) \quad (9)$$

Where W is the weight used to represent the information between similar pairs of samples and dissimilar pairs, which is represented by Formula 10:

$$W_{ij} = \begin{cases} \frac{|S|}{|S_1|}, s_{ij} = 1 \\ \frac{|S|}{|S_0|}, s_{ij} = 0 \end{cases} \quad (10)$$

$S_1 = \{s_{ij} \in S : sij = 1\}$ represents a set of similar pairs, $S_0 = \{s_{ij} \in S : sij = 0\}$ represents a set of dissimilar pairs, $P(S_{ij} \mid h_i, hj)$ is the conditional probability of a pair of hash codes $(h_i, hj)$ given by the similarity label $S_ij$, which is expressed by Formula 11:

$$P\left(s_{ij} \mid h_i, h_j\right) \begin{cases} \sigma\left(d\left(h_i, h_j\right)\right), s_{ij} = 1 \\ 1 - \sigma\left(d\left(h_i, h_j\right)\right), s_{ij} = 0 \end{cases} \quad (11)$$
$$= \sigma\left(d\left(h_i, h_j\right)\right)^{s_{ij}} \left(1 - \sigma\left(d\left(h_i, h_j\right)\right)\right)^{1 - s_{ij}}$$

$\sigma$ is a well-defined probability function, and the sigmod function is commonly used. In this paper, the modified Cauchy distribution function is used to define $\sigma$. The original Cauchy distribution is expressed by Formula 12. Where $x_0$ is the position parameter used to represent the peak value of the distribution, $\delta$ is the scale parameter in Cauchy distribution, and the modified formula is represented by 13, where $\gamma$ is used to control different Hamming distance radius, the value range of this value can be [2, 200]. In this paper, reference is made to related literature[13] to set it to 20, and the formula 14 is obtained.

$$f(x; x_0, \delta) = \frac{1}{\Pi} \left[ \frac{\delta}{(x - x_0)^2 + \delta^2} \right] \quad (12)$$

$$\sigma(d(h_i.h_j)) = \frac{\gamma}{\gamma + d(h_i, h_j)} \quad (13)$$

$$P(h_i) = \frac{\gamma}{\gamma + d(|h_i, 1|)} \quad (14)$$

The value d($h_i$,$h_j$ ) of Hamming distance and the normalized Euclidean distance can be expressed by Formula 15:

$$d(h_i, h_j) = \frac{K}{4} \left\| \frac{h_i}{|h_i|} - \frac{h_j}{|h_j|} \right\|_2^2 = \frac{K}{2}(1 - cos(h_i, h_j)) \quad (15)$$

We want to maximize the likelihood estimation, that is, minimize the negative log-likelihood function. Then, in Hash2 layer, by substituting the above formula into Formula 9, the loss function CL that we finally want to minimize is expressed as L+Q, where L is shown by Formula 16 and Q is shown by Formula 17:

$$L = \sum_{s_{ij} \in S} w_{ij} (s_{ij} log \frac{d(h_i, h_j)}{\gamma} + log(1 + \frac{\gamma}{d(h_i, h_j)})) \quad (16)$$

$$Q = \sum_{i=1}^{N} log(1 + \frac{d(|h_i|, 1)}{\gamma}) \quad (17)$$

### 3.3 NMV-DH-HL-CL model

The model structure of NVM-DH-HL-CL combined with loss function HL+CL is shown in Fig. 4.

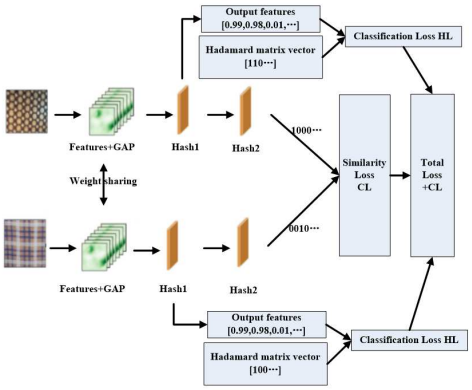


**Fig. 4** NVM-DH-HL-CL model structure diagram

## 4 Experiment and analysis

### 4.1 Experimental data

#### 4.1.1 Experimental environment

In the experiment, the model framework is Pytorch1.4, the cpu model is i7-10700F, the graphics card is GTX1080ti, the operating system is

Ubuntu18.04, the learning rate for reference is 1E-4, and the batch-size is 64.

#### 4.1.2 Dataset and data processing

In academic research, there is still no universal and open dataset of fabric images. Therefore, a fabric image retrieval dataset ZSTU-Silk-Set is made according to the real factory. This dataset contains fabric images of more than 3,500 kinds of fabrics, of which 1,720 are labeled, totaling 8,600, and the remaining 1,800 classes include 13,888 images. In order to simulate the real application scene, different photos were taken for each fabric, and 5-10 images were collected for each fabric under different conditions, and the collection conditions were changed from the following angles.

(1) On the shooting equipment: using various mobile phone models, such as Android, iOS, etc., the mobile phone configuration is also from low-end to high-end to realize the diversity of image acquisition.

(2) In the shooting environment: choose to shoot a fabric in different environments and at different times.

(3) In the shooting view: the fabric can be rotated to capture images from different angles; take photos with far and near perspectives, so as to collect some images with different scales; in the actual process, some fabric images with sundries will be photographed. At the same time, in the selection of fabric images, fabric images with different repetition degree of meta-texture will be photographed according to the actual situation.

All 1720 kinds of fabrics are merged into the dataset, and the images of the same fabric will be labeled with identical labels. During training, 1/2 of 1720 samples were randomly selected as the training set, and the rest were used as the test set. Therefore, there are 860 kinds of fabric images composed training sets and 860 kinds of fabric images composed testing sets. In the test process, 1/5 of each class in the test set will be randomly selected as the query image, while the search set will be composed of the remaining 4/5 images and the remaining images of more than 1800 classes.

#### 4.1.3 Evaluation indicators

CBIR has many evaluation indicators, all of which are handled according to a set of guidelines. Some of the most common evaluation indicators are used in the following literature:

(1) Image retrieval average precision (AP) and image retrieval mean average precision (map), which are calculated as shown in formula 18:

$$AP = \frac{1}{N}\sum_{i=1}^{N} \frac{i}{position(i)} \tag{18}$$

Where N indicates the number of the top K images of a single class containing this class, and position(i) indicates the position of the ith image in the search result list. MAP is the average value of all retrieval picture precision.

(2) As an indicator, the Recall is used to indicate the number of correct class in the top K samples. Its calculation is shown in Formula 19:

$$Recall = \frac{A}{P} \tag{19}$$

In Formula 19, A represents the number of similar images retrieved, and P represents the total number of all similar images in the dataset.

### 4.2 Experiment and result analysis

#### 4.2.1 Precision experiment of NMV

In order to find the backbone network which is suitable for this task, and to explore the feasibility of NMV structure. In this paper, the above-mentioned fabric image datasets and related algorithms are used for the following experiments. In this experiment, the last classification layer is removed from the backbone network during retrieval, and the output of the penultimate layer is used for retrieval. The experimental results of the model are shown in Table 4:

Table 4 shows the retrieval MAP and recall of different models. Some conclusions are drawn as follows by analyzing these data:

(1) Among these models, the output dimension of VGG19 is the highest, but the effect of the highest dimension is the worst. The output dimension of Inception v3 is 2048, but the accuracy is only 81.7%. Therefore, in this task, the high output dimension does not mean the high experimental accuracy.

(2) ResNet101 is 51 layers deeper than ResNet50, but the MAP is only increased by 0.4%, while ResNet152 is 101 layers deeper, but the

**Table 4** Fabric image retrieval experiments of different models

| Different network models | MAP(%) | Recall(%) | Output dimension |
|---|---|---|---|
| VGG19 | 76.5 | 52.6 | 4096 |
| ResNet50 | 82.4 | 58.2 | 1024 |
| ResNet101 | 82.8 | 58.8 | 1024 |
| ResNet152 | 78.4 | 53.2 | 1024 |
| DenseNet121 | 84.5 | 63.2 | 1024 |
| DenseNet169 | 84.7 | 64.2 | 1024 |
| DenseNet201-dropout = 0.5 | 81.7 | 60.2 | 1024 |
| DenseNet201-dropout = 0.2 | 81.6 | 60.3 | 1024 |
| Inception v3 | 81.7 | 60.6 | 2048 |
| MobileNetv1 v3 | 78.8 | 59.2 | 1280 |
| MobileNetv2 1.0 v3 | 79.6 | 61.2 | 1280 |
| MobileNetv3 small-1.25 v3 | 83.6 | 63.4 | 1280 |
| shuffleNetV2 1.0 v3 | 82.6 | 62.8 | 1024 |
| NMV 1.0 v3 | 81.4 | 62.6 | 1024 |

MAP value is 4% lower than ResNet50. It is considered that the main reason is the phenomenon of over-fitting.

(3) DenseNet169 has the highest accuracy, but it is only 0.2% higher than DenseNet121, while DenseNet201 has lower accuracy after increasing the number of layers. DenseNet series has higher accuracy than ResNet series, because DenseNet series has the characteristics of feature reuse, which alleviates the influence of over-fitting. However, when DenseNet201 uses dropout of 0.2 or 0.5, the accuracy is not much different, so it is found that the deep models inevitably appear over-fitting.

(4) With the version changes, the effect of MobielNet series is gradually improved, especially MobileNetv3 small-1.25, whose accuracy is only 1.1% lower than that of DenseNet169, but the difference between them in the number of model parameters is at least more than three times. Therefore, it can be judged that this task is more suitable for lightweight models, because such models not only have fast feature extraction speed, but also are less prone to over-fitting. However, because the accuracy of MobileNet series is lower than that of DenseNet121 network model, it is speculated that there is the problem of insufficient feature extraction, and it also shows that cross entropy loss is not completely suitable for such tasks.

(5) At the same time, it is found that ShuffleNet series, MoblieNet series and NMV model have good effects in this experiment, which proves that deep separable convolution can be used in this task.

(6) From the experiment, it is also found that MobileNet series performs well in this experiment, but the residual block of ResNet doesn't seem to have a good effect. At the same time, MobileNetV2 doesn't improve the accuracy of MobileNetV1 much, which means that MobileNetV2 has a general effect compared with the new inverted residual block in MobileNetV1. This shows that in this task, the retrieval accuracy will not be affected if the model doesn't use residual structure.

(7) It is also found that the accuracy of MobileNetv1 is only 78.8%, while that of NMV is 81.4%, and the MAP value of MobileNetv3 small-1.25 has exceeded 82%. Therefore, it is

assumed that it is effective to apply h-swish and SE mechanism modules in this task.

### 4.2.2 Precision comparison experiment of different loss functions

Cross Entropy and HL can be used to represent classification loss, while DSH and CL can be used to represent similarity loss. At the same time, the corresponding effects of different lightweight network structures are completely different. In order to explore the effects of different loss functions and different lightweight models, this paper designs relevant comparative experiments, and the results are shown in Table 5.

Table 5 shows the retrieval MAP and Recall under different methods. By analyzing these data, some conclusions are drawn as follows:

(1) The accuracy of NMV-DH+HL+CL model is 93.2%. Compared with the 1024-dimensional NMV model in the last experiment, NMV-DH is at least 10% higher in MAP and nearly 13% higher in Recall. The result shows that the combination of HL+CL is very effective, and the combination of HL+CL is more effective than that of Cross Entropy +DSH.

(2) It can be seen from the experimental results, compared with MobileNetV3 small-1.25-DH and Shuffle ETV2 1.0-DH, the accuracy of NMV-DH+HL+CL is higher, which indicates that NMV-DH is more suitable for a loss function combination like HL+CL, and further proves the effectiveness of NMV-DH model in this task.

(3) NMV-DH+DSH, a model trained by similarity relation, has a slightly worse effect than the model directly using Cross Entropy loss. However, both CL and DSH can be used as a compensation method to improve the model using Cross Entropy loss, and the effect is remarkable. With this compensation method, MobileNetV3 small-1.25-DH, shuffleNetV2 1.0-DH and NMV-DH all directly surpass the original model without adding -DH layer. This shows that in this task, classification loss may be more effective than similarity loss, and the combination of classification loss and similarity loss can achieve better performance.

### 4.2.3 Visual contrast experiment of different loss functions

In order to further explore the performance of NMV-DH with different loss functions, some

**Table 5** Accuracy comparison experiments of different loss functions

| Different network models | MAP(%) | Recall(%) | Output dimension | Different network models |
|---|---|---|---|---|
| MobileNetV3 small-1.25-DH | 78.5 | 62.4 | 128 | DSH |
| MobileNetV3 small-1.25-DH | 84.2 | 65.2 | 128 | Cross Entropy +DSH |
| MobileNetV3 small-1.25-DH | 85.3 | 65.4 | 128 | Cross Entropy +CL |
| MobileNetV3 small-1.25-DH | 89.2 | 70.5 | 128 | HL+CL |
| shuffleNetV2 1.0-DH | 80.2 | 64.4 | 128 | DSH |
| shuffleNetV2 1.0-DH | 83.8 | 66.4 | 128 | Cross Entropy +DSH |
| shuffleNetV2 1.0-DH | 84.8 | 66.8 | 128 | Cross Entropy +CL |
| shuffleNetV2 1.0-DH | 88.6 | 71.8 | 128 | HL+CL |
| NMV-DH | 79.7 | 63.6 | 128 | DSH |
| NMV-DH | 85.5 | 65.8 | 128 | Cross Entropy +DSH |
| NMV-DH | 87.4 | 69.2 | 128 | Cross Entropy +CL |
| NMV-DH | 93.2 | 75.2 | 128 | HL+CL |

visual experiments were carried out. The results are shown in Fig.5, 6, 7, and 8. Unmarked images represent query images (there are only 4 similar samples in the database), marked images represent results, checkmarks represent right, and red crosses represent wrong.

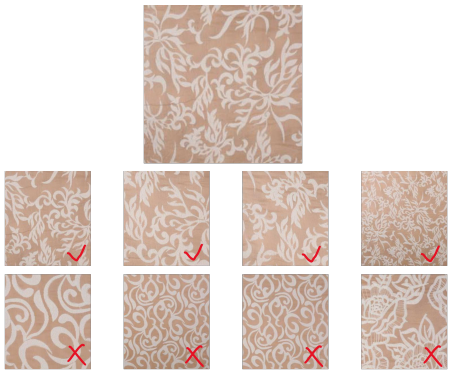

**Fig. 5** HL+CL experimental result

**Fig. 6** DSH+ Cross Entropy experimental result

Through the analysis of the above experimental results, some conclusions are drawn as follows:

(1) The method of HL+CL will search another complete class after searching all the samples in the database, while DSH+ Cross Entropy is more likely to involve other similar classes. This shows that the HL+CL method can classify more effectively than the DSH+Cross Entropy method, which is also in line with the effect that HL method hopes to achieve after improving Cross Entropy, that is, the features of samples within a class can return to the same vector by expanding the feature distance between classes.

**Fig. 7** CL experimental result

**Fig. 8** DSH experimental result

(2) The background color and texture of samples between classes of this dataset are similar, and the gap between samples within classes is not particularly large. Therefore, it is speculated that we don't need a deep network structure to extract particularly complex features, but should pay more attention to how to separate heterogeneous samples, which should be the reason why CL or DSH can improve the effect.

(3) Although the CL method has a high error rate, the background color and texture of the samples obtained by searching are similar from the perspective of the naked eye. However, this is not the case for DSH, which may be related to too many classes and low collision rate of samples (this is a common problem of similarity loss function). Although CL is also low in collision rate, the experimental accuracy is higher.

### 4.2.4 Precision comparison experiment of NMV-DH+HL+Cl

After the above experiments, it is found that NVM-DH with HL and CL has very high accuracy. In order to explore the different performance between this method and other mainstream deep hashing methods, the following experiments are designed. In order to compare the results conveniently, 128-dimensional outputs are selected and the experimental results are shown in Table 6 below:

From the experiment in Table 6, it can be found that the model using NMV-DH as the backbone network is better than the model using AlexNet-DH in this paper, and the effect of HL+CL is better than other loss functions when using the same network structure. When NMV-DH is used, the proposed method improves the MAP and Recall by 4.6% and 12.9% and 14.0% compared with DPN and DPSH respectively.

### 4.2.5 Experiment of exploring the effect of NMV-DH+HL+Cl

In the HL+CL loss function, in order to understand the contribution degree of the loss function more clearly and explore the influence of different dimensions on NMV-DH, the following experiments are designed, and the results are shown in Table 7 below: By analyzing the data in Table 7, we get some conclusions as follows:

(1) In NMV-DH model, the output of 128 dimensions is better than that of 64,48 dimensions.

(2) When used in combination, whether DSH is matched with HL or CL is matched with HL, the performance of HL can be greatly improved. However, the performance of CL is inferior to that of HL when used alone.

(3) When CL loss function is used alone, the changes of MAP and Recall are not particularly obvious in each output dimension of 48, 64 and 128. However, when HL is used alone, the 128-dimension HL increases by 2.2 percentage points on MAP and 3.6 percentage points on Recall. This shows that HL is more sensitive to the output dimension than CL.

**Table 6** Comparison experiments with mainstream methods

| Different network models | MAP(%) | Recall(%) | Output dimension | Loss |
|---|---|---|---|---|
| AlexNet-DH | 73.5 | 54.4 | 128 | DPSH[11] |
| AlexNet-DH | 78.8 | 62.4 | 128 | ADSH[14] |
| AlexNet-DH | 77.6 | 61.2 | 128 | PCDH[15] |
| AlexNet-DH | 78.4 | 61.5 | 128 | DSHSD[16] |
| AlexNet-DH | 75.2 | 58.2 | 128 | DBDH[17] |
| AlexNet-DH | 80.2 | 63.4 | 128 | DPN[18] |
| AlexNet-DH | 86.4 | 70.4 | 128 | HL+CL |
| NMV-DH | 80.3 | 61.2 | 128 | DPSH |
| NMV-DH | 84.2 | 66.4 | 128 | ADSH |
| NMV-DH | 82.8 | 65.8 | 128 | PCDH |
| NMV-DH | 83.8 | 66.8 | 128 | DSHSD |
| NMV-DH | 81.5 | 61.2 | 128 | DBDH |
| NMV-DH | 88.6 | 70.6 | 128 | DPN |
| NMV-DH | 93.2 | 75.2 | 128 | HL+CL |

**Table 7** HL and CL exploration experiments

| Different network models | MAP(%) | Recall(%) | Output dimension | Loss |
|---|---|---|---|---|
| NMV-DH | 87.6 | 69.8 | 48 | HL |
| NMV-DH | 83.5 | 64.4 | 48 | CL |
| NMV-DH | 89.8 | 70.6 | 48 | HL+DSH |
| NMV-DH | 91.4 | 72.8 | 48 | HL+CL |
| NMV-DH | 88.6 | 70.8 | 64 | HL |
| NMV-DH | 83.9 | 64.8 | 64 | CL |
| NMV-DH | 91.0 | 73.6 | 64 | HL+DSH |
| NMV-DH | 92.8 | 74.8 | 64 | HL+CL |
| NMV-DH | 89.7 | 71.4 | 128 | HL |
| NMV-DH | 84.2 | 64.6 | 128 | CL |
| NMV-DH | 91.2 | 73.4 | 128 | HL+DSH |
| NMV-DH | 93.2 | 75.2 | 128 | HL+CL |

### 4.2.6 Dominant visualization experiment of NMV-DH+HL+Cl

Based on NMV-DH+HL+CL, Fig.9 shows some visual experimental results. Unmarked images represent query images, marked images represent results, checkmarks represent right, and red crosses represent wrong.

**Fig. 9** NMV-DH+HL+CL experimental result

Through the analysis of the experimental effect diagram in Fig.9, some conclusions can be drawn as follows:

(1) This method can effectively deal with the influence of scale change.

(2) This method is not affected by color and background to some extent.

(3) This method shows good performance in both simple texture fabric images and complex texture fabric images.

### 4.2.7 Experiment of feature extraction speed

The purpose of constructing NMV is not only to obtain a lightweight network suitable for this experiment, but also to spend less time on feature extraction. In order to study the feature extraction time of different models, this paper has carried out the following experiments: firstly, load a model with Pytorch, then read and preprocess a single picture, then start network model to extract features and start time at the same time, and finally finish timing. Using different models, but similar code writing for testing, the single image feature extraction speed between different models is shown in the following Table 8:

**Table 8** Feature extraction time for different models

| Different models | time/ms |
|---|---|
| ResNet50 | 0.151 |
| DenseNet121 | 0.321 |
| Inception v3 | 0.284 |
| MobileNetV3 small-1.25 | 0.061 |
| shuffleNetV2 1.0 | 0.035 |
| NMV | 0.025 |

**Table 9** Feature matching times for different distance functions

| Different function X1=128,time/ms | X2 = 1e6, |
|---|---|
| Euclidean Distance | 30.6 |
| Manhattan Distance | 51.9 |
| Cosine Similarity v3 | 13.5 |
| Chebyshev distance | 19.3 |
| hamming distance 1.0 | 3.3 |

From the results in Table 8, we can find that the extraction speed of network structure features such as NMV is very fast, and the time spent is almost 1/16 of that of DenseNet121, which is very suitable for application in the dataset of this paper.

### 4.2.8 Experiment of feature matching speed

The speed of image retrieval also depends on the feature matching process after feature extraction. The output characteristics of deep hashing can be represented by binary coding, which can meet the requirements of hamming distance, and this is the biggest advantage of deep hashing compared with other non-hashing methods. In order to explore the speed of NMV-DH model in feature matching, this paper simulates feature matching and uses some commonly used codes to efficiently calculate time. In the experiment, X1 is used to represent the vector dimension of the output, X2 is used to simulate the number of images in the database, and the experimental results of the final speed are shown in Table 9:

It can be seen from Table 9 that when the characteristic dimension of the output is 128, the calculation time of hamming distance is basically 1/10 of Euclidean Distance, 1/17 of Manhattan Distance and 1/4 of Cosine Similarity. Therefore, using Hamming distance for feature matching can achieve great speed advantage in fabric image retrieval.

## 5 Conclusion

Aiming at the problem of efficient fabric image retrieval, this paper constructs NMV-DH network model, which improves the speed of feature extraction and feature matching. A new loss function HL+CL is proposed based on label information, which reduces the influence of large feature difference between similar samples and small feature difference between heterogeneous samples on fabric retrieval accuracy. Among them, by constructing Hadamard matrix, a common center vector is allocated for similar output features, so that the Hamming distance between output feature vectors of different fabric classes is expanded by using class information. By constructing similarity loss function, the similarity between pairs of samples is used for hash coding to minimize the distance between each pair of similar fabric samples and maximize the distance between each pair of different fabric samples. The experimental results showed that this method can consider both retrieval accuracy and retrieval speed, and has certain practical application value. In the future, the method of decision tree will be integrated to realize more efficient fabric image retrieval.

## Declarations

Some journals require declarations to be submitted in a standardised format. Please check the Instructions for Authors of the journal to which you are submitting to see if you need to complete this section. If yes, your manuscript must contain the following sections under the heading 'Declarations':

- Funding
  This research was supported by National Key R&D Program of China (2022YFD2000600),the Key R&D Program of Zhejiang Leading Goose Program (2022C02052). All authors declare no conflict of interest and have approved the journals submitted to you.
- Conflict of interest/Competing interests (check journal-specific guidelines for which heading to

use)
Not applicable

- Ethics approval
  Not applicable
- Availability of data and materials
  Raw data cannot be provided due to laboratory policies or confidentiality agreements
- Authors' contributions
  All authors contributed to the study conception and design. Material preparation, data collection and analysis were performed by Jiangsheng Gui. The first draft of the manuscript was written by Dongwei Wu and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

# References


[1] M.S. Ko, Y.H. Lee, C. Cho, H. Song, in *2021 International Conference on Information and Communication Technology Convergence (ICTC)* (IEEE, 2021), pp. 1659–1661

[2] A. Farruggia, R. Magro, S. Vitabile, A text based indexing system for mammographic image retrieval and classification. Future Generation Computer Systems **37**, 243–251 (2014)

[3] F. Shen, L. Lin, M. Wei, J. Liu, J. Zhu, H. Zeng, C. Cai, L. Zheng, in *2019 IEEE 4th International Conference on Image, Vision and Computing (ICIVC)* (IEEE, 2019), pp. 247–251

[4] A.H. Rangkuti, V.H. Athala, N.F. Luthfi, S.V. Aditama, M.M. Ramadhan, A.H. Aslamia, in *2021 IEEE International Conference on Computing (ICOCO)* (IEEE, 2021), pp. 224–229

[5] S. Tena, R. Hartanto, I. Ardiyanto, in *2021 4th International Seminar on Research of Information Technology and Intelligent Systems (ISRITI)* (IEEE, 2021), pp. 150–154

[6] J. Xiang, N. Zhang, R. Pan, W. Gao, Wool fabric image retrieval based on soft similarity and listwise learning. Textile Research Journal **92**(21-22), 4470–4483 (2022)

[7] N. Zhang, R. Shamey, J. Xiang, R. Pan, W. Gao, A novel image retrieval strategy based on transfer learning and hand-crafted features for wool fabric. Expert Systems with Applications **191**, 116229 (2022)

[8] B. Koonce, B. Koonce, Mobilenetv3. Convolutional Neural Networks with Swift for Tensorflow: Image Recognition and Dataset Categorization pp. 125–144 (2021)

[9] J. Hu, L. Shen, G. Sun, in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2018), pp. 7132–7141

[10] J. Liu, C. Li, F. Liang, C. Lin, M. Sun, J. Yan, W. Ouyang, D. Xu, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition* (2021), pp. 11486–11495

[11] W.J. Li, S. Wang, W.C. Kang, Feature learning based deep supervised hashing with pairwise labels. arXiv preprint arXiv:1511.03855 (2015)

[12] H. Liu, R. Wang, S. Shan, X. Chen, in *Proceedings of the IEEE conference on computer vision and pattern recognition* (2016), pp. 2064–2072

[13] Y. Cao, M. Long, B. Liu, J. Wang, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (2018), pp. 1229–1237

[14] Q.Y. Jiang, W.J. Li, in *Proceedings of the AAAI conference on artificial intelligence*, vol. 32 (2018)

[15] Y. Chen, X. Lu, Deep discrete hashing with pairwise correlation learning. Neurocomputing **385**, 111–121 (2020)

[16] L. Wu, H. Ling, P. Li, J. Chen, Y. Fang, F. Zhou, Deep supervised hashing based on stable distribution. IEEE Access **7**, 36489–36499 (2019)

[17] X. Zheng, Y. Zhang, X. Lu, Deep balanced discrete hashing for image retrieval. Neurocomputing **403**, 224–236 (2020)

[18] L. Fan, K.W. Ng, C. Ju, T. Zhang, C.S. Chan,
in *IJCAI* (2020), pp. 825–831