CBIR Using Features Derived by Deep Learning

SUBHADIP MAJI and SMARAJIT BOSE, Indian Statistical Institute, Kolkata

In a Content-based Image Retrieval (CBIR) System, the task is to retrieve similar images from a large database given a query image. The usual procedure is to extract some useful features from the query image and retrieve images that have a similar set of features. For this purpose, a suitable similarity measure is chosen, and images with high similarity scores are retrieved. Naturally, the choice of these features play a very important role in the success of this system, and high-level features are required to reduce the "semantic gap."

In this article, we propose to use features derived from pre-trained network models from a deep-learning convolution network trained for a large image classification problem. This approach appears to produce vastly superior results for a variety of databases, and it outperforms many contemporary CBIR systems. We analyse the retrieval time of the method and also propose a pre-clustering of the database based on the above-mentioned features, which yields comparable results in a much shorter time in most of the cases.

CCS Concepts: • **Information systems → Novelty in information retrieval;**

Additional Key Words and Phrases: Content based image retrieval, feature selection, deep learning, pre-trained network models, pre-clustering

ACM Reference format:
Subhadip Maji and Smarajit Bose. 2021. CBIR Using Features Derived by Deep Learning. *ACM/IMS Trans. Data Sci.* 2, 3, Article 26 (August 2021), 24 pages.
https://doi.org/10.1145/3470568

1 INTRODUCTION

Given a query image, often similar images may need to be retrieved from a large database. This is called **Content-based Image Retrieval (CBIR)**. The standard procedure is to find similar images based on some features extracted from the images. Ideally these features should describe the content information of the images. That is why high-level features are needed, and the low-level features like pixel values, and others, are not very useful.

IBM developed the first commercial version of CBIR system naming **Query By Image Content (QBIC)** [28] in 1995. It allows user to query by user-constructed sketches, example images and drawings. This system uses a combination of texture, shape and colour. **Colour co-occurrence matrix (CCM)** is used to extract low-level features from images, which has been widely used in many works [17, 30, 38] in the area of CBIR. Bose et al. [9] used some visual descriptors to

S. Maji and S. Bose contributed equally to this research.

Authors' address: S. Maji, and S. Bose, Indian Statistical Institute, Kolkata, PIN-700108, Kolkata, Plot No, 203, BT Rd, Dunlop, Bonhooghly Government Colony, Baranagar, 700108; emails: subhadipmaji.jumech@gmail.com, {qr1705, smarajit}@isical.ac.in.

2577-3224/2021/08-ART26 $15.00
https://doi.org/10.1145/3470568

extract features from MPEG-7 standard [22, 25] along with CCM features that achieved some improvement.

Lohite et al. [24] worked with the widely used color, texture and edge features of the images, and optimized the result using **Support Vector Machine (SVM)** classifier. Mehmood et al. [27] presented a CBIR method named **weighted average of triangular histograms (WATH)** of visual words. This method adds image spatial elements to inverted index of **bag-of-visual-words (BoVW)** model, corrects overfitting problem on larger size of dictionary and tries to bride the semantic gap between low-level and high-level features.

Rashno et al. [34] proposed a new scheme that suggests to transform the input RGB image to three subsets in **neutrosophic (NS)** domain. For each of the segment, statistic component, histogram, colour features including **dominant colour descriptor (DCD)** and wavelet features are extracted. These features are then used to retrieve images.

Kumar et al. [37] introduced a new feature descriptor called **local mean differential excitation pattern (LMDeP)**, which can produce robust features. Sarwar et al. [36] recommended a method based on **bag-of-words (BoW)** model, which integrates visual words with **local intensity order pattern (LIOP)** feature and **local binary pattern variance (LBPV)** feature to reduce the semantic gap issue and enhance CBIR performance. Rana et al. [33] proposed image retrieval by combining colour and shape features with nonparametric ranklet transformed texture features. Yusuf et al.[46] gained improvement in CBIR performance on the basis of visual words fusion of **scale invariant feature transform (SIFT**) and **local intensity order pattern (LIOP)** descriptors. Sharif et al.[39] came up with another feature descriptor called **binary robust invariant scalable keypoints (BRISK)** along with SIFT. Ashraf et al. [6] developed a method that retrieves images using YCbCr colour scheme with canny edge histogram and discrete wavelet transform. In Obulesu et al.'s [29] two extended versions of **motif co-occurrence matrices (MCM)** are calculated and combined to improve the CBIR performance. Yosr et al. [45] introduced FSM, a new fuzzy similarity measure motivated by near sets theory and Type-2 Fuzzy logic. Then, they proposed three new IT-2 FSMs with mathematical justification to demonstrate that the proposed FSMs satisfy proximity properties (i.e. reflexivity, transitivity, symmetry, and overlapping). Alsmadi [5] proposed a combination of methods, e.g., neutrosophic clustering algorithm and Canny edge method to extract shape features, YCbCr color with discrete wavelet transform and Canny edge histogram to extract color features, and gray-level co-occurrence matrix to extract texture features. Khan et al. [20] used structured matrix decomposition method to highlight the prominent area and then used **two-dimensional principal component analysis (2DPCA)** to extract feature that results faster recognition. Garg et al. [14] used GLCM features and texture fused LBP variants to extract the statistical characteristics. Singh et al. [42] proposed a **Bi-layer Content-based Image Retrieval (BiCBIR)** method that consists of two modules, the very first module extracts the features of images from database in terms of texture, color and shape. The second module contains two sub-methods: in the first method all images are compared with query image for texture and shape features, and indexes of M most similar images to the given query image are retrieved. Subsequently, the M images that were retrieved from previous layer are matched with query image for color and shape features and finally F images similar to the query image are returned as a output.

After the introduction and evolution of Deep-learning Neural Network, the performance of CBIR has got a boost, because by the help of deep models, we can finally extract higher-level features along with the low-level features from the image to reduce the semantic gap mentioned above. Khokhar et al. [21] described how **Back-propagation Feed-forward Neural Network (BFNN)** can be used for classification in CBIR after exploiting some features of images, e.g., geometric, colour and texture. Ashraf et al. [7] presented a bandlet transform-based image representation

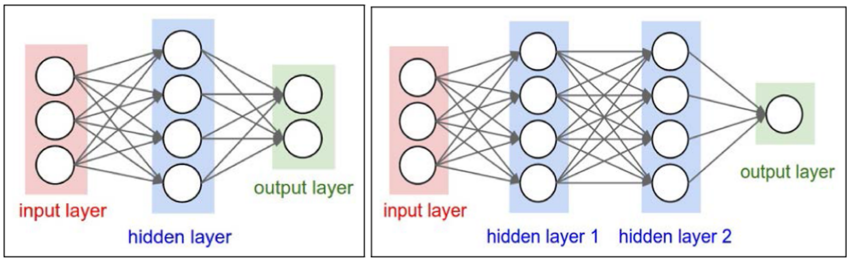

Fig. 1. Left: A two-layer Neural Network (one hidden layer of four neurons and one output layer with two neurons) with three inputs. Right: A three-layer neural network with two hidden layers of four neurons each, one output layer with a single neuron and three inputs [1].

technique that returns information about major objects present in the image reliably. Finally to retrieve images Artificial Neural Network has been used. Xu et al. [44] proposed **part-based weighting aggregation (PWA)** for CBIR. This PWA utilizes discriminative filters of deep convolutional layers as part detectors. Several other recent CBIR techniques can be found in the review paper [48]. Hussain et al. [18] performed K-Nearest Neighbour's model for CBIR using MapReduce model framework and Apache Spark.

In this article, features derived from a pre-trained network model from a deep-learning convolution neural network trained for a large image classification have been used for retrieval of similar images. The resulting algorithms appears to achieve remarkable success in terms of retrieval accuracy, and appears to outperform many contemporary state-of-the-art CBIR methods. The algorithm is quite fast, however, to reduce retrieval time, a concept of pre-clustering the database has also been introduced that seems to work faster without sacrificing retrieval performance.

The article is organized is as follows: Section 2 describes the motivation behind this approach, presents a review of the key ingredients and explores the characteristics of the derived features from a pre-trained network model. In Section 3, we present the details regarding the pre-trained models, similarity measures and evaluation of performance that have been used in this work. Section 4 contains the results of extensive experiments while Section 5 discusses the time complexity. In Section 6, we introduce a concept of pre-clustering of the database and in Section 7, we present the concluding remarks and some future directions.

2 PROPOSED METHOD USING FEATURES DERIVED BY DEEP LEARNING

In a Content-based Image Retrieval system, images are represented by a set of low-level or/and high-level features. This is called feature encoding where an image from RGB or HSV space is encoded to a n-dimensional feature vector. In this article, we propose to derive feature vectors of an image with the help of some pre-trained deep-learning models. For this purpose, we first present a brief description of the key concepts such as neural network, pre-trained models, and so on.

2.1 Neural Network

Neural Networks are set up as collections of neurons that are connected in a non-cyclic graph. These models are often depicted into separate layers of neurons. Generally, the most common type is the fully connected Neural Network layer in which neurons between two adjacent layers are fully pairwise connected, but there is no connection between neurons within a single layer. Below are two examples of fully connected Neural Network [1].

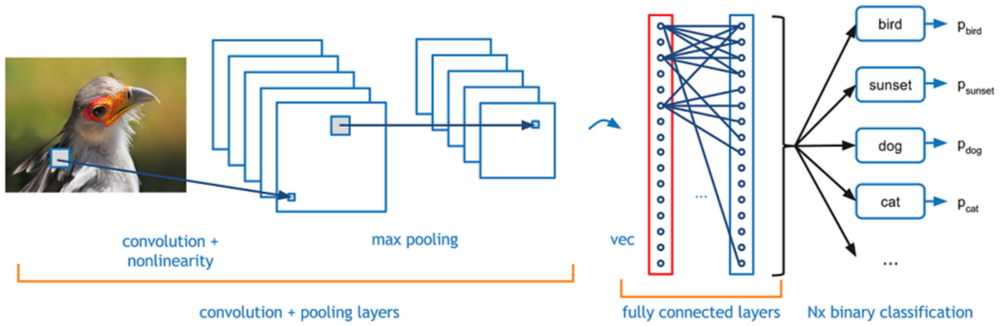

Fig. 2. Every layer of a CNN converts the 3D input volume to a 3D output volume of activations. In given example, the image of the bird is the input layer, so its height and width are the dimensions of the image, and the depth would be three (Red, Green, and Blue channels).

2.2 Convolutional Neural Network (CNN)

Convolutional Neural Networks take input as images and they handle the architecture in a more sensible way. The layers of a CNN have neurons that are arranged in three dimensions: width, height, depth. Here, the word depth refers to the third dimension of an activation volume of a layer. The neurons in a layer are connected to a small region of the preceding CNN layer, unlike to all the neurons, which is a norm in a fully connected neural network. The visualization is shown in Figure 2.

As mentioned above, a simple CNN is a sequence of layers, and every layer of a CNN transforms one volume of activation to another by passing through certain differentiable functions. There are mainly three types of layers in CNN architectures: Convolutional Layer, Pooling Layer, and Fully Connected Layer (shown in Figure 1). We will stack these layers on top of each other to form a fully operational CNN architecture [1]. Figure 2 shows the CNN model architecture of a classification problem. This network consists of convolution, pooling, fully connected layers and some activation layers (e.g., ReLU, softmax, etc.).

2.3 Pre-Trained Neural Network Model

A pre-trained model [26] is a model that was trained on a large benchmark dataset to solve some specific problems similar to the ones we want to solve. Accordingly, because of the high computational cost of training such deep-learning models, it is a common practice to import and use models from a published architecture (e.g., VGG, ResNet, Xception, etc.). A comprehensive review of pre-trained models' performance on computer vision problems using data from the ImageNet [13] challenge is presented by Canziani et al. [10].

Being motivated by this, we have used a pre-trained Neural Network model that was trained on the ImageNet Dataset. This dataset contains more than 14 million images that belong to more than 20,000 classes. It also provides bounding box annotations for around 1 million images, which can be used in Object Localization tasks. Multiple layers of convolutional layers, average pooling layers, max pooling layers, and so on, are stacked up with one another with different combinations to build up the model architectures. The final layer is then unwrapped to an n-dimensional vector, which is called the dense (or fully connected) layer. Several fully connected layers can be stacked on this unwrapped dense layer. Finally, a softmax activation layer of dimension: the number of class is stacked over the final dense layer to get the probabilities of each class for classification problems or a linear activation layer of single dimension can be placed to predict the regressed

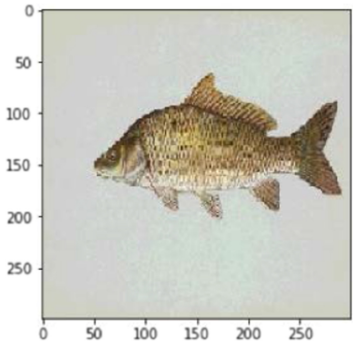

Fig. 3. A sample image from DB2000 Dataset.

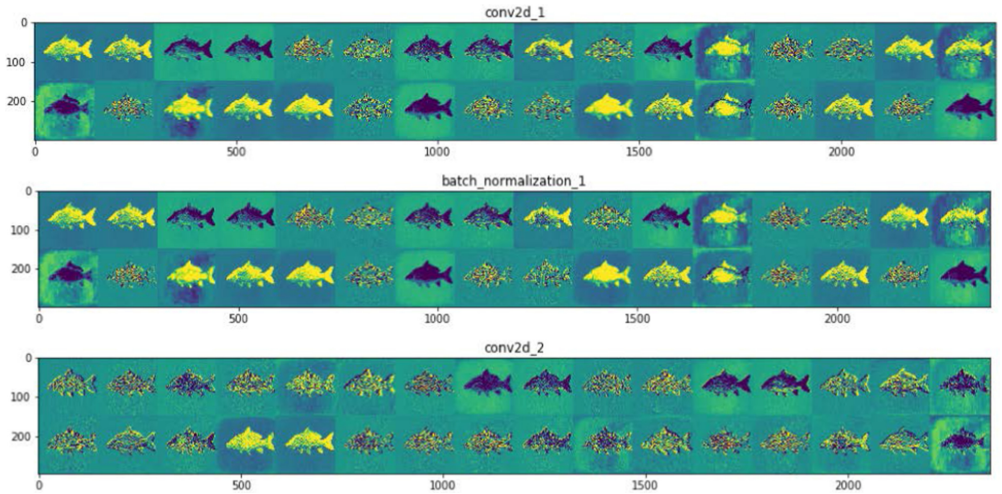

Fig. 4. First few intermediate activations of the image shown in Figure 3.

value for regression problems or other kind of structured activation layers are placed according to the desired problems to solve.

2.4 Visualizing What CNN Learn

It is often referred that deep-learning models are "black boxes." For certain types of deep-learning models it may be true but for CNN it is not absolutely true. The representations and features, learned by CNN are highly amenable to visualization, in large part because they are representations of visual concepts. Since 2013, different types of techniques have been developed for visualizing and interpreting these feature representations of CNN. Below are some methods to visualize the learnings of CNN [12].

2.4.1 Visualizing Intermediate CNN Outputs (Intermediate Activations). For this sample image shown in Figure 3: from ImageDB2000 Dataset, the first few intermediate activations for the above image are shown in the Figure 4. The last few intermediate layer activations for Figure 3 are shown in the Figure 5.

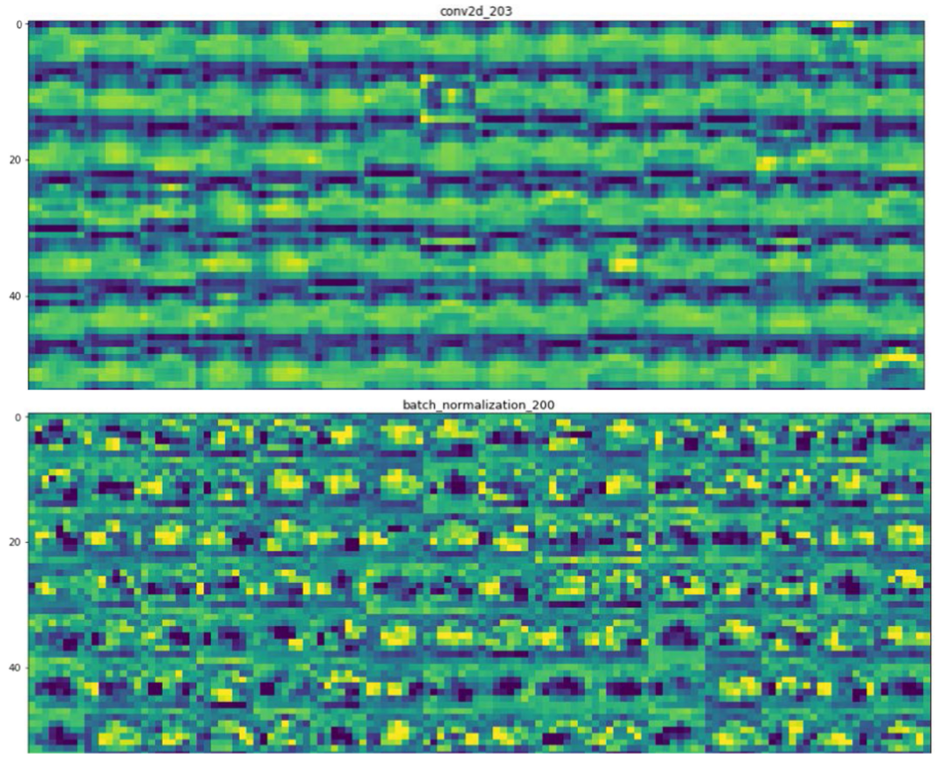

Fig. 5. Last few intermediate activations of the image shown in Figure 3.

There are a few things to note here:

(1) The first layers are basically the edge detectors. At this stage, the activations retain almost all of the information present in the picture fed in the network.
(2) As we go higher in the model, the activations outputs from each layer become increasingly abstract and less visually interpretable. They begin to encode higher-level features such as "fish fin" and "fish eye." Higher feature representations carry increasingly less information about the visual contents of the image, and increasingly more information related to the class of the image.
(3) The sparsity of the activations increases as we go deep in the layers of CNN.

2.4.2 Visualizing Intermediate Convnet Outputs (Adaptive Deconvolutional Networks). This model produces an over-complete image feature representation that can be used as input to standard neural network object classifiers. This model is learned from natural images and, given a new image, requires inference to compute. It decomposes an image in a hierarchical fashion using multiple alternating layers of convolutional sparse coding (deconvolution [47]) and max-pooling. Each of this deconvolution layers attempts to minimize the reconstruction error of the input image under a sparsity constraint on an over-complete set of feature maps. After doing so, for this sample image shown in Figure 6. Some of the shallow-level (low-level) convolution features are shown in Figure 7. And, some of the deep-level (high-level) convolution features are shown in Figure 8.

Here, we actually see that the model represents edge, texture type low-level features in the first layers, where in the last layers the model learns to represent some higher-level features. As an example, in Figure 8 the deep layers of the model represents fish fin, fish body types concepts.

Fig. 6. A sample image of Fish.

Fig. 7. Some low-level features of the image shown in Figure 6.

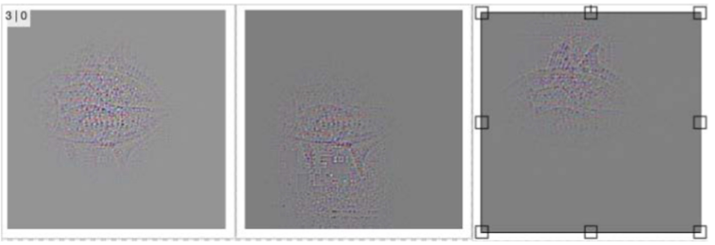

Fig. 8. Some of the deep-level features of the image shown in Figure 6.

Visualizing convent filters has been clearly described in chapter 5 of Chollet's book [12]. It has been shown that the filters in the earlier layers encode directional edges, colors and textures. Also, as we visualize the filters in the deeper layer, we find that the filters lend to learn textures found in natural images: feathers, eyes, leaves, and so on.

2.5 Proposed Method of Feature Extraction by Deep Learning

From the above section, it is evident that as the model architecture goes deep, it starts to learn high-level features from the low-level features. We propose to use these higher-level features for the feature representation of images in a CBIR system. So, we removed the last softmax activation layer used for calculating probabilities of each class and selected the outputs of the preceding fully connected layer to be our feature vector representation for CBIR. As this vector is the deepest layer of the model, this represents the most learned high-level features. We encoded (predicted) the images of our CBIR database through our pre-trained model and got an n-dimensional feature vector for each of the images. The flowchart of this process is shown in Figure 10 for better understanding. The value of n varies with the selection of deep-learning network architecture.

3 DETAILS OF THE PROPOSED ALGORITHM AND PERFORMANCE EVALUATION

3.1 Pre-trained Models Used

We have tried the following network architectures all pre-trained on the ImageNet dataset:

- DenseNet [16]
- InceptionResNetV2 [43]
- InceptionV3 [43]
- MobileNetV2 [35]
- NasNet Large [49]
- ResNet50 [15]
- VGG19 [41]
- Xception [11]

The main advantage of this method of feature extraction is that now we are able to extract higher level features without exploiting the CBIR database. This is necessary, because from the practical point of view, the CBIR dataset will be without any class information. The user will try to find similar images from this database based on the query image he/she has. Now as the CBIR dataset does not have any pre-defined class, training model on our dataset is not a feasible task unless we manually try to assign class to each of the images of our database dump consisting millions or billions of images, which is very time consuming and prone to subjective error for classifying images. To avoid this problem, we are using a Neural Network model pre-trained to classify a huge separate dataset (ImageNet) to perform feature extraction independently on our CBIR datasets [19, 21]. It is expected that the features derived to classify such a huge number of classes in an enormous dataset will be very effective for CBIR.

3.2 Database Used

The CBIR methods were applied on the following image databases, which vary in number as well as types of images.

- **ImageDB2000:** The database contains 2,000 images from 10 different categories each containing 200 images. The categories are Flowers, Fruits, Nature, Leaves, Ships, Faces, Fishes, Cars, Animals, and Aeroplanes [9].
- **ImageDBCaltech (Caltech101):** This database contains 9,144 images from 102 categories. The number of images in each category varies from 34 to 800 [23].
- **ImageDBCorel:** This dataset contains 1,000 images belonging to 10 categories. Each category contains 100 images. The categories are: African People, Beach, Building, Bus, Dinosaurs, Elephant, Flower, Horse, Mountain, and Food [31].

None of these datasets are parts of the ImageNet database.

3.3 Similarity Measure

The similarity (or dissimilarity) between a **query image (Q)** given by the user and a **database image (I)** stored in the system is measured by some distance metric. It is assumed that this distance will accurately measure the dissimilarity (or similarity) between the images as well. A smaller calculated distance implies more similarity. The similarity between two images can be different for different user's way of perceiving the images. Broadly speaking, there are mainly two types of similarity measures, geometric and probabilistic [32]. In the first case, the similarity is based on the distance between the feature vectors. A most widely used one is Minkowski, of which L1-norm and L2-norm are most popular. In probabilistic type, a Gaussian classifier is often used

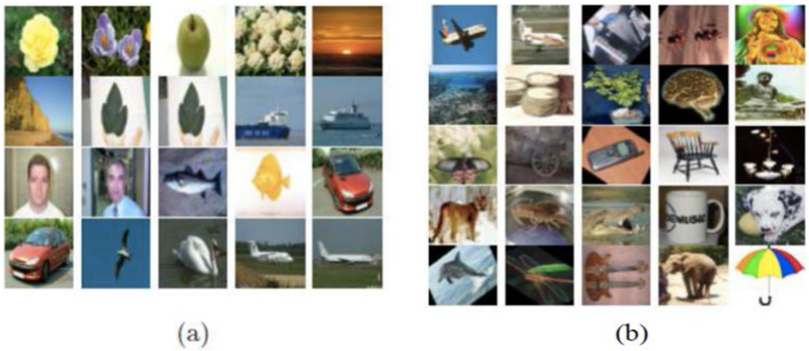

Fig. 9. Sample images from Dataset (a) ImageDB2000 and (b) DBCaltech.

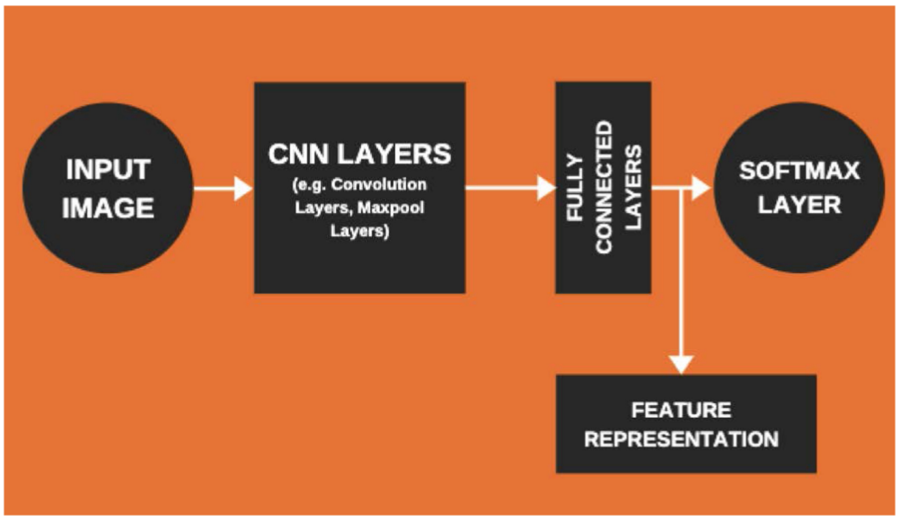

Fig. 10. CBIR image feature representation with the help of pre-trained deep-learning models.

to measure the relevance between the query image and the database image so that the pairs that had a high likelihood ratio were classified as relevant and the pairs having a low likelihood ratio is considered as irrelevant. Although past research [4] shows that the probabilistic methods perform significantly better than the geometric methods, they are computationally expensive. Throughout this article, we have used L1 or L2 norm as dissimilarity measure.

Manhattan Distance (L1 norm) between the extracted features of Q and I is formulated by

$$D(I,Q) = \sum_{i=1}^{n} |x_{i,I} - x_{i,Q}|, \tag{1}$$

and Euclidean Distance (L2 norm) between Q and I is given by

$$D(I,Q) = \sum_{i=1}^{n} (x_{i,I} - x_{i,Q})^2, \tag{2}$$

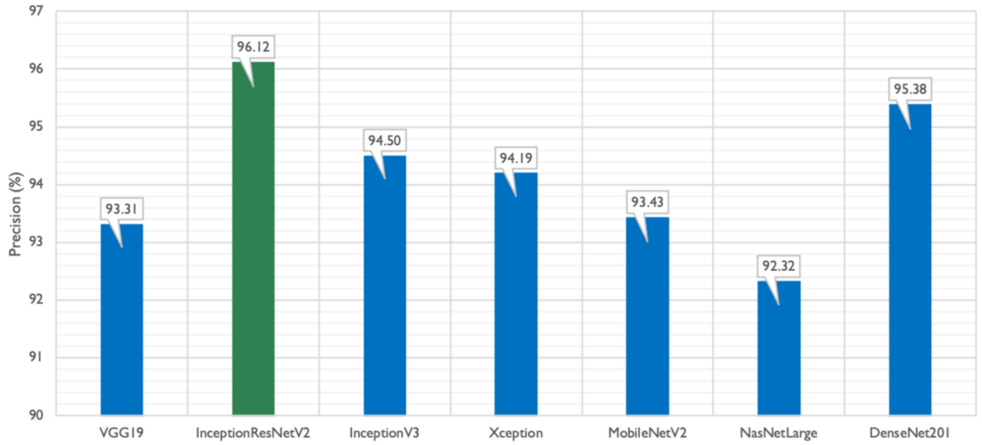

Fig. 11. Comparison of Deep-learning Architecture on Corel Dataset.

where n is the feature dimension of the images. $x(i, I)$ and $x(i, Q)$ are the ith feature value of database image and query image, respectively.

3.4 Evaluation of Performance

The most commonly used measures for evaluating the performance of a CBIR system is Precision, which is defined in Equation (3):

$$\texttt{Precision} = \frac{\texttt{Number of relevant images retrieved}}{\texttt{Number of retrieved images}}. \tag{3}$$

Generally, the number of images retrieved by any CBIR method is a pre-specified positive integer. This is called the scope of the system. Precision value is calculated for each image in the database, and these values are averaged over all images in the database. Usually, the greater the scope, the larger is the number of relevant images retrieved, typically leading to decreasing values of precision.

4 RESULTS

In this section, we present all the results, including selection of the best deep-learning network architecture, retrieval results of our CBIR system with respect to the sample query images taken from our datasets, category wise image retrieval precision for our datasets, and comparison of precision between the proposed method and some other recent ones on CBIR. The codes are available here: https://github.com/pidahbus/deepCBIR.

4.1 Selection of Best Deep-learning Network Architecture

We did a comparative study to select the best deep-learning architecture as described in Section 2.5 by calculating the average precision for each one of them for a scope value of 20 on DBCorel dataset. Euclidean Distance (L2 norm) is used as the dissimilarity metric. From the results given in Figure 11 it is clearly seen that InceptionResNetV2 is the winner with 96.115% average precision value. Therefore, we decided to use the InceptionResNetV2 network architecture for all the subsequent experiments.

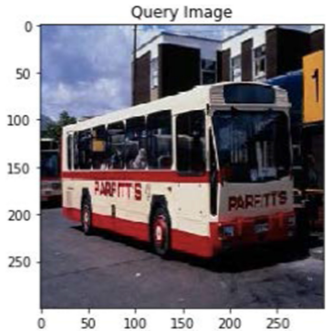

Fig. 12. A sample query image from DBCorel.

Fig. 13. Retrieved Results for the query image shown in Figure 12.

4.2 Image Retrieval of Sample Query Images

Using the InceptionResNetV2 architecture on the Corel Dataset for the scope of 20, for the example query image shown in Figure 12.

We retrieve 20 results shown in Figure 13. From Figure 13, we find that the query images belong to the "bus" category and all 20 results are relevant to the query image. So, the precision for this query image is 1.

For another query image from the corel dataset shown in Figure 14, the retrieved results are shown in Figure 15.

Here, we see that the query image belongs to the "African People" category and out of 20 retrieved results 13 results are relevant to the query image. So, for this specific image precision value is 13/20 = 0.65.

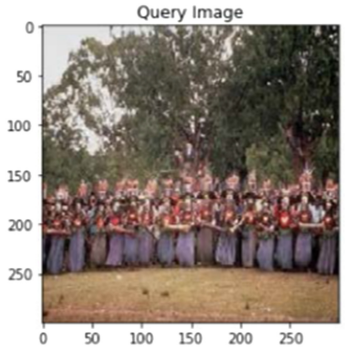

Fig. 14. A sample query image from DBCorel.

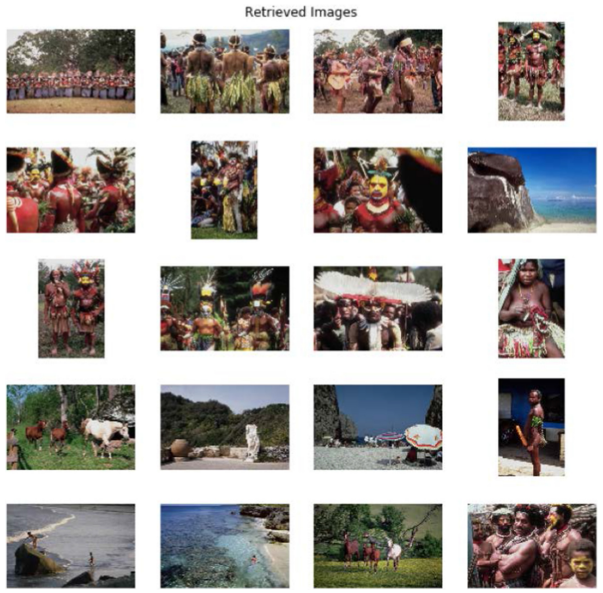

Fig. 15. Retrieved Results for the query image shown in Figure 14.

4.3 Category Wise Precision Calculation

In this subsection, we produce the category wise average precision on DBCorel and DB2000 for a scope of 20 using Euclidean Distance as dissimilarity metric. Figure 16 and Table 1 illustrate the results. From the result, we see that in DBCorel, for Bus, Dinosaurs and Elephant category the pre-trained InceptionResNetV2 model retrieves all the images with 100% precision but performs comparatively poorly for the African People category resulting in 79.35% precision. The overall average precision for this dataset is 96.115%. For DB2000 the best retrieved category is Airplane (precision: 99.975%) and worst category is Leaf (precision: 91.125%), overall average precision being 97%.

4.4 Result Comparison with Other Recently Proposed Algorithms

For DB2000, we select Bose et al.'s paper [9] as the baseline result. Their paper [9] extracted features from the images in two ways: features from colour co-occurrence matrix and features from

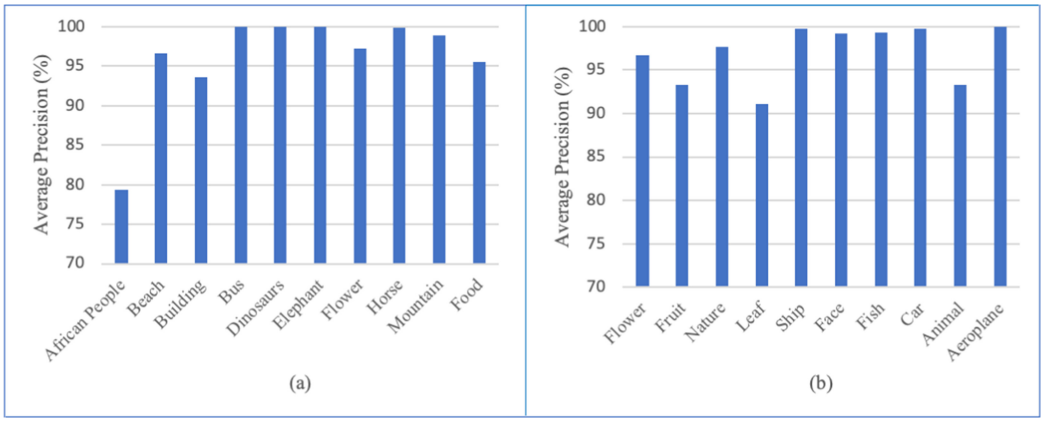

Fig. 16. Category-wise average precision for scope of 20 on (a) DBCorel (b) DB2000.

Table 1. Category-wise Average Precision for Scope of 20 on (a) DBCorel (b) DB2000

Categories	Average Precision(%)
African People	79.35
Beach	96.6
Building	93.55
Bus	100
Dinosaurs	100
Elephant	100
Flower	97.25
Horse	99.9
Mountain	98.95
Food	95.55

(a)

Categories	Average Precision(%)
Flower	96.65
Fruit	93.25
Nature	97.675
Leaf	91.125
Ship	99.8
Face	99.275
Fish	99.3
Car	99.725
Animal	93.3
Aeroplane	99.975

(b)

MPEG-7. Since in this work Relevance Feedback [9] is not applied, we are only comparing the precision without relevance feedback of that paper [9] with ours. Figure 17 shows that our proposed method outperforms all the methods discussed in Reference [9].

In recent years, many researchers have worked [2, 3, 6–8, 21, 24, 27, 34, 36, 39, 46] on DBCorel Dataset extracting different kinds of features and similarity distances. We present two types of comparison with these recent papers on DBCorel Dataset: Category wise precision comparison (Figure 18) and average precision comparison (Table 2). It shows that our proposed method outperforms all other methods published in the recent papers. Lohite et al. [24] calculated category wise precision for scope of 50 instead of 20. We did a comparative study with this algorithm too in Figure 19 and showed that except African People category our proposed method works better even for scope 50.

For DBCaltech (Caltech101) Dataset, we produce the comparison with two algorithms given in References [9, 33]. Figure 20 compares the average precision. Multiple CBIR techniques are described in References [9, 33]. Hence, we picked the average precision of the best methods. Clearly the proposed method outperformed both of the other methods.

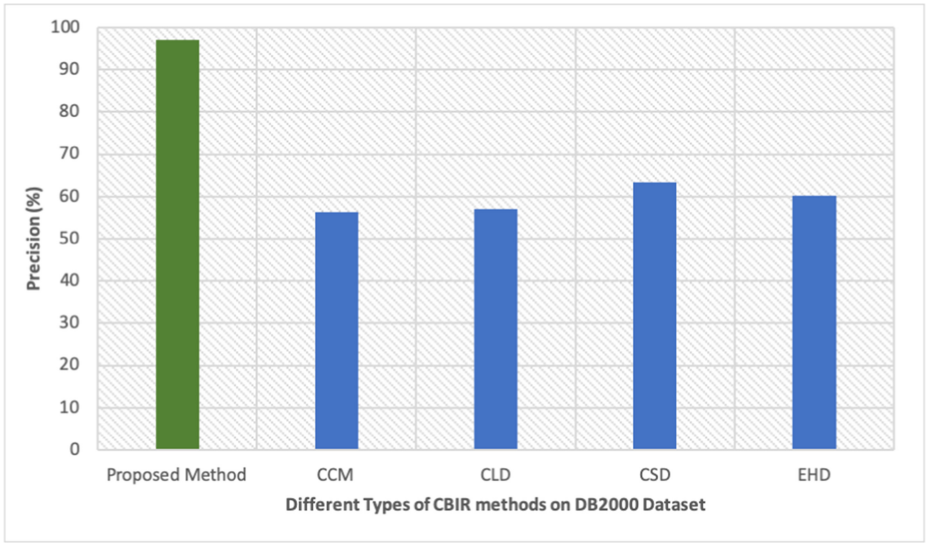

Fig. 17. Comparison of average precision between our proposed method and Bose et al. [9] methods for scope of 20 on DB2000.

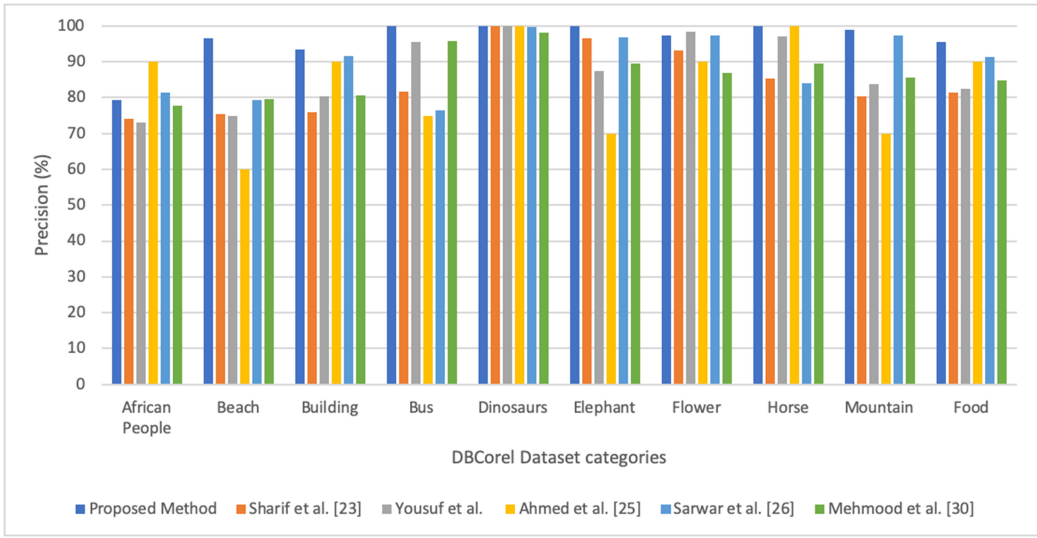

Fig. 18. Comparison of category wise precision between our proposed method and recent papers methods for scope of 20 on DBCorel.

4.5 Effect of Illumination/Brightness on CBIR

In this section, we have analyzed the effect of brightness on our CBIR method. To define the brightness of an image we have used a brightness factor i, that means new image is i times brighter then an original image. The brightness factor of original image is 1. Figure 21 shows the visual illustration of a sample image for different values of brightness factors.

Table 2. Comparison of Average Precision between Our Proposed Method and Recent Papers' Methods for Scope of 20 on DBCorel

Methods	Average Precision (%)
Proposed Method	**96.12**
Ashraf et al. [6]	73.50
Sharif et al. [39]	84.39
Yousuf et al. [46]	87.30
Ahmed et al. [2]	83.50
Sarwar et al. [36]	89.58
Ahmed et al. [3]	76.50
Ashraf et al. [7]	82.00
Rashno et al. [34]	65.95
Mehmood et al. [27]	87.85
Khokhar et al. [21]	94.30
Ahamed et al. [8]	82.00
Yosr et al. [45]	88.65
Singh et al. [42]	92.00

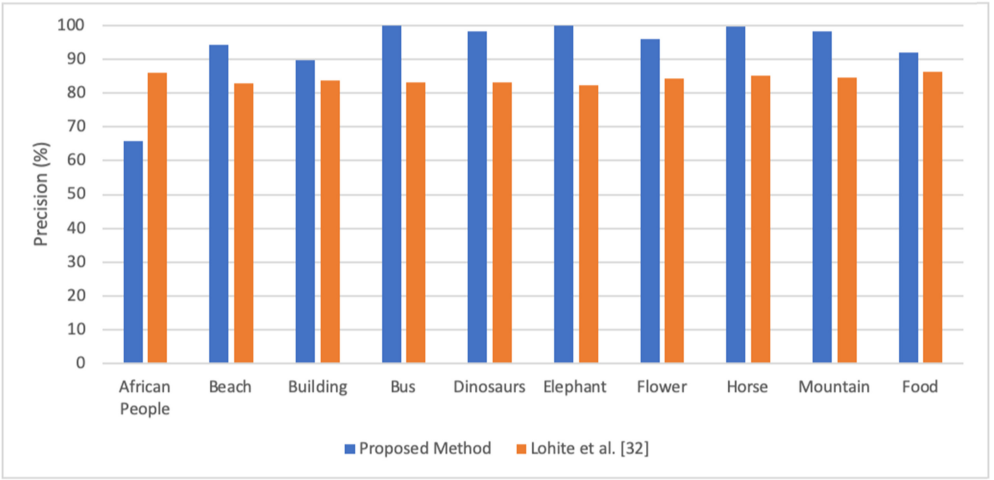

Fig. 19. Comparison of category wise precision between our proposed method and Lohite et al. [24] methods for scope of 50 on DBCorel. Average precision of our proposed method: 93.39%. Average precision of Lohite et al. [24]: 84.226%.

To analyze the effect of the brightness factor, we have performed two experiments. In Experiment 1, we have calculated the average precision for different values of brightness factor on DBCorel [31], DB2000 [9], and DBCaltech [23]. For every query the brightness factors of database images and the query image were the same. However, in Experiment 2, we performed a similar exercise, using different values of the brightness factor for the query image without changing the brightness of the database images. We report class-wise average precision values for different values of brightness factor on DBCorel. Tables 3 and 4 show the class-wise precision for different values of brightness factor for Experiments 1 and 2, respectively, on DBCorel. Figures 22 and 23

Table 3. Class-wise Average Precision for Different Values of Brightness Factor on the Query Image of DBCorel for Experiment 1

Categories	Brightness Factor												
	0	**0.05**	**0.1**	**0.15**	**0.2**	**0.25**	**0.5**	**1**	**2**	**4**	**6**	**8**	**10**
African People	5.00	61.00	70.90	75.40	75.05	74.80	76.50	79.35	76.10	76.00	74.25	70.40	67.00
Beach	0.00	64.60	83.90	89.60	92.25	93.45	96.10	96.60	95.45	83.80	77.80	71.35	64.90
Building	0.00	86.65	91.50	93.15	94.40	94.30	94.60	93.55	93.60	90.15	83.40	77.95	73.35
Bus	0.00	99.85	100.00	100.00	100.00	100.00	100.00	100.00	100.00	99.95	99.90	99.90	99.70
Dinosaurs	0.00	97.90	99.85	99.95	99.95	100.00	100.00	100.00	99.70	98.00	97.75	96.25	94.00
Elephant	0.00	97.35	100.00	100.00	100.00	100.00	100.00	100.00	100.00	95.40	82.05	69.80	63.00
Flower	95.00	96.00	97.00	96.30	97.50	97.50	97.65	97.25	98.05	99.40	97.25	96.95	95.15
Horse	0.00	93.60	99.80	99.85	99.95	100.00	100.00	99.9	99.80	96.05	89.50	82.00	77.20
Mountain	0.00	87.85	96.55	98.00	98.80	99.05	99.00	98.95	98.45	85.55	78.10	68.25	62.10
Food	3.20	71.95	88.00	91.50	92.95	93.80	94.90	95.55	95.25	92.75	90.25	86.75	83.40
Average	**10.32**	**85.68**	**92.75**	**94.38**	**95.08**	**95.29**	**95.88**	**96.12**	**95.64**	**91.70**	**87.03**	**81.96**	**77.97**

The precision values are calculated for the scope of 20.

Table 4. Class-wise Average Precision for Different Values of Brightness Factor on the Query Image of DBCorel for Experiment 2

Categories	Brightness Factor												
	0	**0.05**	**0.1**	**0.15**	**0.2**	**0.25**	**0.5**	**1**	**2**	**4**	**6**	**8**	**10**
African People	0.10	69.90	77.70	78.45	78.50	78.55	78.50	79.35	76.00	65.75	57.70	55.15	55.55
Beach	0.00	61.70	94.80	96.25	96.95	97.20	97.20	96.60	96.55	88.10	88.75	88.25	82.30
Building	0.00	90.20	92.40	93.05	94.05	94.40	94.45	93.55	93.80	92.10	91.00	91.05	90.95
Bus	0.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00
Dinosaurs	0.00	99.85	100.00	100.00	100.00	100.00	100.00	100.00	99.40	92.70	91.40	89.70	88.15
Elephant	0.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	100.00	92.30	78.70
Flower	67.00	85.40	95.85	96.45	97.00	97.25	97.25	97.25	96.50	96.15	95.95	95.50	94.85
Horse	0.00	99.85	100.00	99.95	99.95	100.00	100.00	99.90	99.75	98.80	98.60	93.15	85.40
Mountain	0.00	97.75	97.90	98.05	98.45	98.85	98.85	98.95	99.10	99.15	98.10	97.60	93.05
Food	36.40	76.25	89.95	92.65	93.55	94.05	94.55	95.55	94.55	92.20	89.70	86.20	81.05
Average	**10.35**	**88.09**	**94.86**	**95.49**	**95.85**	**96.03**	**96.08**	**96.12**	**95.57**	**92.50**	**91.12**	**88.89**	**85.00**

The precision values are calculated for the scope of 20.

show overall precision for the two experiments on each of DBCorel, DB2000 and DBCaltech. From the results it can be clearly seen that our method is quite robust for certain range of illumination, which is evident from the stability of the precision.

5 REAL-TIME CBIR

It is evident from the previous section that due to introduction of very deep neural network model (InceptionResNetV2), the retrieval results improved by a significant amount, but naturally the retrieval time of images becomes a matter of concern. Whatever models are used, the ultimate goal is to retrieve images in real time. In this section, we discuss about the time complexity of the proposed CBIR system and will show that in spite of introducing deep models, the proposed system can retrieve images in real time by an illustration with the DBCaltech dataset. The image retrieval time for the other two datasets DB2000 and DBCorel, will be anyway quite low.

Further by introducing the application of Principal Component Analysis [40] it appears that the image retrieval can be made even faster without sacrificing the precision. This idea has been presented in Section 5.1 below.

5.1 Principal Component Analysis

Principal Component Analysis (PCA) [40], is a dimensionality reduction method generally used to reduce the dimensionality of large data-sets, by converting large set of variables into

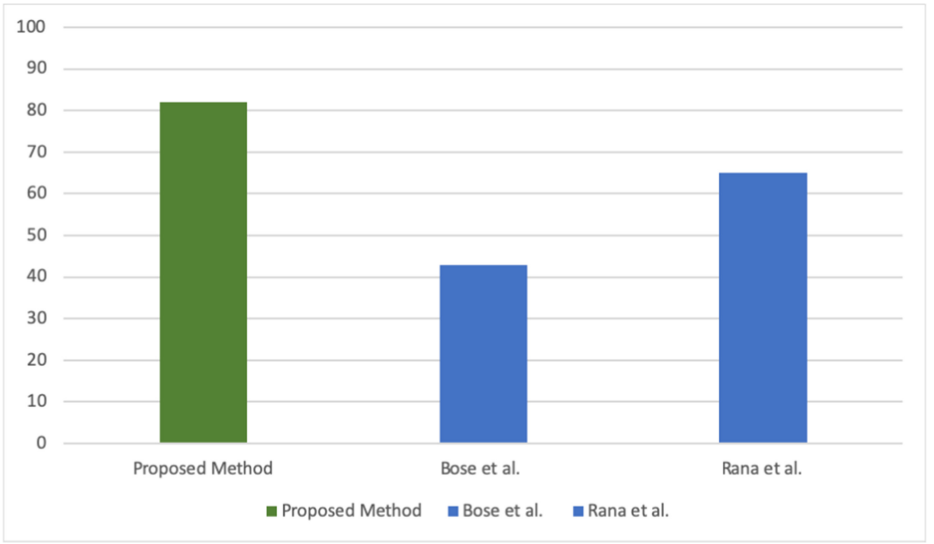

Fig. 20. Comparison of average precision among our proposed method, Bose et al. [9] and Rana et al. [33] methods for scope of 20 on DBCaltech.

Fig. 21. A sample image from DBCorel with different values of brightness factor. The brightness factor of original image is 1.

smaller ones, which contains most of the information of the original dataset. PCA is a technique to reduce the number of variables of a data set, while preserving as much information as possible.

Reducing the number of variables of a data set results in loss of information contained in it, but the idea in dimensionality reduction is to trade a little information for simplicity. The reason behind is that smaller data sets are easier to handle and visualize. Further, analyzing data becomes much easier and faster for machine learning algorithms in the derived smaller datasets.

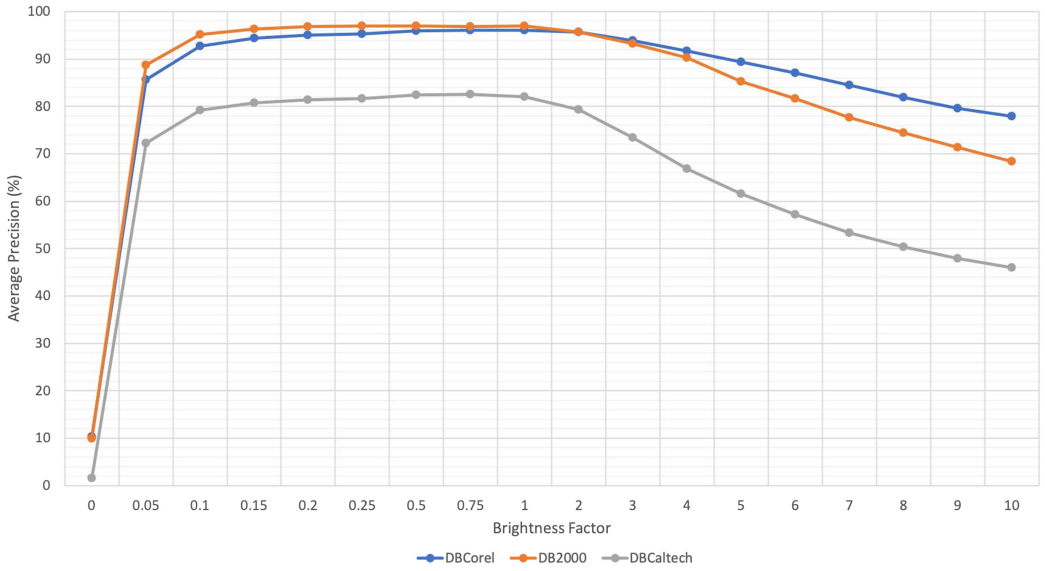

Fig. 22. Result of Experiment 1 (where the brightness factor of the query and database images were the same) on DBCorel, DB2000, and DBCaltech. The precision values are calculated for scope of 20.

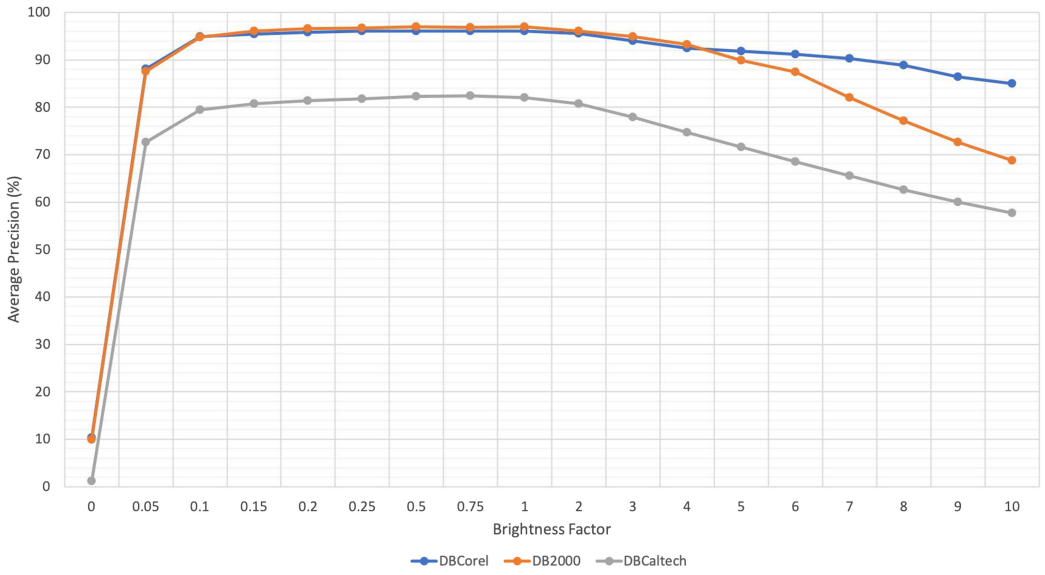

Fig. 23. Result of Experiment 2 (where the brightness factor of the query image was varied) on DBCorel, DB2000, and DBCaltech. The precision values are calculated for scope of 20.

5.2 PCA on the Encoded Features

The encoded feature vector dimension for InceptionResNetV2 is 1,536, which is on the larger side. Therefore, Principal Component Analysis was employed on the 1,536 feature vector to reduce its dimension and the number (M) of principal components (which are linear combinations of the original features) was chosen for which the average precision value is maximum. For DBCaltech

dataset roughly 100 PCs were used to calculate the precision. It is seen that taking the first handful number of PCs results in almost the same or sometime better average precision as compared to the whole set of 1,536 features. For example, the average precision with and without PCA were 82.54% and 82.02%, respectively, in the DBCaltech dataset. In summary PCA tried to preserve the precision level while reducing computational time by reducing the dimension.

The following procedure was adopted for incorporating PCA. Features (derived from Inception-ResNetV2 without the last softmax layer) were extracted from all images in the database, and a PCA was performed. These principal components were stored and used to calculate modified features for each of the image in the database.These modified features were then stored as a feature bank.

For a new query image, similarly the initial features (using InceptionResNetV2) were derived, which were then modified using the stored principal components. These modified features of the query image were then compared with features of every image stored in the database using the similarity measure. Finally those images whose features are closer to the query image features were retrieved.

5.3 Comparing CBIR with and without PCA

As discussed in the previous section, the difference between the proposed system without or with PCA is that both follow the same algorithm but in the latter case the features go through a transformation using PCA before comparison. Hence the image retrieval time is the time between the feeding of query image and retrieving similar images and Figure 24 shows this average image retrieval time. We use the term "average," because each of the images in the database was used as query image and their average retrieval time is reported. This process is tested on two machines:

The local machine

- 1.8 GHz Intel Core i5 processor
- 8 GB LPDDR3 RAM

GPU Machine

- GPU: 1 NVIDIA Pascal GPU
- CUDA Cores: 2,048
- Memory Size: 16 GB GDDR5
- H.264 1080p30 streams: 24
- Max vGPU instances: 16 (1 GB Profile)
- vGPU Profiles: 1 GB, 2 GB, 4 GB, 8 GB, 16 GB
- Form Factor: MXM (blade servers)
- Power: 90 W (70 W opt)
- Thermal: Bare Board

Since GPUs are highly specialized in parallel computing, the time required for image retrieval is very little as compared to the local machine. This can be clearly seen in Figure 24. Also it is evident that the use of PCA has in fact reduced the retrieval time marginally.

DBCaltech has 9,144 images with 1,536 dimensional features (without PCA) and DB2000 has 2,000 images with 1,536 dimensional features. We can see that our GPU machine and somewhat our local machine also can retrieve images from these dataset in real time. Image retrieval time depends on both the Database size and dimension. Dimension of the feature vector will be the same if we use the same architecture (InceptionResNetV2 in our case). However for a dataset having a very high number of images (say millions) the image retrieval time increases rapidly as the system searches through the whole database for relevant images. In that case, perhaps we could use a

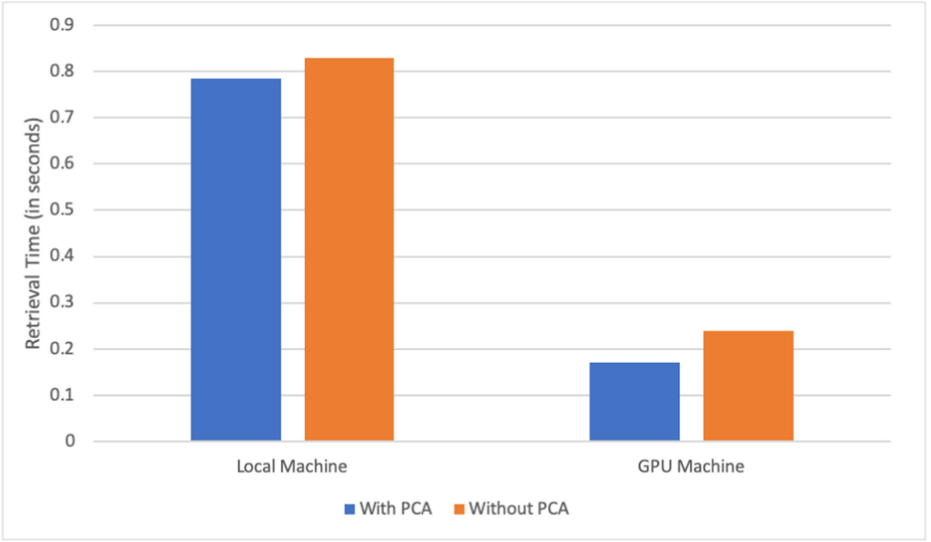

Fig. 24. Tested average image retrieval time on DBCaltech is shown in the figure. The standard deviation of the image retrieval time is about 0.01 s.

random sample of a manageable size (say 10,000 or 20,000) and retrieve images from it instead from the whole database.

6 FAST IMAGE RETRIEVAL WITH IMAGE CLUSTERING

As mentioned above that as the size of the database increases, the image retrieval time also increases. Here, we propose a novel method to further improve the image retrieval time. This method does clustering of the database images and then given a query image, only searches for similar images within an appropriate cluster. The method is described further below.

6.1 The Proposed Method

The pre-trained model, InceptionResNetV2 that we have used for the CBIR method earlier was originally trained to predict 1,000 classes. There, we ignored the last softmax layer and chose the last dense layer for feature extraction. Now, we propose to use both the last dense layer output as features and the softmax layer probability output for assigning an image to different clusters. The method is explained step by step below. This method has been applied on the Caltech Dataset.

(1) First, we calculate for each of the images in the database the last dense layer features and also the probabilities that they belong to the 1,000 pre-specified classes from the last softmax layer.
(2) Now according to the probabilities, we assign each of the images to the top five classes. For example, let us say Image_2.jpg has the probability to be assigned to class 2 with 0.4 probability, class 78 with 0.2 probability, class 9 with 0.15 probability, class 324 with 0.1 probability, class 639 with 0.05 probability, and so on, with decreasing probabilities, then we assign Image_2.jpg to class 2, class 78, class 9, class 324, and class 639. We also store the last dense layer feature values for every image in the database in memory. Therefore, every image in the database are assigned to five different clusters.
(3) Similarly at the time of retrieval, we calculate both the last dense layer feature values and the class probabilities for the query image as well and assign the query image to top five classes

Fig. 25. Comparison of tested average image retrieval time on DBCaltech between proposed Fast Retrieval method and previous method. The standard deviation of the image retrieval time is about 0.01 s.

based on the probabilities. Let us say the assigned classes for the query image query_1.jpg are class 11, class 258, class 750, class 54, and class 23.

(4) We accumulate all the images belonging to these classes and calculate similarity measure with the last dense layer features and retrieve similar images only from the accumulated images.

(5) This gives us a much faster retrieval with respect to the previous method as now we are searching relevant images in only a small subset of the whole database instead of the whole database containing 9,144 images. For searching in the top five classes the average number of images to be searched for any image retrieval becomes only 229.

(6) The reduction in the image retrieval time is shown in Figure 25. This experiment has been tested for with and without PCA on both Local and GPU Machine. We define the image retrieval time of an image to be the time between feeding the query image to the system to retrieving the similar images. This process is repeated by treating each of the images of the database as query image. We finally calculate the mean image retrieval time for all these images.

Note: The precision for this fast retrieval method comes down a little to 81.48% from the previous value of 82.02%. Therefore, the precision value is not sacrificed much while the image retrieval time comes down by about 2.5 times. The reason behind the success in preserving almost the same precision value is that, the InceptionResNetV2 model predicts similar images to the same classes, so the clusters of similar images are formed in the classes.

7 CONCLUSION

This article shows that using pre-trained deep-learning features, one can achieve better precision results compared to the features derived by traditional methods using CCM, wavelets, and others. We can improve the retrieval time significantly by using PCA and a clever clustering approach using the same pre-trained network without sacrificing the precision value.

Another approach to improve retrieval performance for a specific dataset is to incorporate a method called Relevance Feedback. Relevance Feedback is basically the obtaining of information from the user after each retrieval regarding which images are relevant to the query images and which are not. Using this feedback, the CBIR system will start learning and will improve the result gradually. We plan to incorporate this feature in the proposed CBIR system in the future.

There is another limitation of features derived by deep learning, and that is that these features are not rotation-invariant. This means that if we try to retrieve similar images given the same query image but with different orientation angles, then the retrieval results may change significantly. Building a rotation-invariant CBIR system may be another improvement that will be investigated in the future as well.

ACKNOWLEDGMENTS

We acknowledge two anonymous referees whose valuable comments made significant improvements to the article.

REFERENCES

[1] Stanford University. [n.d.]. *CS231n: Convolutional Neural Networks for Visual Recognition*. Retrieved from http://cs231n.stanford.edu/.

[2] Khawaja Ahmed, Shahida, and Muhammad Iqbal. 2018. Content-based image retrieval using image features information fusion. *Info. Fusion* 51 (Nov. 2018), 76–99. https://doi.org/10.1016/j.inffus.2018.11.004

[3] K. T. Ahmed, S. A. H. Naqvi, A. Rehman, and T. Saba. 2019. Convolution, approximation and spatial information based object and color signatures for content based image retrieval. In *Proceedings of the International Conference on Computer and Information Sciences (ICCIS'19)*. 1–6. https://doi.org/10.1109/ICCISci.2019.8716437

[4] S. Aksoy and R. M. Haralick. 2000. Probabilistic vs. geometric similarity measures for image retrieval. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR'00)*, Vol. 2. 357–362. https://doi.org/10.1109/CVPR.2000.854847

[5] Mutasem K. Alsmadi. 2020. Content-based image retrieval using color, shape and texture descriptors and features. *Arab. J. Sci. Eng.* 45, 4 (2020), 3317–3330. https://doi.org/10.1007/s13369-020-04384-y

[6] Rehan Ashraf, Mudassar Ahmed, Sohail Jabbar, Shehzad Khalid, Awais Ahmad, Sadia Din, and Gwangil Jeon. 2018. Content based image retrieval by using color descriptor and discrete wavelet transform. *J. Med. Syst.* 17, 6 (Mar. 2018), 3552–3580. https://doi.org/10.1007/s10916-017-0880-7

[7] Rehan Ashraf, Khalid Bashir, Aun Irtaza, and Muhammad Mahmood. 2015. Content based image retrieval using embedded neural networks with bandletized regions. *Entropy* 17 (June 2015), 3552–3580. https://doi.org/10.3390/e17063552

[8] Mohamed Uvaze Ahamed Ayoobkhan, C. Eswaran, and Kannan Ramakrishnan. 2017. CBIR system based on prediction errors. *J. Info. Sci. Eng.* 33 (Mar. 2017), 347–365. https://doi.org/10.1688/JISE.2017.33.2.5

[9] Smarajit Bose, Amita Pal, Disha Chakrabarti, and Taranga Mukherjee. 2017. Improved content-based image retrieval via discriminant analysis. *Int. J. Mach. Learn. Comput.* 7 (June 2017), 44–48. https://doi.org/10.18178/ijmlc.2017.7.3.618

[10] Alfredo Canziani, Adam Paszke, and Eugenio Culurciello. 2016. An analysis of deep neural network models for practical applications. Retrieved from https://arXiv:1605.07678.

[11] François Chollet. 2016. Xception: Deep learning with depthwise separable convolutions. Retrieved from https://arXiv:1610.02357.

[12] Francois Chollet. 2017. *Deep Learning with Python* (1st ed.). Manning Publications, Greenwich, CT.

[13] J. Deng, W. Dong, R. Socher, L. Li, Kai Li, and Li Fei-Fei. 2009. ImageNet: A large-scale hierarchical image database. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 248–255. https://doi.org/10.1109/CVPR.2009.5206848

[14] Meenakshi Garg and Gaurav Dhiman. 2021. A novel content-based image retrieval approach for classification using GLCM features and texture fused LBP variants. *Neural Comput. Appl.* 33 (2021), 1311–1328. https://doi.org/10.1007/s00521-020-05017-z

[15] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2015. Deep residual learning for image recognition. Retrieved from https://arXiv:1512.03385.

[16] Gao Huang, Zhuang Liu, Laurens van der Maaten, and Kilian Q. Weinberger. 2016. Densely connected convolutional networks. Retrieved from https://arXiv:1608.06993.

[17] Jing Huang. 1998. *Color-spatial Image Indexing and Applications*. Ph.D. Dissertation. Ithaca, NY. Advisor(s) Zabih, Ramin.

[18] D. Mansoor Hussain and D. Surendran. 2020. The efficient fast-response content-based image retrieval using spark and MapReduce model framework. *J. Ambient Intell. Human. Comput.* 12, 3 (2020), 4049–4056. https://doi.org/10.1007/s12652-020-01775-9

[19] Safia Jabeen, Zahid Mehmood, Toqeer Mahmood, Tanzila Saba, Amjad Rehman, and Muhammad Mahmood. 2018. An effective content-based image retrieval technique for image visuals representation based on the bag-of-visual-words model. *PLoS ONE* 13, 4 (Mar. 2018). https://doi.org/10.1371/journal.pone.0194526

[20] Aamir Khan and Anand Jalal. 2021. A visual saliency-based approach for content-based image retrieval. *International J. Cogn. Info. Natural Intell.* 15 (Jan. 2021), 1–15. https://doi.org/10.4018/IJCINI.2021010101

[21] Suman Khokhar and Satya Verma. 2017. Content based image retrieval with multi-feature classification by back-propagation neural network. *Int. J. Comput. Appl. Technol. Res.* 6 (July 2017), 278–284. https://doi.org/10.7753/IJCATR0607.1002

[22] Harald Kosch. 2003. *Distributed Multimedia Database Technologies Supported by MPEG-7 and MPEG-21.* CRC Press.

[23] Li Fei-Fei, R. Fergus, and P. Perona. 2004. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. In *Proceedings of the Conference on Computer Vision and Pattern Recognition Workshop.* 178–178. https://doi.org/10.1109/CVPR.2004.383

[24] Mr. Yogen Mahesh Lohite and Prof. Sushant J. Pawar. 2017. A novel method for content based image retrieval using local features and SVM classifier. *Int. Res. J. Eng. Technol.* 4, 7 (2017).

[25] B. S. Manjunath, J. Ohm, V. V. Vasudevan, and A. Yamada. 2001. Color and texture descriptors. *IEEE Trans. Circ. Syst. Video Technol.* 11, 6 (June 2001), 703–715. https://doi.org/10.1109/76.927424

[26] Pedro Marcelino. [n.d.]. *Transfer learning from pre-trained models.* Retrieved from https://towardsdatascience.com/transfer-learning-from-pre-trained-models-f2393f124751.

[27] Zahid Mehmood, Toqeer Mahmood, and Muhammad Arshad Javid. 2018. Content-based image retrieval and semantic automatic image annotation based on the weighted average of triangular histograms using support vector machine. *Appl. Intell.* 48, 1 (Jan. 2018), 166–181. https://doi.org/10.1007/s10489-017-0957-5

[28] Wayne Niblack, Ronald Barber, William Equitz, Myron Flickner, Eduardo Glasman, Dragutin Petkovic, Peter Yanker, Christos Faloutsos, and Gabriel Taubin. 1993. The QBIC Project: Querying images by content, using color, texture, and shape. In *Proceedings of the SPIE Conference on Storage and Retrieval for Image and Video Databases.* 173–187.

[29] A. Obulesu, Vakulabharanam Vijaya Kumar, and Sumalatha Lingamgunta. 2018. Content based image retrieval using multi motif co-occurrence matrix. *Int. J. Image Graph. Signal Process.* 10 (Apr. 2018), 59–72. https://doi.org/10.5815/ijigsp.2018.04.07

[30] T. Ojala, Mika Rautiainen, Esa Matinmikko, and M. Aittola. 2001. Semantic image retrieval with HSV correlograms. (Jan. 2001).

[31] Michael Ortega-Binderberger. [n.d.]. *Corel Image Features Data Set.* Retrieved from https://archive.ics.uci.edu/ml/datasets/corel+image+features.

[32] Jing Peng, Bir Bhanu, and Shan Qing. 1999. Probabilistic feature relevance learning for content-based image retrieval. *Comput. Vision Image Understand.* 75 (1999), 150–164.

[33] Soumya Rana, Maitreyee Dey, and Siarry Patrick. 2018. Boosting content based image retrieval performance through integration of parametric & nonparametric approaches. *J. Visual Commun. Image Represent.* 58 (Nov. 2018), 205–219. https://doi.org/10.1016/j.jvcir.2018.11.015

[34] A. Rashno and S. Sadri. 2017. Content-based image retrieval with color and texture features in neutrosophic domain. In *Proceedings of the 3rd International Conference on Pattern Recognition and Image Analysis (IPRIA'17).* 50–55. https://doi.org/10.1109/PRIA.2017.7983063

[35] Mark Sandler, Andrew Howard, Menglong Zhu, Andrey Zhmoginov, and Liang-Chieh Chen. 2018. MobileNetV2: Inverted residuals and linear bottlenecks. Retrieved from https://arXiv:1801.04381.

[36] Amna Sarwar, Zahid Mehmood, Tanzila Saba, Khurram Ashfaq Qazi, Ahmed Adnan, and Habibullah Jamal. 2019. A novel method for content-based image retrieval to improve the effectiveness of the bag-of-words model using a support vector machine. *J. Info. Sci.* 45, 1 (2019), 117–135. https://doi.org/10.1177/0165551518782825 arXiv:https://doi.org/10.1177/0165551518782825

[37] G. V. Satya Kumar and P. G. Krishna Mohan. 2018. Local mean differential excitation pattern for content based image retrieval. *SN Appl. Sci.* 1, 1 (Nov. 2018), 46. https://doi.org/10.1007/s42452-018-0047-2

[38] Seong-O Shim and Tae-Sun Choi. 2003. Image indexing by modified color co-occurrence matrix. In *Proceedings of the International Conference on Image Processing*, Vol. 3. III–493. https://doi.org/10.1109/ICIP.2003.1247289

[39] Uzma Sharif, Zahid Mehmood, Toqeer Mahmood, Dr. Javid, Amjad Rehman, and Tanzila Saba. 2018. Scene analysis and search using local features and support vector machine for effective content-based image retrieval. *Artific. Intell. Rev.* 52, 2 (June 2018), 901–925. https://doi.org/10.1007/s10462-018-9636-0

[40] Jonathon Shlens. 2014. A tutorial on principal component analysis. Retrieved from https://arXiv:1404.1100.

[41] Karen Simonyan and Andrew Zisserman. 2014. Very deep convolutional networks for large-scale image recognition. Retrieved from https://arXiv:1409.1556.

[42] Sachendra Singh and Shalini Batra. 2020. An efficient bi-layer content based image retrieval system. *Multimedia Tools Appl.* 79, 25 (July 2020), 17731–17759. https://doi.org/10.1007/s11042-019-08401-7

[43] Christian Szegedy, Sergey Ioffe, Vincent Vanhoucke, and Alex Alemi. 2016. Inception-v4, inception-resnet and the impact of residual connections on learning. Retrieved from https://arXiv:1602.07261.

[44] Jian Xu, Cunzhao Shi, Chengzuo Qi, Chunheng Wang, and Baihua Xiao. 2017. Unsupervised part-based weighting aggregation of deep convolutional features for image retrieval. Retrieved from https://arXiv:1705.01247.

[45] G. Yosr, N. Baklouti, H. Hagras, M. Ben ayed, and A. M. Alimi. 2021. Interval Type-2 beta fuzzy near sets approach to content-based image retrieval. *IEEE Trans. Fuzzy Syst.* (2021), 1–1. https://doi.org/10.1109/TFUZZ.2021.3049900

[46] Muhammad Yousuf, Zahid Mehmood, Hafiz Adnan Habib, Toqeer Mahmood, Tanzila Saba, Amjad Rehman, and Muhammad Rashid. 2018. A novel technique based on visual words fusion analysis of sparse features for effective content-based image retrieval. *Math. Problems Eng.* 2018 (Mar. 2018), 13. https://doi.org/10.1155/2018/2134395

[47] M. D. Zeiler, G. W. Taylor, and R. Fergus. 2011. Adaptive deconvolutional networks for mid- and high-level feature learning. In *Proceedings of the International Conference on Computer Vision.* 2018–2025. https://doi.org/10.1109/ICCV.2011.6126474

[48] Wengang Zhou, Houqiang Li, and Qi Tian. 2017. Recent advance in content-based image retrieval: A literature survey. Retrieved from https://arXiv:1706.06064.

[49] Barret Zoph, Vijay Vasudevan, Jonathon Shlens, and Quoc V. Le. 2017. Learning transferable architectures for scalable image recognition. Retrieved from https://arXiv:1707.07012.

Received October 2020; revised March 2021; accepted June 2021