Journal of Universal Computer Science, *vol. 29*, *no. 8* (*2023*), *938-958*
submitted: *13/12/2022*, *accepted*: *28/3/2023*, *appeared*: *28/8/2023*

Robust Authentication Analysis of Copyright Images through Deep Hashing Models with Self-supervision

Jaeyoung Yang
(AiDeep, Seoul, South Korea,
jyyang@aideep.ai)

Sooin Kim
(Dankook University, Yongin-si, South Korea,
https://orcid.org/0009-0004-1695-6223, 32190756@dankook.ac.kr)

Sangwoo Lee
(AiDeep, Seoul, South Korea,
lswross012@aideep.ai)

Won-gyum Kim
(AiDeep, Seoul, South Korea,
wgkim@aideep.ai)

Donghoon Kim
(Arkansas State University, Jonesboro, AR, USA,
https://orcid.org/0000-0003-4642-4231, dhkim@astate.edu)

Doosung Hwang
(Dankook University, Yongin-si, South Korea,
https://orcid.org/0000-0003-1840-9296, dshwang@dankook.ac.kr)

Abstract: The increased usage of the internet and ICT has posed a significant challenge to protect copyrighted content due to advanced image forgery techniques that make image authentication extremely difficult. The aim of this paper is to establish a binary classification method for determining copyright images from copyright-free ones. A deep hashing model is introduced for an image authentication system, which uses deep learning-based perceptual hashing. Hash codes from a deep hashing model trained with a copyright image dataset are used to identify images. The deep learning model is able to learn features that represent the implicit meaning or structural information of an image. The copyright dataset, which lacks class labels, is trained with deep hashing models with self-supervision. The proposed model is based on an autoencoder or variational autoencoder model and is improved by including convolutional filters, residual blocks, and vision transformer blocks. Experimental results show that the proposed model performs a one-to-one mapping with most stored images and can retrieve related images using image features in hash collisions. The model can find the query image among the top 5 images with comparable hash codes. The results indicate that the proposed deep hashing approach is robust and applicable.

Keywords: Deep hashing, autoencoder, copyright image, data augmentation, self-supervision
Categories: E.2, H2.8, H.3.3, H.3.4, I.2.0
DOI: 10.3897/jucs.98824

1 Introduction

The widespread usage of the Internet and ICT (Information and Communication Technology) enhances the use of various digital content and the availability of shared software, but it makes copyrighted content protection challenging. Image forgery has advanced to the point where multimedia authentication and security are becoming extremely difficult. Image editing software allows for a variety of manipulations, such as modifying the color of the image, changing the salient item, and forging copy movement. This shift in the environment is raising the expectations for digital copyright protection systems. With the expanding copyright issue, there is an urgent demand for technology that can assess image attributes and modulation concerns [Mao et al., 2018, Du et al., 2020, Kim et al., 2021]. The image authentication system compares the hash codes of the original image with the test image. A typical application framework includes feature selection, image hashing, database building, and authentication checks as shown in Figure 1. The hash code of the test image is utilized as a key in the database search and is used to determine whether or not to authenticate.

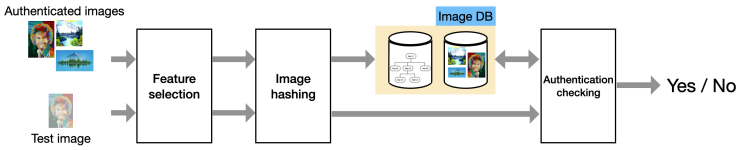

Figure 1: *A typical application framework for image authentication*

Text-based image retrieval (TBIR) is a text-matching search that employs traditional text annotation techniques [Ashok Kumar et al., 2021, Srinivasan et al., 2021]. An input query is text that is used to search for images related to that query. The database stores images together with the selected text annotations for those images, which can be time-consuming. Content-based image retrieval (CBIR) extracts and stores features from images and searches for images with similar features to the query images [Lew et al., 2006, Liang et al., 2017]. CBIR analyzes image content for features, such as color, shape, texture, and other image analysis data, whereas TBIR analyzes image metadata, such as keywords, tags, or descriptions. TBIR and CBIR offer images similar to an input query by associating an image with a key for speedy image retrieval. Perceptual-based image retrieval (PBIR) is utilized in building large-scale image databases to ensure fast retrieval and low storage media usage [Du et al., 2020]. PBIR may build a fixed-length hash code from the image. Hashing is quite efficient in terms of computation and storage. Hashing is the process of converting high-dimensional feature vectors into low-dimensional hash codes so that the hash codes of similar objects are as close as possible, and the hash codes of non-similar objects are as far apart as possible. Although mathematical hashing algorithms are commonly used to create hash keys, deep hashing has recently gained popularity [Luo et al., 2022]. Deep hashing is a feature transformation of trained deep learning models that converts high-dimensional images into low-dimensional hash codes for efficient storage and retrieval. By generating discriminative hash codes, it minimizes quantization errors and facilitates the search for similar images to the input image.

Image hashing allows for efficient storage and accurate retrieval. To accomplish this, binary hash codes are utilized to replace the image's contents. Spectral hash-

ing [Salakhutdinov and Hinton, 2009] and semantic hashing [Weiss et al., 2008] enable the learning of data-independent hash functions with local sensitive hashing in the sub-region of the image. However, it is challenging to map data-independent hashing to the hash code of the forgery image [Paulevé et al., 2010]. The robust feature representation of deep learning can provide very useful end-to-end hashing code in many applications [Zhang et al., 2016, Venkateswara et al., 2017, Qin et al., 2018]. Data-dependent deep hashing generates identifiable hash codes by exploring several types of data within the image, such as shape, texture, and color. Deep learning hashing is widely utilized in the field of image retrieval due to the advantages of rapid query and low storage [Luo et al., 2022, Liu et al., 2019].

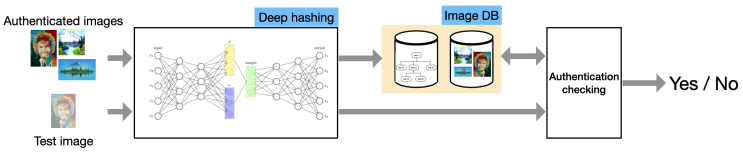

Figure 2: *Deep hashing framework for copyright image authentication*

In this paper, we propose an image authentication system for copyright status determination using deep hashing based perceptual hashing as a binary classification problem. Deep hashing models are employed to generate hash codes from the copyright images, and the hash codes are mapped to the images to create a database. Image authentication using CBIR and TBIR has limitations in associating the external structural features that the image represents. On the other hand, the deep learning model can learn the implicit meaning of the image or the feature that reflects structural information. The hash code generated by the deep hashing model serves as a key to retrieve an image in database. If the retrived image and the query image are the same, the image is determined to be copyrighted; otherwise, it is copyright-free. The difference information is evaluated by comparing the query image with the retrieved image. Figure 2 depicts an image authentication framework.

The database consists of images registered with the Korean Literature, Academic works and Art Copyright Association[1] (KOLAA) and images that do not hold selected copyrights in the public data. A deep hashing model with improved autoencoder and variational autoencoder is proposed and compared using cross-validation. Furthermore, we evaluate the performance of deep hashing by conducting experiments on datasets that apply data augmentation using image manipulation to training datasets. The main contributions are described below:

– A novel deep learning hashing model is proposed for image authentication that utilizes perceptual hashing based on deep hashing. This approach can potentially decrease the high expenses and time needed for image annotation in TBIR and CBIR.

– By learning the implicit meaning of an image or the feature that reflects its structural information, the deep hashing model addresses the issue of limitations in connecting the external structural features that the image represents.

[1] www.kolaa.kr

– This study benefits from the utilization of a vast dataset of images sourced from various formats for experimentation, such as KOLAA with more than 3M images and CIFAR-10 dataset. By utilizing a diverse dataset with a substantial number of images, we are able to comprehensively evaluate and test the effectiveness of our approach in handling a wide range of images. As a result, our results are more robust and generalizable.

– This study elaborates on various existing models with self-supervision and extends them to develop more advanced models using convolutional filters, residual blocks, and vision transformer blocks. Extending existing models offers multiple benefits, including enhancing the model's performance and capabilities, improving its robustness and generalizability, and enabling it to handle more complex tasks.

– This study examines the limitations of the performance evaluation metrics: PSNR (peak signal-to-noise ratio) and SSIM (structural similarity index map). We emphasize the necessity for better metrics to assess image similarity.

This paper is organized as follows. Section 2 discusses related work on hashing technology. Section 3 proposes various deep hashing methods with detailed descriptions to detect copyright images. Section 4 describes the experimental procedures and evaluates each deep hashing model. Finally, Section 5 concludes this work with a summary.

2 Related works

There are three types of hashing technology: data independence, data dependence, and deep hashing. Data independence techniques utilize fundamental hashing ideas like code balance, uncorrelated bits, and similarity preservation criteria. Locality-sensitive hash [Indyk and Motwani, 2000] and kernelized version [Kulis and Grauman, 2009] are examples. In contrast, data dependence techniques aim to discover more succinct yet dataset-specific hash codes [Chatfield et al., 2014]. Deep learning-based hashing techniques have recently used DNNs' excellent capacity for learning to generate hash codes [Xia et al., 2014, Lai et al., 2015, Li et al., 2016]. Deep learning models utilize various loss functions to generate hash codes from the collected dataset.

Hashing is an NP-hard binary optimization problem [Weiss et al., 2008], which is prone to gradient extinction problems due to discrete and undifferentiable binary hash functions. To resolve this issue, the loss function discarded the discrete constraints. A deep learning model is trained by defining various loss functions based on the collected dataset [Lai et al., 2015, Li et al., 2016]. Hash code binarization applies a sigmoid or tanh function of vector values extracted from the trained model. [Cao et al., 2017].

A hash function must maintain as much distance as possible in the data space in order to guarantee high discriminatory power. Two vectors that are close in hashing space should also be as close to each other in data space as possible. Deep Polarized Network (DPN) defines the differentiable bit-wise hinge-like optimization loss to generate target vectors with the maximum interclass distance [Fan et al., 2021]. Yuan *et al*. [Yuan et al., 2020] proposed a global similarity metric, called central similarity, to improve hash learning efficiency and retrieval accuracy. Alternatively, the binary or ternary loss function and the cosine similarity were reflected in the quantization error [Gong et al., 2012, Hoe et al., 2021]. Furthermore, deep hypersphere embedding learning methods are proposed to improve the angular margin of learned features and impose discriminative

constraints on a hypersphere manifold: sphereFace [Liu et al., 2017], CosFace [Wang et al., 2018], and ArcFace [Deng et al., 2022].

The channel attention has been demonstrated to potentially improve the performance of deep convolutional neural networks [Yang et al., 2022]. ECA-NET [Wang et al., 2020] used a local cross-channel interaction strategy without dimensionality reduction that can be efficiently implemented through 1D convolution. Zhe *et al*. [Zhe et al., 2021] proposed a semantic learning model that preserves deep hashing by adding a parameter-free attention (PFA) layer to ResNet18. DSH (Deep Supervised Hashing) builds a CNN model from a pair of images with a label indicating if the two images are similar to the training input, and it then generates a hash code from the pair of images [Liu et al., 2019]. Vijetha *et al*. [Gattupalli et al., 2019] designed the feature map to be learned through the hashing layer and tag layer using image tags. Hemanth V et.al [Venkateswara et al., 2017] proposed a deep hashing model that can extract hash codes using both supervised and unsupervised learning methods.

Caron *et al*. [Caron et al., 2018] applied k-means to the output of the transfer model to assign clusters and then model learning using backpropagation algorithms. Pre-trained CNNs can give sufficient features that can be used to improve the generalization of models learned from a limited amount of data [Razavian et al., 2014]. Transfer learning is a CNN model that learns ImageNet and fine-tunes the training dataset output by grouping the pseudo-class as the real class of the input data. Osaulenko [Osaulenko, 2020] proposed a binary autoencoder that learns by multiplying an input vector by a random matrix and applying a different threshold for each neuron. The decoder reconstructs the input from the hidden layer by minimizing the error. Variational autoencoders (VAE) train hash code learning with standard backpropagation with self-supervision [Mena and Ñanculef, 2019].

Deep hashing generates hash codes from the collected dataset as opposed to traditional hashing algorithms. The optimization algorithms are performed by introducing a binary loss function, fine-tuning it with transfer learning, and learning in a self-supervised manner. The hash code is generated by hashing the output of the hash output layer or the intermediate layer. The models were evaluated with CIFAR-10, NUS-WIDE, ImageNet, and MS COCO datasets.

3 Proposed approach

3.1 System architecture

Figure 3 depicts a copyright image authentication system consisting of three main components: an image database, a deep hashing model, and a discrimination module. MongoDB is used to build the image database, which contains hash keys of lengths 16, 32, 64, or 128 bits that are linked to copyrighted images. The system is capable of verifying whether a given image is a copyright image or not. This is accomplished by comparing the given image with a reference image, which is already known to be a copyright image. The given image is presented on the left-hand side and the reference image is displayed on the right-hand side of the deep hashing model. Deep hashing models leverage deep learning techniques for self-supervised learning. These models take an image as input and produce a fixed-length hash code, which serves as a search key for retrieving matching images from a database. The database contains the hash code of the copyrighted image, its feature vector, and the path to the original image. The feature vector of an image, which is extracted using a trained deep hashing model, is a unique identifier for the image.

Deep hashing models that output hash codes and low-dimensional feature vectors are AE, Res-AE, Con-VAE and VTE. The image matching the hash code and the feature vector are used to evaluate Hamming dissimilarities with the feature vector from the deep hashing model. If the two hash codes are the same, the input image is determined to be a copyright image. This process is authentication discrimination based on image identification because the same stored image exists as the test image. The input image, on the other hand, is not a copyright image if the same hash code or two feature vectors are not similar. The dissimilarity of the retrieved image with the smallest Hamming distance is calculated for image comparison.

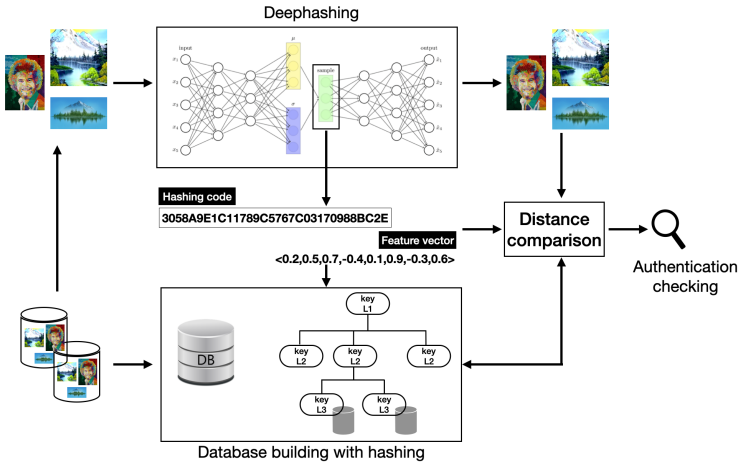

Figure 3: *System architecture*

3.2 Data preparation

The copyright image dataset was provided by KOLAA. The Bluecanvas, Korea Photographers Association, and Korea Culture and Arts Association are organizations that have image datasets that include pictures collected from educational textbooks. Copyright-free image datasets are public image datasets. Table 1 shows the sources and numbers of data in the copyright and public datasets. Each data set has various characteristics, such as image size, storage format, color, image type, and resolution. The dataset comprises a total of 3,677,015 images, and out of those, 15.14% are copyrighted-protected images. To train the deep hashing model, a random selection of 100,000 images is used as the training datasets. This dataset includes 80,000 copyrighted images and 20,000 copyright-free images. Five training and testing datasets were used, and any duplicate images have been removed.

Most images consist of pixels within $8,000 \times 12,000$. The image formats are JPEG, PNG, SVG, WebP, GIF, BMP, TIFF, AVIF, CR2, DNG, ARW, and EPS. KOLAA provides a text-based search service by storing images along with tag information. However, the tag information is manually processed and is not available for every image. Therefore, supervised learning is not possible when choosing a deep hashing model.

Type	Provider	No	Total	Dataset
Copyright	Bluecanvas	229,756	556,802	80,000
	Korea Photographers Association	24,925		
	Korean Culture and Arts Association	301,758		
	Textbook scans	363		
Copyright free	ILSVRC2012	1,500,000	3,120,213	20,000
	Place365	716,680		
	Sun Database	207,939		
	ADE20k	403,900		
	Flicker	25,000		
	CIFAR10	60,000		
	Nus-Wide	206,694		
Total			3,677,015	100,000

Table 1: *Experimental data*

3.3 Deep learning analysis

Copyright images have various sizes and storage formats. The image dataset was transformed into a grayscale image of size 128×128 and scaled to a value between $[0, 1]$. The proposed deep hashing model extracts the hash code $\mathbf{h} = [h_i]_d$ from encoded vector $\mathbf{v} = [\mathbf{v}_i]_d$ by setting the input vector $\mathbf{x} = [x_i]_{128\times128}$ to low d dimensions. Vectors encoded in low dimensions are latent vectors and contain the key characteristics of input data. Latent vectors are binarized to generate their hash code $\mathbf{h}$. The threshold vector $\boldsymbol{\theta}$ uses the average of the training dataset. The hash code $\mathbf{h} = [h_i]_d$ of image $\mathbf{x}$ of the deep hashing model from the training dataset $\mathcal{D} = \{\mathbf{x}^{(i)} | i = 1, 2, \ldots, N\}$ is as follows.

$$\boldsymbol{\theta} = \frac{1}{N} \sum_{i=1}^{N} \mathbf{x}^{(i)}$$

$$h_i = \begin{cases} 1 & \text{if } x > \theta_i; \\ 0 & \text{otherwise.} \end{cases}$$

Autoencoder

The autoencoder (AE) compresses and encodes the input data and then reconstructs it as an output that approximates the original input [Goodfellow et al., 2016, Salakhutdinov and Hinton, 2009]. Through this process, the noise of the data is reduced or used for reduced-dimensional data representation. The architecture of the autoencoder consists of two main parts. The first is an encoding step in which input data is systematically reduced in complexity by multiple layers inside a neural network. The encoder output is a compressed vector of the input, called a low-dimensional latent vector. In the decoding step of the second part, the latent vector is reconstructed as close as possible to the input data. Although information loss via latent vector representations occurs, AE performs well enough to closely match the input data. The latent vector is used to reduce the complexity of the input data. Thus, the input data can be represented in a lower dimension than the input dimension while still providing a vector with consistent characteristics.

The low-dimensional intermediate layer is a latent vector that captures the hidden characteristics of the input data and generates hash codes. An autoencoder consists of neurons the same size as the input and the output layer. In general, the layers of the encoder and decoder are set to be symmetrical. The number of neurons in the encoder decreases toward the latent layer, and the number of neurons increases towards the output

layer. After learning, the latent layer output becomes an encoding vector that matches the low-dimensional input data. The decoder generates an image similar to the input data as an input of the decoder. A hash code is generated by binarizing the latent vector of the input image.

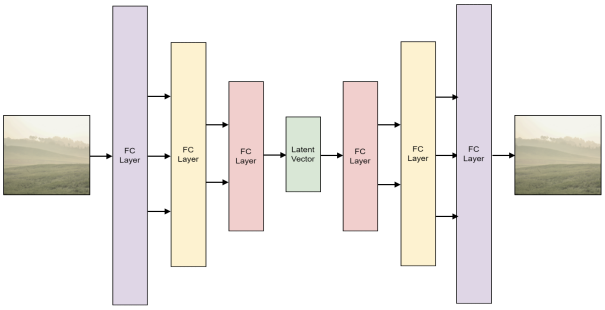

Figure 4: *Autoencoder model for analyzing copyright images*

Figure 4 is an autoencoder model (AE) for copyright image authentication. The input image is flattened into a one-dimensional vector. The encoder and decoder each consist of three fully connected layers. The activation function of the decoder is sigmoid, and that of the encoder is ReLU. The dropout rate of the fully connected layer is set to 0.2 to prevent overfitting [Srivastava et al., 2014]. The loss function is the mean squared error between an input image and its corresponding reconstructed image. The Adam optimizer [Kingma and Ba, 2014] is employed to learn the weights of the designed model by minimizing the loss. Autoencoders typically require longer training times because they rely on fully connected layers, which can increase the number of parameters in the model. Furthermore, the input data is often flattened in autoencoders, which can result in the loss of spatial information.

Autoencoder with Residual Block

Residual blocks are a model construction strategy that helps mitigate the problem of expected *gradient vanishing* [Goodfellow et al., 2016] in training deep neural networks. A residual block is applied by a skip connection block that learns residual functions by referring to the input of a layer instead of learning entirely new functions [He et al., 2015]. When representing the basic mapping as $H(\mathbf{x})$, $F(\mathbf{x})$ of the accumulated nonlinear layer fits the other mapping $F(\mathbf{x}) = H(\mathbf{x}) - \mathbf{x}$. Thus, the original mapping is recast as $F(\mathbf{x}) + \mathbf{x}$, and $F(\mathbf{x})$ is called an error block. Instead of directly passing the input vector to the output vector, a shortcut connection is added that bypasses a block of consecutive layers and adds the input vector directly to the output vector. A skipped connection block makes it easy for the model to map similarly to the input. The residual block strategy has been proven to be effective in the improvement of deep neural network such as ResNet-50 and ResNet-101.

Autoencoders are known to have poor performance on CIFAR-10 and ImageNet, as compared to other models [He et al., 2015, Mao et al., 2018], and they often require longer training time. To overcome these drawbacks, we design an autoencoder that includes residual blocks. This remedy can help alleviate the training time issue because

residual blocks can learn shortcut functions, leading to reduced training time compared to conventional autoencoders.

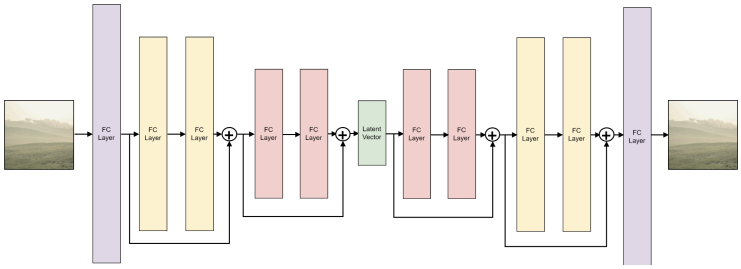

Figure 5: *Autoencoder model with residual blocks for analyzing copyright images*

Figure 5 is an autoencoder model with residual blocks (Res-AE) for the analysis of copyright images. The image is flattened and fed into the encoding block. As the image passes through the encoding layer, the features are mapped to a latent vector through skip connections. The encoder or decoder of Res-AE is composed of two skip connections and five fully connected layers. The activation function of all fully connected layers, including the encoder, uses ReLU. On the other hand, the output layer of the decoder utilizes sigmoid. Res-AE is trained using the same optimizer and loss function as AE.

Convolutional Variational Autoencoder

The likelihood-based generative model known as the variational autoencoder (VAE) is similar to the autoencoder model in that it consists of an encoder and a decoder [Kingma and Welling, 2014]. The encoder transforms the input data into its corresponding latent data, while the decoder takes the latent data and outputs its reconstructed output. Neural network blocks are typically used as the encoders and decoders for these components. By performing variation inference, the posterior of the model is approximated. The encoder maps the input data to its latent data, which corresponds to the parameters of the variance distribution. As a result, the encoder can generate various samples that come from the same distribution. VAE has the advantage of assessing model quality through importance sampling or log-likelihood estimation as a lower bound. However, VAE generates blurrier samples due to the injected noise and imperfect reconstruction by the standard decoder. The decoder has an inverse function that maps from the latent space to the input space for generating data points. While the variance of the noise model can be trained separately, the encoder and decoder are typically trained together using a reparameterization trick.

Figure 6 is a convolutional VAE (Con-VAE) model for copyright image analysis. Autoencoders lose spatial information in the process of flattening an image. Convolutional filters can extract spatial information and localized characteristics of images. When the encoding block is made up of convolutional filters, the shape of the input data is maintained, and the features of the input data can be expressed with the same latent vector shape. In addition, the convolutional filter requires less training time than the fully connected layer. The training image is input to the encoding block as a 2D image array.

The input images are mapped to more detailed features through a convolutional filter every time it passes through the encoding layers. The features of the encoded image

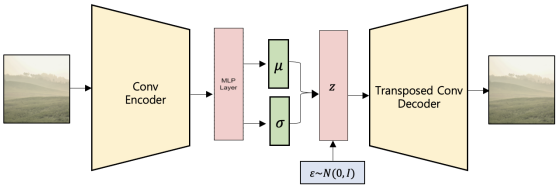

Figure 6: *Convolutional VAE model for analyzing copyright images*

become latent vectors through a flattening process and a variation inference process. The encoding block consists of 6 convolutional layers. The kernel size of the convolutional filter is 2×2, and the stride is 2. The initial number of convolutional filters is 32, and the number of filters in the previous layer doubles as the layer gets deeper. The decoder also consists of 6 transposed convolutional layers, but the convolution layers conduct upsampling from the latent vector to the reconstructed input. The kernel size of the preconvolution filter is 4×4, and the stride is 2. The initial number of preconvolution filters is 512, and each pass through a layer halves the number of filters in the previous layer. The activation function of the decoder output is sigmoid and that of the other layers is ReLU.

Variational Transformer Encoder

The vision transformer model [Dosovitskiy et al., 2020, Dubey et al., 2022] is specifically designed to process and interpret visual data by dividing the image into a series of position-embedding patches. A key feature of this model is its ability to learn the spatial relationships between different parts of the input. Self-attention is a mechanism to focus on specific subregions of the input, so a neural network is able to learn hierarchical and ordering relationships between entities [Yang et al., 2021]. Attention is used either with convolutional neural networks or to replace specific parts of a model while keeping the overall construction intact.

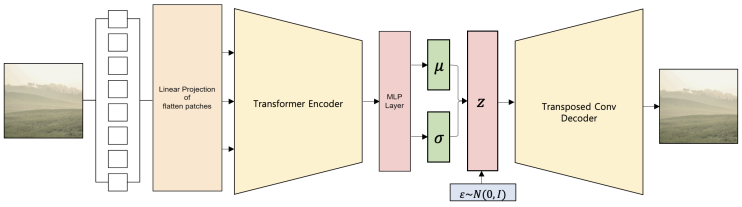

Figure 7: *Variational autoencoder with vision transformer and self-attention*

In many copyrighted images, such as landscapes and crowd photos, it is difficult to identify objects of interest. These images should be differentiated by whole features, not local features. Con-VAE models tend to lose local features when learning features from images. However, the vision transformer can learn relationships between positional embedding patches, whereas the self-attention can learn to highlight key subregions. In deep learning, the addition of vision transformer and self-attention can learn the relationship between local and global features of an image.

Figure 7 shows a variational transformer encoder (VTE) with vision transformer and self-attention. The vision transformer [Dosovitskiy et al., 2020] extracts the global information of an image along with a self-attention module. Self-attention has good computational efficiency and extensibility to other models [He et al., 2015, Yang et al., 2021]. If the encoder is composed of vision transformer, the overall shape and main features of the input data can be extracted. The latent vector catches the global information of the input. In addition, self-attention has fewer parameters than convolutional neural networks, so computing resources are saved during learning. The training image is divided into 16×16 sized chunks and flattened, and this vector is called a patch. The patch is input to the encoding block through linear projection. Features of the input image are extracted through a multi-head attention layer. The extracted image features are the input to the fully connected layer and calculates its latent vectors through variational inference. All activation functions of the encoding layer are GELU [Hendrycks and Gimpel, 2016]. The decoder has the same structure as that of Con-VAE.

4 Experimental results

The experimental dataset is of various sizes and storage formats. Thus, it is transformed to a fixed size and a single storage format JPEG for deep hashing model training. The prepared dataset and the dataset extended with data augmentation are used on deep hashing models. The dataset was converted to a grayscale image of size 128×128 and scaled to have a value between $[0, 1]$. Because there is no label on the copyright image, an unsupervised learning algorithm capable of extracting image features is required. In the test phase, an image with the lowest hamming distance between the hash of the query image and the image existing in the database is selected. If a hash code does not exist for the query image, we apply k-nearest neighbor ($k = 1, 3, 5$) to find the images based on the similarity among hash codes in the database. The learning algorithm sets an epoch of 2,000 and uses cosine annealing for a learning rate of 0.001.

The similarity is evaluated with pairs of input and output images reflecting the characteristics of the distribution. The following metrics are utilized for evaluation: MSE (mean square error), PSNR (peak signal-to-noise ratio), SSIM (structural similarity index map), and mAP (mean average precision). The PSNR index of the query image and the reconstructed image has values in the range of $[0, +\infty)$. Therefore, as the similarity between the two images increases, the PSNR increases proportionally. In the experiment, the PSNR value is limited to the range of $[0, 100]$, and if it is close to 100, the query and the reconstructed image are evaluated as identical. SSIM appears as a value in the $[0, 1]$ range, and if it is close to 1.0, the two images are identical.

4.1 Detection of CIFAR-10 images

We trained the proposed deep hashing models on 10,000 test images from CIFAR-10. The query image was tested by randomly extracting 20,000 images from CIFAR-10. Among the hash codes extracted from the query image and the hash codes stored in the search system, the image with the lowest hamming distance was searched, and the labels of each image were compared to evaluate precision, recall, and mAP performance, respectively.

Table 2 shows the precision, recall, and mAP performance of the CIFAR-10 dataset. For the selected indicators, the VAE model outperformed the AE model. All models perform poorly at 16 hash bits, but the precision, recall, and mAP metrics increase when

the hash length increases. When the hash length is 128 bits, the precision and recall of Con-VAE is about 97%, and the recall when the hash length is 64 bits is close to 96%. The deep hashing results of AE and Res-AE are very poor, so training for hash code generation has not been done well. This trend is similarly analyzed in the mAP index and is directly linked to the low evaluation results of low AE and Res-AE.

Model	Metric	Hash bits			
		16	32	64	128
AE	Precison	0.1486	0.1824	0.2523	0.4620
	Recall	0.1524	0.1846	0.2428	0.4594
	mAP	0.1109	0.1185	0.1368	0.2690
Res-AE	Precison	0.0099	0.0230	0.0229	0.2724
	Recall	0.0990	0.0982	0.0985	0.9622
	mAP	0.0999	0.0998	0.0999	0.1038
Con-VAE	Precison	0.7925	0.9623	0.9622	0.9743
	Recall	0.8514	0.9741	0.9743	0.9622
	mAP	0.7926	0.9623	0.9622	0.9622
VTE	Precison	0.3443	0.9058	0.9244	0.9584
	Recall	0.4410	0.9290	0.9412	0.9462
	mAP	0.3443	0.9058	0.9244	0.9541

Table 2: *The performance comparsion with CIFAR-10*

Figure 8 is an example of train and test samples of the Con-VAE model of 64 hash bits. The figure shows input images, model output images, retrieved images, and evaluation metrics for selected queries. QP (Query PSNR) and QS (Query SSIM) denote the PSNR (peak signal-to-noise ratio) and SSIM (structural similarity index) values for the query and search images, respectively. On the other hand, MP (Model PSNR) and MS (Model SSIM) denote the PSNR and SSIM values computed between the input and output images searched by the model. Although the same image was retrieved when the trained image was input with the Con-VAE model, there are differences in the PSNR and SSIM values computed between the input and output images of the model. When testing the Con-VAE model on a copyright image that was not included in the training set, the model retrieved the images in appearance.

4.2 Detection of copyright images

Copyrighted datasets contain duplicate images. It was composed of a dataset in which duplicate images were removed through a batch operation. Therefore, the cross-validation dataset was prepared as a classification problem without duplicate images. Experimental evaluations are PSNR and SSIM averages for the training and test sets of the five cross-validation datasets. The selected evaluation index is suitable for copyright image classification without class labels. Because the goal of the proposed deep hashing model is image mapping and identical image search according to hashing code, the identity of the test image and the search image are revealed by comparing each pixel.

Table 3 compares the test results of the proposed model for copyright images (C) and copyright-free images (CF). Top-k is the PSNR and SSIM values of the most similar image among the images retrieved according to the k-nearest neighbor algorithm ($k = 1, 3, 5$). One-to-one mapping was possible within rank 5 for all images included in the training dataset. The PSNR of the copyright image is 94.73 ± 4.02, and the SSIM is 0.92 ± 0.05. The PSNR of the copyright-free image is 28.39 ± 0.25, and the SSIM is 0.22 ± 0.04. The low PSNR and SSIM values show a very high discrepancy between training and test images.

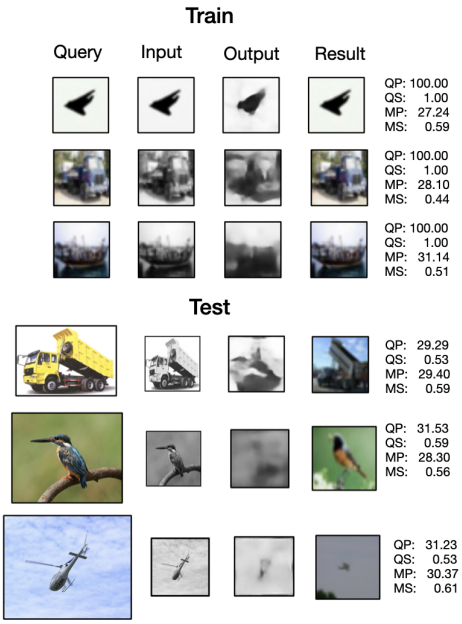

Figure 8: *Selected samples for the CIFAR-10 dataset*

Model	Type	Metric	16bits			32bits			64bits			128bits		
			Top-1	Top-3	Top-5	Top-1	Top-3	Top-5	Top-1	Top-3	Top-5	Top-1	Top-3	Top-5
AE	C	PSNR	87.39	89.13	92.23	94.84	99.04	100.00	99.99	100.00	100.00	100.00	100.00	100.00
	CF	SSIM	0.83	0.84	0.90	0.92	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	C	PSNR	28.31	29.31	29.38	28.33	28.34	29.38	28.12	28.23	28.34	28.43	28.58	28.73
	CF	SSIM	0.10	0.27	0.29	0.13	0.25	0.30	0.12	0.27	0.26	0.13	0.24	0.25
	Mean	PSNR	57.85	59.22	60.81	61.58	63.69	64.69	64.06	64.12	64.17	64.22	64.29	64.37
		SSIM	0.46	0.56	0.60	0.53	0.63	0.65	0.56	0.63	0.63	0.57	0.62	0.63
Res-AE	C	PSNR	79.98	79.98	80.10	81.99	82.19	84.92	85.42	90.24	94.44	92.24	92.30	92.99
	CF	SSIM	0.74	0.74	0.74	0.75	0.79	0.81	0.81	0.84	0.90	0.88	0.89	0.90
	C	PSNR	27.88	27.87	28.12	27.32	27.55	28.44	28.19	28.19	29.12	28.34	28.34	28.34
	CF	SSIM	0.06	0.07	0.08	0.17	0.17	0.18	0.12	0.12	0.13	0.15	0.15	0.15
	Mean	PSNR	53.93	53.93	54.11	54.66	54.87	56.68	56.80	59.21	61.78	60.29	60.32	60.66
		SSIM	0.40	0.40	0.41	0.46	0.48	0.49	0.47	0.48	0.52	0.52	0.52	0.52
Con-VAE	C	PSNR	92.32	93.33	94.31	94.84	99.04	100.00	99.99	100.00	100.00	100.00	100.00	100.00
	CF	SSIM	0.91	0.91	0.93	0.92	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	C	PSNR	28.32	28.33	28.34	28.43	28.44	28.44	28.44	28.44	28.44	28.44	28.44	28.44
	CF	SSIM	0.25	0.25	0.25	0.25	0.33	0.33	0.28	0.29	0.29	0.29	0.29	0.29
	Mean	PSNR	60.32	60.83	61.33	61.63	63.74	64.22	64.21	64.22	64.22	64.22	64.22	64.22
		SSIM	0.58	0.58	0.59	0.59	0.66	0.67	0.64	0.64	0.64	0.64	0.64	0.64
VTE	C	PSNR	91.93	93.02	94.10	94.93	99.97	100.00	99.99	100.00	100.00	100.00	100.00	100.00
	CF	SSIM	0.90	0.91	0.92	0.93	1.00	1.00	1.00	1.00	1.00	1.00	1.00	1.00
	C	PSNR	28.32	28.32	28.34	28.32	28.33	28.34	28.42	28.43	28.44	28.44	28.44	28.44
	CF	SSIM	0.24	0.24	0.25	0.25	0.25	0.25	0.28	0.29	0.29	0.29	0.29	0.29
	Mean	PSNR	60.12	60.67	61.22	61.63	64.15	64.17	64.21	64.22	64.22	64.22	64.22	64.22
		SSIM	0.57	0.58	0.59	0.59	0.63	0.63	0.64	0.64	0.64	0.64	0.64	0.64

Table 3: *Performance comparison of deep hashing models*

The hash length is compared based on the mean squared error (MSE) of the feature vector extracted from the query image, the latent vector stored in the database, and the PSNR of the original images. Figure 9 is a graph of changes between MSE and PSNR values of the validation dataset tested during training of the proposed model. The smaller the change in MSE and PSNR, the more appropriate the size of the hash space in which the hash size is mapped one-to-one with the training data. As MSE decreases, PSNR increases proportionally. This change is necessary to select an appropriate hash size. For

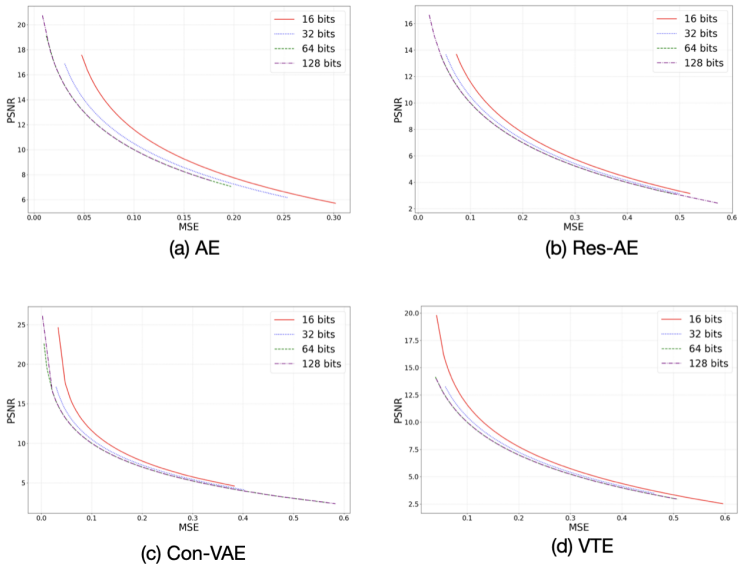

Figure 9: *Comparison between MSE and PSNR*

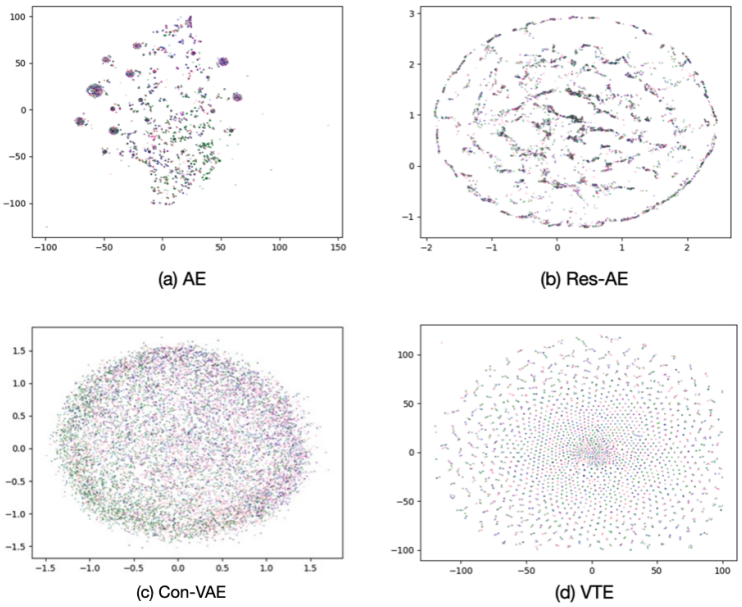

Figure 10: *Hash code distribution by t-SNE*

the proposed deep hashing algorithm, a hash size of 64 bits or 128 bits is appropriate.

Figure 10 is the t-SNE distribution of hash codes learned from the proposed deep hashing model. As shown in Figures 10 (a) and (b), the hash codes generated by AE and Res-AE are represented by scattered chunks concentrated in small areas. While hash codes in different areas are well distinguished, it is difficult to distinguish them in the same area clearly. However, in Figures 10 (c) and (d), the hash code distribution is expressed as being evenly distributed. The hash codes generated by Con-VAE and VTE provide high discriminatory power. The deep hashing method is able to learn hash codes that reflect the intra-distance between similar images and the inter-distance among dissimilar groups.

For a dataset of 40,000 copyright images, the hash collision rate was analyzed.

$$\text{Collision rate} = \frac{\text{No. of collided hashs}}{\text{No. of images}}$$

The hash collision rate of the test data is a bit higher than that of the training data. Most of the proposed deep hashing models have less than 100 hash collisions (Table 4). Although the collision rate increases with hash size, the hash collision rate of Con-VAE and VTE is the smallest for the 64 bit hash code. Overall, the collision rates of the copyright and copyright-free datasets are very small, so a deep hashing model is likely to be applied.

Model	Data	Hash bits			
		16	32	64	128
AE	Copyright	0.0034	0.0032	0.0029	0.0027
	Copyright-free	0.0036	0.0028	0.0026	0.0030
Res-AE	Copyright	0.1207	0.0633	0.0213	0.0032
	Copyright-free	0.1090	0.1025	0.0310	0.0039
Con-VAE	Copyright	0.0028	0.0023	0.0021	0.0021
	Copyright-free	0.0031	0.0026	0.0024	0.0022
VTE	Copyright	0.0025	0.0023	0.0021	0.0021
	Copyright-free	0.0028	0.0024	0.0022	0.0022

Table 4: *Collision rate of the proposed models*

The learned Con-VAE model output the identical images for most of training images. The PSNR and SSIM of the input and output images of Con-VAE differ in Figure 11. For test images that were not included in the training dataset, their similar images were retrieved, yet with different PSNR and SSIM values.

4.3 Detection of copyright images with data augmentation

For the purpose of comparing the performance of the proposed models, the experiments are conducted on transformed images. Additional training datasets are image flip, rotation, cropping, and noise injection. Rotation was selected at 5° intervals from -15° to 15°. Thus, a total of 6 additional images were added. Crop selected four types of center cropped images at 5% intervals from 80% to 95% compared to the original. When the size of the image dataset varies for cropping, the center of the image is cropped and transformed into a fixed size. A noise-injected image was added using the Gaussian kernel. As four images were added for each type, the number of training data increased by a factor of 16, and the training time also increased proportionally.

Table 5 is the result of the experiment with the added images, and the experiment setting is the same as the previous experiment. Compared to the case without data

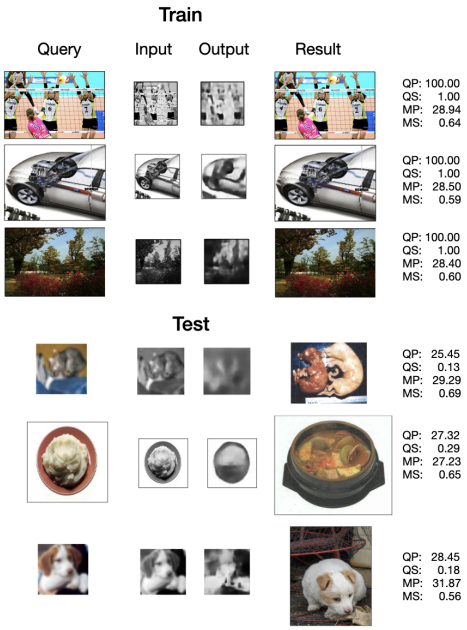

Figure 11: *Selected samples for the copyright dataset*

Model	Type	Metric	16bits			32bits			64bits			128bits		
			Top-1	Top-3	Top-5	Top-1	Top-3	Top-5	Top-1	Top-3	Top-5	Top-1	Top-3	Top-5
AE	C	PSNR	27.74	28.41	28.55	27.40	28.49	30.05	27.82	28.83	30.20	28.98	29.10	30.14
	CF	SSIM	0.53	0.60	0.63	0.60	0.64	0.65	0.61	0.62	0.62	0.61	0.62	0.62
	C	PSNR	26.06	26.10	26.30	26.80	26.93	26.95	27.92	28.43	28.68	28.72	29.57	29.29
	CF	SSIM	0.27	0.27	0.28	0.27	0.27	0.27	0.28	0.28	0.28	0.28	0.28	0.29
	Mean	PSNR	26.90	27.25	27.42	27.10	27.71	28.50	27.87	28.63	29.44	28.85	29.34	29.72
		SSIM	0.40	0.44	0.45	0.44	0.46	0.46	0.45	0.45	0.45	0.45	0.45	0.45
Res-AE	C	PSNR	26.11	27.00	27.49	26.15	27.03	27.44	26.41	27.15	27.39	26.35	27.94	28.12
	CF	SSIM	0.29	0.34	0.43	0.23	0.25	0.23	0.28	0.30	0.33	0.27	0.28	0.31
	C	PSNR	26.28	27.20	27.34	26.94	27.34	27.54	27.67	27.78	27.89	27.58	28.13	28.12
	CF	SSIM	0.05	0.06	0.07	0.17	0.16	0.19	0.12	0.13	0.13	0.14	0.15	0.15
	Mean	PSNR	26.20	27.10	27.42	26.54	27.19	27.49	27.04	27.47	27.64	26.97	28.04	28.12
		SSIM	0.17	0.20	0.25	0.20	0.20	0.21	0.20	0.21	0.23	0.21	0.21	0.23
Con-VAE	C	PSNR	26.13	27.21	28.34	26.25	28.83	30.20	27.82	28.83	30.20	28.98	29.10	30.20
	CF	SSIM	0.50	0.53	0.55	0.58	0.60	0.60	0.61	0.62	0.62	0.61	0.62	0.63
	C	PSNR	27.18	27.23	27.30	27.00	27.92	28.92	27.55	28.50	28.67	28.29	29.95	29.44
	CF	SSIM	0.28	0.29	0.30	0.28	0.28	0.29	0.30	0.30	0.30	0.30	0.30	0.31
	Mean	PSNR	26.66	27.22	27.82	26.62	28.38	29.56	27.69	28.67	29.44	28.64	29.53	29.82
		SSIM	0.39	0.41	0.42	0.43	0.44	0.45	0.46	0.46	0.46	0.46	0.46	0.47
VTE	C	PSNR	26.30	27.95	28.05	26.25	28.84	30.23	27.89	28.82	30.19	28.82	30.21	30.21
	CF	SSIM	0.58	0.59	0.60	0.58	0.59	0.61	0.61	0.62	0.61	0.62	0.62	0.63
	C	PSNR	27.21	27.24	27.29	27.24	27.93	28.97	27.49	28.52	28.66	28.35	29.83	29.90
	CF	SSIM	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.29	0.30	0.29	0.30	0.31
	Mean	PSNR	26.76	27.60	27.67	26.74	28.38	29.60	27.69	28.67	29.43	28.58	30.02	30.06
		SSIM	0.43	0.44	0.45	0.43	0.44	0.45	0.45	0.45	0.46	0.45	0.46	0.47

Table 5: *Performance comparison of deep hashing models on the augmented dataset*

augmentation (Section 4.2), the difference between PSNR and SSIM of training data is too high. However, in the test results, the difference is insignificant. The PSNR of the copyright image is 28.25 ± 1.07, and the SSIM is 0.53 ± 0.03. PSNR of the copyright free image is 27.14 ± 0.90, SSIM is 0.25 ± 0.02.

Figure 12 shows an example of training and test samples of data augmentation-based Con-VAE models. In contrast to the previous experiments, the majority of the training images were found in Top-5 during the search. However, there were differences in the

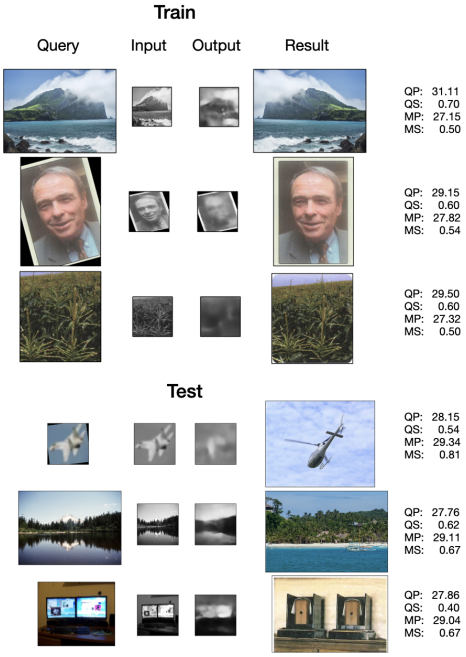

Figure 12: *Selected samples for the copyright dataset with data augmentation*

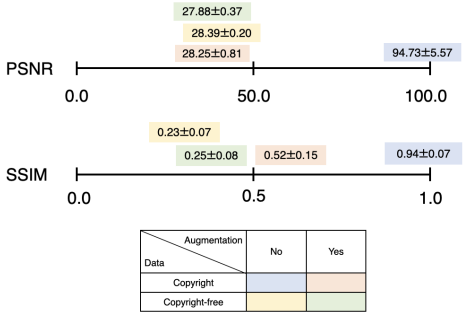

Figure 13: *Comparison between PSNR and SSIM*

PSNP and SSIM values. When testing with images that were not included in the training, similar images to the trained image were returned, but their PSNR and SSIM values varied.

Figure 13 analyzes the experiments of both Table 3 and Table 5. For evaluating deep hashing models, SSIM is analyzed to be more suitable than PSNR. Each performance metric is displayed as the mean and variance of four cases in different colors. Without data augmentation, image authentication can be evaluated according to the PSNR index. However, the two test results show that SSIM is more likely to retrieve the same image in Top-k than PSNR when data augmentation is implemented. Our experiments revealed that using PSNR and SSIM to evaluate the performance of deep learning model-based

image hashing systems can yield considerable discrepancies, even when there are only minor differences in image size or pixels. This study's results confirm this pattern, suggesting that these metrics may not accurately reflect the level of similarity between two images. Thus, it is important to explore and develop alternative metrics that can provide more comprehensive and accurate measures of similarity.

5 Conclusion

By defining copyright image discrimination as a classification problem, improved deep hashing methods based on AE or VAE models are proposed and evaluated. The problem is a binary classification problem from copyright and copyright-free images. The proposed deep hashing methods are AE using dropout, Res-AE including residual blocks, Con-VAE with a convolution layer inserted in VAE's encoder, and VTE connecting image domains with sequence patches. The methods can be expanded from the learned deep hashing model to build an image authentication database for fast image search using hash values. The proposed methods can retrieve original and augmented images by using the learning ability of deep hashing.

The proposed methods were compared using the copyright and copyright-free image datasets. Deep hashing models were trained in a self-supervised manner without class labels that outputs the same output image as the input image. In the mAP evaluation of the CIFAR-10 dataset, VAE was superior to AE. The copyright image dataset was compared with PSNR and SSIM for the original images and the manipulated images. One-to-one mapping was possible for hash codes and feature comparisons of images trained in the two problems, and the collision rate was also analyzed to be low. Based on this experiment, copyright images can be detected by applying an image authentication approach using a deep hashing model. In addition, it is recommended to investigate and create alternative metrics that can provide more comprehensive and accurate measures of similarity in order to overcome the limitations of using PSNR and SSIM metrics to evaluate image similarity.

Acknowledgements

This research is supported by Ministry of Culture, Sports and Tourism and Korea Creative Content Agency(Project Number: 2021-EC-9500).

References

[Ashok Kumar et al., 2021] Ashok Kumar, P. M., Subha Mastan Rao, T., Arun Raj, L., and Pugazhendi, E. (2021). An efficient text-based image retrieval using natural language processing (nlp) techniques. In Satapathy, S. C., Bhateja, V., Janakiramaiah, B., and Chen, Y.-W., editors, Intelligent System Design, pages 505–519, Singapore. Springer Singapore.

[Cao et al., 2017] Cao, Z., Long, M., Wang, J., and Yu, P. S. (2017). Hashnet: Deep learning to hash by continuation. In 2017 IEEE International Conference on Computer Vision (ICCV), pages 5609–5618, Los Alamitos, CA, USA. IEEE Computer Society.

[Caron et al., 2018] Caron, M., Bojanowski, P., Joulin, A., and Douze, M. (2018). Deep clustering for unsupervised learning of visual features. In Ferrari, V., Hebert, M., Sminchisescu, C., and Weiss, Y., editors, Computer Vision – ECCV 2018, pages 139–156, Cham. Springer International Publishing.

[Chatfield et al., 2014] Chatfield, K., Simonyan, K., Vedaldi, A., and Zisserman, A. (2014). Return of the devil in the details: Delving deep into convolutional nets. BMVC 2014 - Proceedings of the British Machine Vision Conference 2014.

[Deng et al., 2022] Deng, J., Guo, J., Yang, J., Xue, N., Kotsia, I., and Zafeiriou, S. (2022). Arcface: Additive angular margin loss for deep face recognition. IEEE Transactions on Pattern Analysis and Machine Intelligence, 44(10):5962–5979.

[Dosovitskiy et al., 2020] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., and Houlsby, N. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. ArXiv, abs/2010.11929.

[Du et al., 2020] Du, L., Ho, A. T., and Cong, R. (2020). Perceptual hashing for image authentication: A survey. Signal Processing: Image Communication, 81:115713.

[Dubey et al., 2022] Dubey, S., Singh, S., and Chu, W. (2022). Vision transformer hashing for image retrieval. In 2022 IEEE International Conference on Multimedia and Expo (ICME), pages 1–6, Los Alamitos, CA, USA. IEEE Computer Society.

[Fan et al., 2021] Fan, L., Ng, K. W., Ju, C., Zhang, T., and Chan, C. S. (2021). Deep polarized network for supervised learning of accurate binary hashing codes. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI'20.

[Gattupalli et al., 2019] Gattupalli, V., Zhuo, Y., and Li, B. (2019). Weakly supervised deep image hashing through tag embeddings. pages 10367–10376.

[Gong et al., 2012] Gong, Y., Kumar, S., Verma, V., and Lazebnik, S. (2012). Angular quantization-based binary codes for fast similarity search. In Pereira, F., Burges, C., Bottou, L., and Weinberger, K., editors, Advances in Neural Information Processing Systems, volume 25. Curran Associates, Inc.

[Goodfellow et al., 2016] Goodfellow, I., Bengio, Y., and Courville, A. (2016). Deep Learning. MIT Press. http://www.deeplearningbook.org.

[He et al., 2015] He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition.

[Hendrycks and Gimpel, 2016] Hendrycks, D. and Gimpel, K. (2016). Gaussian error linear units (gelus).

[Hoe et al., 2021] Hoe, J. T., Ng, K. W., Zhang, T., Chan, C. S., Song, Y.-Z., and Xiang, T. (2021). One loss for all: Deep hashing with a single cosine similarity based learning objective.

[Indyk and Motwani, 2000] Indyk, P. and Motwani, R. (2000). Approximate nearest neighbors: Towards removing the curse of dimensionality. Conference Proceedings of the Annual ACM Symposium on Theory of Computing, 604-613.

[Kim et al., 2021] Kim, D., Heo, S., Kang, J., Kang, H., and Lee, S. (2021). A photo identification framework to prevent copyright infringement with manipulations. Applied Sciences, 11:9194.

[Kingma and Ba, 2014] Kingma, D. and Ba, J. (2014). Adam: A method for stochastic optimization. International Conference on Learning Representations.

[Kingma and Welling, 2014] Kingma, D. P. and Welling, M. (2014). Auto-Encoding Variational Bayes. In 2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings.

[Kulis and Grauman, 2009] Kulis, B. and Grauman, K. (2009). Kernelized locality-sensitive hashing for scalable image search. pages 2130 – 2137.

[Lai et al., 2015] Lai, H., Pan, Y., Liu, Y., and Yan, S. (2015). Simultaneous feature learning and hash coding with deep neural networks. CoRR, abs/1504.03410.

[Lew et al., 2006] Lew, M. S., Sebe, N., Djeraba, C., and Jain, R. (2006). Content-based multimedia information retrieval: State of the art and challenges. ACM Trans. Multimedia Comput. Commun. Appl., 2(1):1–19.

[Li et al., 2016] Li, W.-J., Wang, S., and Kang, W.-C. (2016). Feature learning based deep supervised hashing with pairwise labels. In Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI'16, page 1711–1717. AAAI Press.

[Liang et al., 2017] Liang, Zheng, Yi, Yang, Qi, and Tian (2017). Sift meets cnn: A decade survey of instance retrieval. IEEE Transactions on Pattern Analysis & Machine Intelligence.

[Liu et al., 2019] Liu, H., Wang, R., Shan, S., and Chen, X. (2019). Deep supervised hashing for fast image retrieval. International Journal of Computer Vision, 127.

[Liu et al., 2017] Liu, W., Wen, Y., Yu, Z., Li, M., Raj, B., and Song, L. (2017). Sphereface: Deep hypersphere embedding for face recognition. In Proceedings of the IEEE conference on computer vision and pattern recognition, pages 212–220.

[Luo et al., 2022] Luo, X., Wang, H., Wu, D., Chen, C., Deng, M., Huang, J., and Hua, X.-S. (2022). A survey on deep hashing methods. ACM Trans. Knowl. Discov. Data.

[Mao et al., 2018] Mao, J., Zhong, D., Hu, Y., Sheng, W., Xiao, G., and Qu, Z. (2018). An image authentication technology based on depth residual network. Systems Science & Control Engineering, 6:57–70.

[Mena and Ñanculef, 2019] Mena, F. and Ñanculef, R. (2019). A binary variational autoencoder for hashing. In Nyström, I., Hernández Heredia, Y., and Milián Núñez, V., editors, Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications, pages 131–141, Cham. Springer International Publishing.

[Osaulenko, 2020] Osaulenko, V. (2020). Binary autoencoder with random binary weights. arXiv preprint arXiv:2004.14717.

[Paulevé et al., 2010] Paulevé, L., Jégou, H., and Amsaleg, L. (2010). Locality sensitive hashing: A comparison of hash function types and querying mechanisms. Pattern Recognition Letters, 31(11):1348–1358.

[Qin et al., 2018] Qin, C., Chen, X., Luo, X., Zhang, X., and Sun, X. (2018). Perceptual image hashing via dual-cross pattern encoding and salient structure detection. Information Sciences, 423:284–302.

[Razavian et al., 2014] Razavian, A., Azizpour, H., Sullivan, J., and Carlsson, S. (2014). Cnn features off-the-shelf: An astounding baseline for recognition. volume 1403.

[Salakhutdinov and Hinton, 2009] Salakhutdinov, R. and Hinton, G. (2009). Semantic hashing. International Journal of Approximate Reasoning, 50(7):969–978. Special Section on Graphical Models and Information Retrieval.

[Srinivasan et al., 2021] Srinivasan, K., Raman, K., Chen, J., Bendersky, M., and Najork, M. (2021). Wit: Wikipedia-based image text dataset for multimodal multilingual machine learning.

[Srivastava et al., 2014] Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: A simple way to prevent neural networks from overfitting. Journal of Machine Learning Research, 15(56):1929–1958.

[Venkateswara et al., 2017] Venkateswara, H., Eusebio, J., Chakraborty, S., and Panchanathan, S. (2017). Deep hashing network for unsupervised domain adaptation.

[Wang et al., 2018] Wang, H., Wang, Y., Zhou, Z., Ji, X., Gong, D., Zhou, J., Li, Z., and Liu, W. (2018). Cosface: Large margin cosine loss for deep face recognition. In 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition, pages 5265–5274.

[Wang et al., 2020] Wang, Q., Wu, B., Zhu, P., Li, P., Zuo, W., and Hu, Q. (2020). Eca-net: Efficient channel attention for deep convolutional neural networks. pages 11531–11539.

[Weiss et al., 2008] Weiss, Y., Torralba, A., and Fergus, R. (2008). Spectral hashing. In Koller, D., Schuurmans, D., Bengio, Y., and Bottou, L., editors, Advances in Neural Information Processing Systems, volume 21. Curran Associates, Inc.

[Xia et al., 2014] Xia, R., Pan, Y., Lai, H., Liu, C., and Yan, S. (2014). Supervised hashing for image retrieval via image representation learning. Proceedings of the National Conference on Artificial Intelligence, 3:2156–2162.

[Yang et al., 2021] Yang, L., Zhang, R.-Y., Li, L., and Xie, X. (2021). Simam: A simple, parameter-free attention module for convolutional neural networks. In Meila, M. and Zhang, T., editors, Proceedings of the 38th International Conference on Machine Learning, volume 139 of Proceedings of Machine Learning Research, pages 11863–11874. PMLR.

[Yang et al., 2022] Yang, W., Wang, L., and Cheng, S. (2022). Deep parameter-free attention hashing for image retrieval. Scientific Reports, 12.

[Yuan et al., 2020] Yuan, L., Wang, T., Zhang, X., Tay, F., Jie, Z., Liu, W., and Feng, J. (2020). Central similarity quantization for efficient image and video retrieval. In 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pages 3080–3089, Los Alamitos, CA, USA. IEEE Computer Society.

[Zhang et al., 2016] Zhang, Z., Chen, Y., and Saligrama, V. (2016). Efficient training of very deep neural networks for supervised hashing. pages 1487–1495.

[Zhe et al., 2021] Zhe, X., Ou-Yang, L., Chen, S., and Yan, H. (2021). Semantic hierarchy preserving deep hashing for large-scale image retrieval. 2021 17th International Conference on Machine Vision and Applications (MVA), pages 1–5.