Efficient Conversion of Deep Features to Compact Binary Codes Using Fourier Decomposition for Multimedia Big Data

Jamil Ahmad, *Student Member, IEEE*, Khan Muhammad, *Student Member, IEEE*, Jaime Lloret, *Senior Member, IEEE*, and Sung Wook Baik, *Member, IEEE*

***Abstract*—Exponential growth of multimedia data has been witnessed in recent years from various industries, such as e-commerce, health, transportation, and social networks, etc. Access to desired data in such gigantic datasets require sophisticated and efficient retrieval methods. In the last few years, neuronal activations generated by a pretrained convolutional neural network (CNN) have served as generic descriptors for various tasks including image classification, object detection and segmentation, and image retrieval. They perform incredibly well compared to handcrafted features. However, these features are usually high dimensional, requiring a lot of memory and computations for indexing and retrieval. For very large datasets, utilization of these high dimensional features in raw form becomes infeasible. In this paper, a highly efficient method is proposed to transform high dimensional deep features into compact binary codes using bidirectional Fourier decomposition. This compact bit code saves memory and eases computations during retrieval. Further, these codes can also serve as hash codes, allowing very efficient access to images in large datasets using approximate nearest neighbor (ANN) search techniques. Our method does not require any training and achieves considerable retrieval accuracy with short length codes. It has been tested on features extracted from fully connected layers of a pretrained CNN. Experiments conducted with several large datasets reveal the effectiveness of our approach for a wide variety of datasets.**

***Index Terms*—Deep learning, Fourier transform, hash codes, image retrieval, industrial informatics.**

I. Introduction

BIG data has recently emerged as a key concept, denoting the gigantic volume of data generated at a rapid pace due to the progress in sensing, communication, storage, cloud computing technologies, and algorithms. Recent statistics reveal that 1200 Exabytes of data is generated annually and the rate is growing rapidly [1]. A huge fraction of this data is multimedia data (images and videos), generated by various industries, such as health, surveillance, agriculture, social web, online streaming services, movies, games, and internet protocol television (IPTV) industry [2]. For example, Facebook alone contains more than 40 billion photos [3]. Similarly, more than 500 h of videos are uploaded to YouTube every minute [4]. These massive amounts of data present enormous challenges for businesses and industries. At the same time, it provides opportunities for impressive future growth, based on effective utilization of the data for analysis. For instance, progress in medical imaging technologies allows visual analysis of patient through a variety of means including endoscopy, magnetic resonance imaging, radiography, ultrasonography, and many others. It causes huge amounts of data to be generated, which is stored for immediate or future use. Similarly, surveillance cameras deployed in wake of the recent security concerns throughout the globe, also generate huge amounts of multimedia data, required to be stored and properly indexed for possible future use. Major issues with these gigantic multimedia repositories include transmission, management, storage, and their efficient indexing and retrieval.

Providing reliable and efficient access to relevant data in large image repositories based on their contents is a highly challenging task which has been studied over the course of almost three decades. Content-based image retrieval (CBIR) methods allow retrieval of relevant images based on the content similarity between the query and target images [5], [6]. A core component of CBIR systems aims to represent images as feature vectors or feature histograms that correspond to the color or texture content of the image [7]. These systems can also be used to personalize and recommend contents for IPTV delivery services [8]. Traditionally, CBIR relied on hand-engineered features, such as scale invariant features transform [9], bag-of-visual-words histograms [10], [11], fisher vectors [12], vectors of locally aggregated descriptors [13], GIST [14], and CENsus TRansform hISTogram [15]. Each of these methods represented images in terms of low-level features; however, these features often fail to model high-level semantics in images. Therefore, their performance in large and challenging datasets was not very satisfactory [16]. In recent years, the hand-engineered feature extraction methods have been overshadowed by the feature learning based methods including

Manuscript received September 29, 2017; revised December 29, 2017; accepted January 22, 2018. This work was supported by the National Research Foundation of Korea grant funded by the Korea government (MSIP) (2016R1A2B4011712). Paper no. TII-17-2275. *(Corresponding author: Sung Wook Baik.)*

J. Ahmad, K. Muhammad, and S. W. Baik are with the Intelligent Media Laboratory, Digital Contents Research Institute, Sejong University, Seoul 143-747, South Korea (e-mail: jamilahmad@ieee.org; khan.muhammad@ieee.org; sbaik@sejong.ac.kr).

J. Lloret is with the Instituto de Investigacion para la Gestion Integrada de Zonas Costeras, Universitat Politecnica de Valencia, València 46022, Spain (e-mail: jlloret@dcom.upv.es).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TII.2018.2800163

deep convolutional neural networks (CNN), and deep denoising auto-encoders [17], [18]. They automatically extract features from images, which have been used in a variety of tasks, such as image classification, object localization, recognition, segmentation, and image retrieval [19]. CNNs have been widely used by the image retrieval community and have achieved state-of-the-art performance [16], [20]–[22]. These architectures have several convolutional, pooling, and fully connected (FC) layers, arranged in a hierarchy where successive layers learn complex features of the input [23]. Deep features are usually extracted from the FC layers of CNN which correspond to activation values of the neurons in those layers. In a typical CNN, these features often have thousands of dimensions. Though, these features are capable of representing images effectively, image indexing, and matching using these features become infeasible for large datasets [21].

Hash-based image retrieval methods aim at allowing efficient access to relevant data in large datasets using approximate nearest neighbor (ANN) search approaches. In wake of the growing demands for efficient access to large image repositories, these methods have appealed significant attention in recent years [24]. They work on the principle of locality sensitive hash functions that transform high dimensional features to low-dimensional hamming space (binary codes) and attempt to preserve original neighbors in the hamming space [25]. These compact codes are then used to directly retrieve nearest neighbors of the query image from the hamming space without exhaustive search. A large variety of hashing methods have been proposed in recent years, which attempt to derive compact binary codes from image features. A few notable methods include locality sensitive hashing (LSH) [25], [26], principal component analysis based hashing (PCAH) [27], spectral hashing (SH) [28], spherical hashing (SpH) [29], and density sensitive hashing (DSH) [30], etc. Hash methods may be data-dependent or data-independent. They may be trained in either supervised or unsupervised manner. Typically, these methods are trained for a particular dataset to generate hash codes of a certain length. If the data changes or the length of the hash code needs to be modified, the training procedure has to be rerun. These characteristics limit their utilization in real applications.

In this paper, we propose an efficient method to transform selected deep features directly into compact binary codes. It does not require any training and can be efficiently executed on a graphics processing unit (GPU) to quickly convert deep features to binary codes. We show that deep features from the FC layers of CNNs are highly redundant, hence, we propose a feature selection algorithm to identify effective deep features based on neuronal sensitivity and diversity. The proposed hash codes yield considerable retrieval performance for 256 and 512 bit codes. Major contributions in this work are summarized as follows.

1) We show that the high dimensional deep features extracted from FC layers of a pretrained CNN are redundant and a significant number of activation features can be removed without any loss in retrieval performance, particularly when dealing with images of a particular category such medical or surveillance.
2) An effective feature selection algorithm is proposed for deep feature based on neuronal sensitivity and diversity measures.
3) A highly efficient method is proposed for transforming deep features into compact binary codes, which can be used as hash codes for efficient image search. Our method uses bidirectional fast Fourier transform (BD-FFT) which allows hash codes of desired length to be computed directly without requiring any training. The method can be easily implemented on a GPU for significant speedup in hash code computation at large scale.
4) We also show that the selected deep features yield better hash codes with the proposed BD-FFT method, and offer better locality sensitivity with 256 and 512 bit codes.

The rest of the paper is organized as follows: Section II highlights strengths and weaknesses of recent hash-based retrieval methods. Section III explains the proposed method in detail, highlighting the key features of the presented algorithms. Section IV reports evaluation results of the proposed method on several popular datasets. The paper is concluded in Section V with some future research directions.

II. Related Work

Extraction of discriminative features is a primary factor in the success of CBIR systems. The recent deep learning based methods, especially CNNs yield highly discriminative features, which achieve state-of-the-art performance in CBIR. Several frameworks have been proposed for utilizing deep features for image retrieval in challenging scenarios. For instance, Krizhevsky *et al*. [23] showed that neuronal activations extracted from FC layers can be used as feature descriptors and image matching can be performed using standard Euclidean distance. They also showed that these high dimensional features can be easily compressed with dimensionality reduction methods, such as principal component analysis (PCA), sacrificing accuracy for some degree of efficiency. Razavian *et al*. [17], [18] and Babenko *et al*. [21], [22] showed that features from a pretrained CNN can be used as generic descriptors for image retrieval and other related tasks. They showed that features from a pretrained CNN, trained on a very large dataset (ImageNet [31]) achieve state-of-the-art performance, surpassing traditional hand-engineered features by a huge margin. Deep features from FC layers are very powerful global representations, however, they are high dimensional and directly utilizing them becomes inefficient, particularly for large scale datasets [32].

Large scale datasets demand efficient methods for storing millions of images in memory and quickly finding relevant images to a query image. ANN search methods like LSH have shown promising results in recent years. Typically images are represented as features vectors in high dimensional Euclidean space, such that the Euclidean distance corresponds to image similarity. The main objective of hashing methods is to generate a low-dimensional embedding in hamming space while preserving the neighborhood. Hence, when a query is issued, the hash code of the query image is used to efficiently access nearest

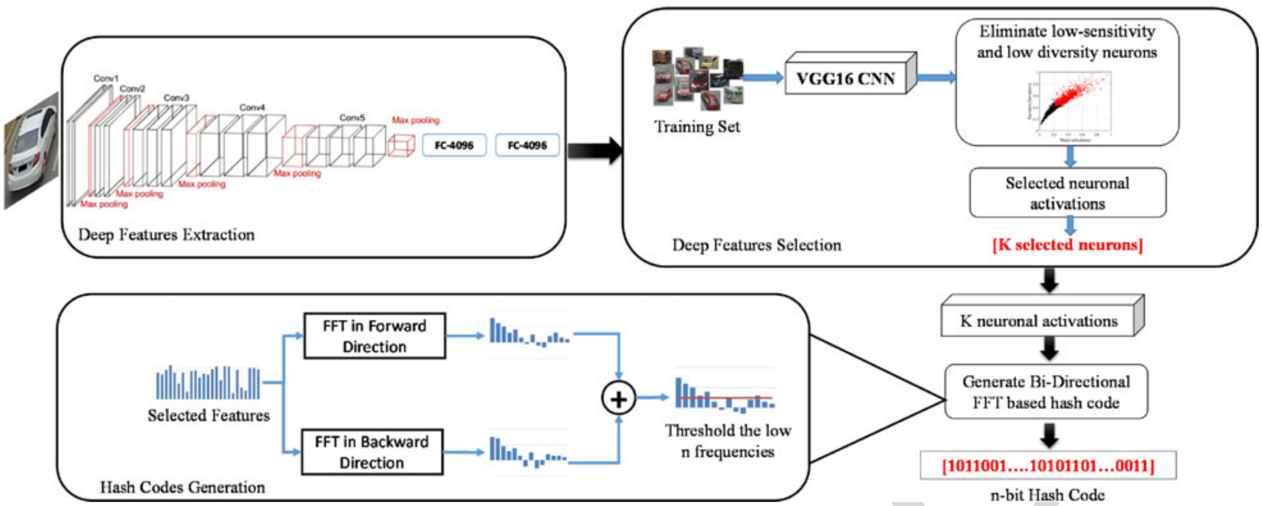

Fig. 1. Proposed framework.

neighbors of the query image using hamming distance. Based on this idea, several approaches have been presented in the recent years. For instance, PCAH [33] used principle directions of data as the projection vectors to transform features to binary codes. In LSH [25], [34], the binary code is computed through random linear projection with a random threshold. In theory, hamming distance between LSH codes and Euclidean distance between image pairs are highly correlated, however, in practice it can lead to very inefficient codes. SH [28] selects binary codewords though minimum distance between similar points, where similarity is defined by an approximate proximity matrix. Theoretically, it performs better than LSH, however, its optimization is difficult to generalize for new data points. This problem is solved with SpH [29] which uses Eigen functions of weighted Laplace–Beltrami operations with the assumption of having a multidimensional uniform distribution. It is highly efficient than SH for hash code generation, however, its optimization is computationally expensive. DSH [30] is an extension of LSH which utilizes random projections and also uses geometrical structure of the data to guide the projections. It partitions the data points into k-groups and splits each pair of adjacent groups with a projection vector. From all such projections, DSH selects the vectors based on the maximum entropy principle.

Hash-based image retrieval methods significantly improve retrieval efficiency in large scale datasets. However, these methods are difficult to implement in real applications and some of them require sufficient training data and time, while others are slow at transforming feature vectors to hash codes. An ideal hashing method is computationally efficient, simple to implement and yield state-of-the-art performance for a variety of datasets. In this paper, we present a simple and highly efficient way of transforming deep features to compact binary codes using BD-FFT.

III. Proposed Method

The proposed framework consists of two modules, feature selection and hash code generation as shown in Fig. 1. First, we studied deep features from FC layer of a pretrained VGG-16 CNN [35] in order to determine optimal set of features for a particular type of data. Once the optimal features are selected, they are converted to binary codes of different lengths using bidirectional FFT. Details of both modules are provided in the subsequent sections.

A. Deep Features for Image Retrieval

Informatics and analytics systems make use of efficient ways to access relevant information from large datasets. Visual data constitute a large fraction of the data generated by different industries, where accurate and efficient access will allow analysts and experts make better and timely decisions. Features extracted from deep CNNs have shown state-of-the-art performance in image retrieval from large datasets due to their impressive representational capabilities. We used features from FC-4096 layer of the VGG16 model [35] which was trained on ImageNet. These features are regarded as generic descriptors for visual recognition tasks including image classification and retrieval [17], [18]. However, we argue that these features are highly powerful, capable of representing a huge variety of visual data, and a subset of these features will be sufficient to effectively represent images of a particular type like medical radiographs or surveillance images of vehicles, etc. In such specific datasets, subsets of these generic features can prove to be more appropriate than the full set of features. For this purpose, we propose an efficient method to select deep features from a pretrained CNN for representing images of a particular type. Deep features from the FC layer are constructed as global representations by combining the local features extracted by various convolutional layers. VGG16 contains three FC layers having 4096, 4096, and 1000 neurons, respectively. We used activation values of the second FC-4096 layer in our experiments because of their superior performance. Each of these neurons are sensitive to particular objects or parts of objects [36]. When a particular object appears in an image, a subset of these neurons generate high activations indicating its presence. Though these features are considered generic and high level, their high dimensionality hinder their use in practical applications.

B. Optimal Deep Features Selection

Feature reduction offers improvements in efficiency and accuracy as it helps in getting rid of the less useful and often misleading features [37]. We propose an efficient method to select optimal features from a pretrained CNN. An input image is usually feed-forwarded though a deep CNN (e.g., VGG16) and the activation values from the FC-4096 layer are extracted, which are then used to index or retrieve images. In hash-based retrieval systems, these features are transformed to compact binary codes and then images are retrieved using hamming distance. However, utilizing all these features for hash code generation and retrieving images of specific type is ineffective.

Deep features from FC layers are global high level features where particular neurons are sensitive to particular objects or their parts. They respond actively when that particular part appears somewhere in the image. For a dataset consisting of a particular type of images, e.g., medical, it is highly unlikely that object parts belonging to other categories, such as sports, surveillance, or animals, may be encountered. In such a case, utilizing all the features to represent images become ineffective which may lead to decreased performance. In recent works, we have seen that fine-tuning pretrained CNNs on particular datasets yield better results, which is also a verification of the fact that specific features perform better than generic ones [16], [38]. Instead of fine-tuning, we propose to discard irrelevant features before using them for image retrieval tasks in specific datasets. For this purpose, we selected a representative set of images from a target dataset and extracted deep features from them. We eliminated those neurons which generated negligible activations (low sensitivity to objects of interest) or similar activations (less discriminative) for the training set. Mean activation values μ and standard deviations σ were computed for all 4096 neurons over the entire training set, where the training set Ts consisted of randomly chosen images from all the datasets we used in the experiments and were represented by R^{4096} vectors of deep features. Neurons having μi greater than the threshold t_μ, and σi greater than the threshold t_σ were selected as the data specific discriminative features in a set Fs. This process can be performed for selecting specific features for representing images of a particular category. The feature selection mechanism is presented in Algorithm 1.

Algorithm 1: Selection of optimal deep features.

Input: Training feature vectors Tf_i having size $T \times R^{4096}$ extracted from FC-4096 (VGG16)
Output: Indices of selected deep features F_s
Steps:

1. Calculate mean activation values μ_i and standard deviation σ_i for all 4096 neurons across the entire training set T
   For i = 1 to 4096
   $$\mu_i = \sum_{t=1}^{T} T f_i$$
   $$\sigma_i = \sqrt{\frac{\sum_{t=1}^{T}(T f_i - \mu_i)}{T}}$$
   End for
2. Keep the neurons whose μ_i are greater than t_μ and σ_i is greater than t_σ.
   $$F s_i = \left\{ \begin{array}{ll} Select\,neuron, & \mu_i > t_\mu \text{ and } \sigma_i > t_\sigma \\ Discard\,neuron, & \text{otherwise} \end{array} \right\}$$
   where t_μ and t_σ are selected empirically.
3. Return the indices of selected neurons in F_s.

Algorithm 2: Conversion of deep features to binary codes.

Input: Deep feature vector f_i having R^d
Output: n-bit binary code
Steps:

1. Compute FFT of fi in forward direction to obtain a Fourier spectrum F_f
   $$F_f = \sum_{j=0}^{d-1} f_i e^{-i2\pi kj/n}, \quad k = 0, \ldots, d-1$$
2. Compute FFT of fi in backward direction to obtain F_b
   $$F_b = \sum_{j=d-1}^{0} f_i e^{-i2\pi kj/n}, \quad k = 0, \ldots, d-1$$
3. Compute the sum of F_f and F_b to obtain F.
   $$F = F_f + F_b$$
4. Calculate the real part of F
   $$F' = real(F)$$
5. Calculate the mean frequency component f_m from F' without considering the DC component (F'_0)
   $$f_m = \frac{1}{d}\sum_{i=1}^{d-1} F'_i$$
6. Convert the low-n frequencies in F' to binary codes H using the f_m as a threshold
   $$H = \left\{ \begin{array}{ll} 1, & F'_i > f_m \\ 0, & \text{Otherwise} \end{array} \right\}$$
7. Return the n-bit binary code H.

C. Conversion to Compact Binary Codes

In this paper, we consider the selected feature vector as a one-dimensional signal, and construct its frequency domain representation using FFT. During this transformation, the time-domain signal is represented as a combination of different frequencies. These frequencies correspond to the activation patterns of neurons in the selected feature set. The Fourier spectrum effectively captures those patterns and represents them as frequencies with different amplitudes. The original signal can be reconstructed using a certain representative frequencies of this spectrum as shown in Fig. 2. Each frequency component will indicate the presence or absence of a certain frequency content (i.e., neuronal activation pattern) in the features. Based on this idea, we select low n frequency components of the spectrum (excluding the dc component) and transform them into binary codes as illustrated in Algorithm 2. Frequencies that are less than certain threshold are converted to zero bits and the rest are converted to ones. Though some information is lost during this conversion, the main gist of the spectrum is somehow retained which leads to high performing binary codes. Since each neuron represent a semantic concept (such as object part), a sufficiently

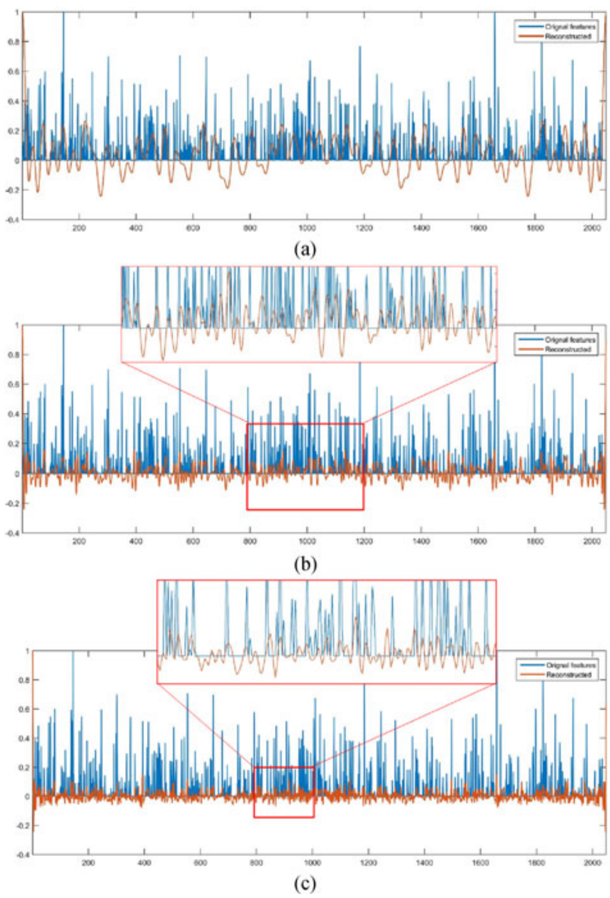

Fig. 2. Reconstruction of features from (a) 64 bits, (b) 256 bits, and (c) 512 bit hash codes generated using BD-FFT.

strong activation usually refer to the presence of that object part. With such high level representation, if the reconstructed signal adequately identify the high activation neurons, the code will be an effective representation of the original features. The procedure for conversion of deep features to binary codes is provided in Algorithm 2.

D. Bidirectional Fourier Decomposition

Though the simple FFT based binary conversion yield strong representative codes [39], their quality can be further improved with bidirectional FFT. In this case, we compute FFT of the features in both forward and backward directions and then add the corresponding frequency spectra. The dc component is ignored and the subsequent n frequency components are binarized to obtain the n-bit binary codes. Since the deep features are not time-dependent, the bidirectional FFT actually helps capture the patterns in neuronal activations more effectively, thereby yielding better codes. Experimental results revealed that the BD-FFT based codes perform much better than the regular FFT based codes as reported in the experiments section.

E. Locality Sensitivity of the Binary Codes

In LSH, the distance between the original features must correlate with the distance between the computed binary codes. To evaluate locality sensitivity of the proposed binary codes,

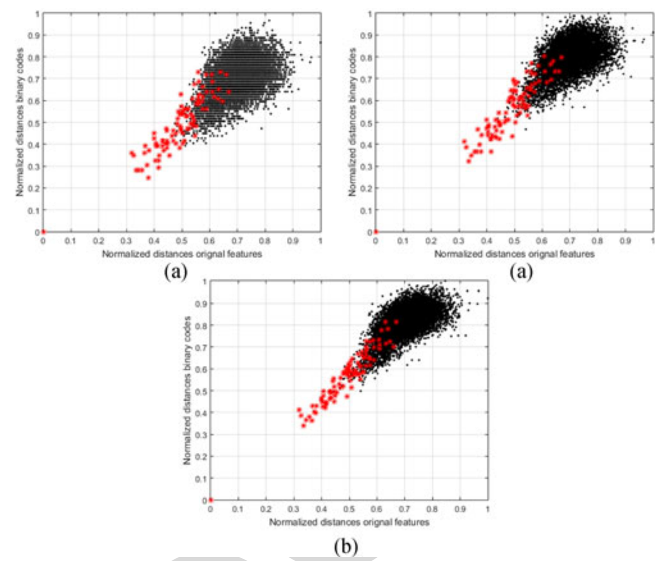

Fig. 3. Locality sensitivity of the proposed binary codes (a) 128 bits, (b) 256 bits, and (c) 512 bits.

we compared the normalized distances between deep features and their corresponding binary codes. Fig. 3 (a)– (c) reports the correlation among the distances between deep features and their corresponding binary codes. The distances of the query image with the rest of the images are shown on the x- and y-axis using deep features and binary codes, respectively. The red dots correspond to the relevant images and the black dots represent the irrelevant images in the dataset. Visualization of the distances reveal that the binary codes strongly correlate with the original deep features, especially for 256 and 512 bit codes, achieving correlation scores of 0.8975 and 0.9447, respectively. Increase in the distance between the original features is appropriately reflected by the distance between the binary codes. The relevant images have relatively smaller distances than the irrelevant ones which shows that those images will be retrieved at higher ranks. This characteristic of the proposed binary codes will help it achieve almost similar performance as the deep features.

IV. Experiments and Results

In this section, we present a detailed evaluation of the proposed method on a number of datasets used for benchmarking image retrieval methods. Different experiments were designed to measure performance of the proposed scheme and the effects of deep feature selection. All the experiments were carried out in MATLAB [40] environment on a Windows 7 PC equipped with 16 GB RAM. All the hashing methods were implemented and evaluated in MATLAB.

A. Datasets

A number of datasets have been used to evaluate retrieval performance of the proposed method, including Corel-10 K, Holiday [41], IRMA-2009 [42], vehicle reidentification (VeRI) dataset [43], and stanford online products (SOP) dataset [44]. Each of these datasets contain thousands of images and are widely used to benchmark CBIR systems. Corel-10 K and Holiday datasets consist a variety of natural images whereas

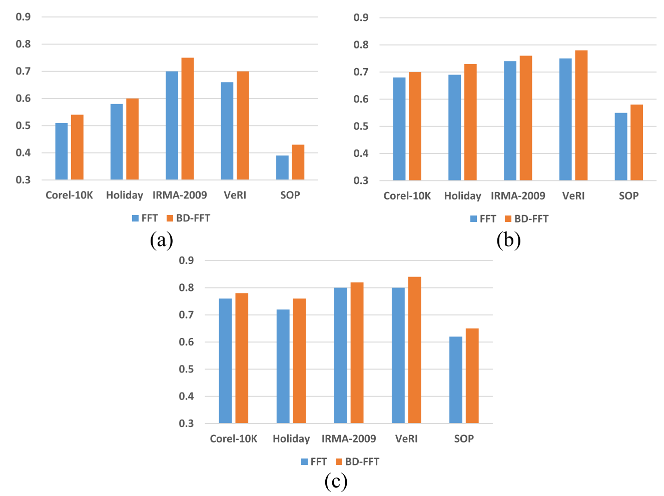

Fig. 4. Retrieval performance comparison FFT and BD-FFT based hash codes for (a) 128-bit, (b) 256-bit, and (c) 512-bit hash codes.

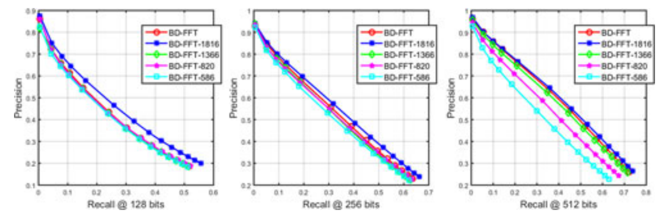

Fig. 5. Retrieval performance with hash codes generated from varying subsets of deep features for Corel-10 K dataset.

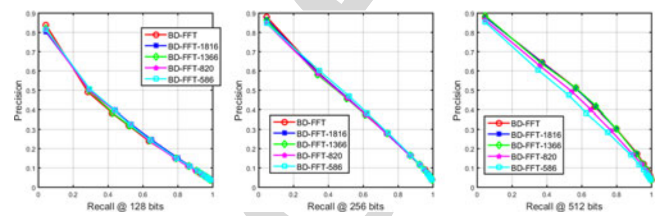

Fig. 6. Retrieval performance with hash codes generated from varying subsets of deep features for holiday dataset.

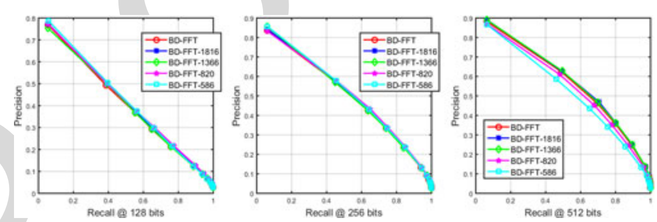

Fig. 7. Retrieval performance with hash codes generated from varying subsets of deep features for IRMA-2009 dataset.

IRMA-2009, VeRI, and SOP contain images of particular categories including medical radiographs, vehicles, and products, respectively.

B. Retrieval Performance of FFT Versus BD-FFT

A bidirectional Fourier decomposition of the feature vector allowed us to capture patterns in the neuronal activations in a much better way. Each bit in the hash code indicate either the presence (1-valued bits) or absence (0 valued bits) of activation pattern in the original features. With BD-FFT, certain high frequency patterns are captured in a much better manner than the regular FFT based codes which leads to its superior performance as reported in Fig. 4. The precision scores for various datasets have been computed at recall $= 0.2$. The results reveal that BD-FFT yield 3% to 10% better performance in terms of precision scores as compared to FFT for all datasets at different code lengths.

C. Retrieval Performance With Hash Codes Using Different Subsets of Deep Features

In these experiments, we evaluated retrieval performance using hash codes of different lengths, computed from different subsets of deep features. Hash codes of 128, 256, and 512 bits were generated for five different sets of features, which contained 4096, 1816, 1366, 820, and 585 neuronal activations. These subsets were obtained by varying the threshold values in Algorithm 1. Several images were selected at random from each dataset and top ranked images were retrieved using hamming distance between the query code and codes in the database. The commonly used metrics including precision and recall were used to report retrieval performance for each dataset. Fig. 5 shows retrieval results in Corel-10 K dataset with 128, 256, and 512 bit codes for five different subsets of features. For each subset of features, the precision-recall curves are presented for hash codes of different lengths. In all of these results, the subset with 1816 activations yield better performance than the other subsets, even the full-feature set. The margin is clearly visible in 128-bit codes and gradually reduces for 256 and 512 bit hash codes, yet it performs better than the other sets of features. Interestingly, the performance of other reduced feature sets remains almost the same as the full feature set, especially at 128 and 256 bit codes. However, the 820 and 586 dimensional features failed to catchup to the performance with other subsets in 512 bit codes. It is important to observe here that performance remains almost unchanged even if significant number of neuronal activations are dropped. In 512 bit code, the scores for 4096, 1816, and 1366 features are almost the same. These results reveal the redundant nature of deep features extracted from the FC layer.

The same experiments were carried out for Holiday image datasets and the results presented in Fig. 6 reveal similar results as compared to Corel-10 K. Features with 820 and 586 scores slightly lower at 128 bits than the other subsets. However, the performance with 4096, 1816 and 1366 features remains the same for all hash codes. Though we did notice slightly better performance at low recall for 1816 and 1366 subsets, the reduced feature set performed almost the same as the full feature set. The same results were observed with IRMA-2009 dataset as shown in Fig. 7, where the reduced feature sets perform slightly better at low recall and yield similar performance to the full feature set for the rest of recall values with 128 and 256 bit codes. However, with 512 bits, 1816-d, and 1366-d features achieve better precision than the full feature set at all recall settings.

The VeRI dataset is quite challenging due to its large volume and diversity. Carefully chosen subsets of features either perform better than the full feature set or yield identical

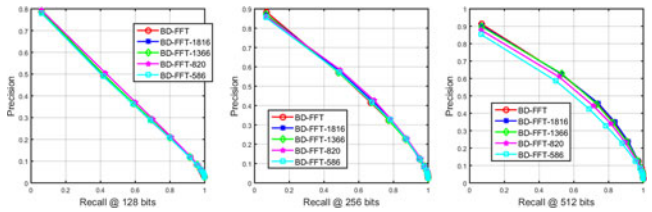

Fig. 8. Retrieval performance with hash codes generated from varying subsets of deep features for VeRI dataset.

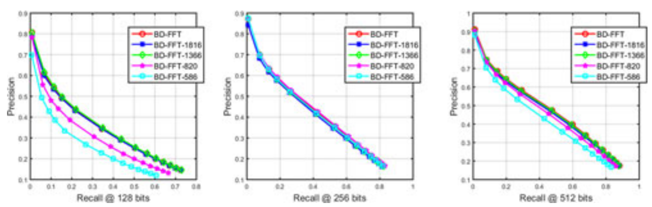

Fig. 9. Retrieval performance with hash codes generated from varying subsets of deep features for SOP dataset.

performance. In this dataset, we observed similar performance for all subsets with 128 and 256 bit codes. With 512 bit codes, 820-d, and 586-d features scored slightly lower precision at all recall settings as shown in Fig. 8. Finally, same experiments were run for the SOP dataset which is the most challenging dataset with huge volume and large number of product categories. Precision scores dropped significantly when recall rates are increased, particularly for 128 bit codes. At this length, the hash codes generated for 4096, 1816, and 1366 features yield similar retrieval performance, whereas the other subsets achieve very low precision scores. With 256 bit codes, all the subsets achieve similar precision scores at all recall rates. At 512 bits, 1816-d, and 1366-d features score almost the same as the 4096-d features as presented in Fig. 9.

With these results, we can conclude that the FC layer features are highly redundant and can be substantially reduced without any loss in performance. Even in some cases, may get improved retrieval results. Through these experiments, we decided to utilize the selected 1816 neuronal activations from the FC-7 layer instead of the 4096 features to generate hash codes for efficient image retrieval in large datasets.

D. Retrieval Performance With State-of-the-Art Hashing Schemes

In this section, we compare the retrieval performance of the proposed hash codes with five other schemes including LSH [25], [34], SH [28], PCAH [33], DSH [30], and SpH [29]. In these experiments, query images were randomly chosen from each dataset and top ranked images were retrieved using hash codes of 128, 256, and 512 bits. Precision-recall scores are reported for each experiment. Fig. 10 presents the retrieval performance of various hashing methods for Corel-10 K dataset. The proposed method performed better than LSH at 128 bits, however, it achieved low precision scores compared to other methods. At 256 bits, BD-FFT outperformed LSH and PCAH at low recalls, and LSH, PCAH, and SH at high recall rates. At

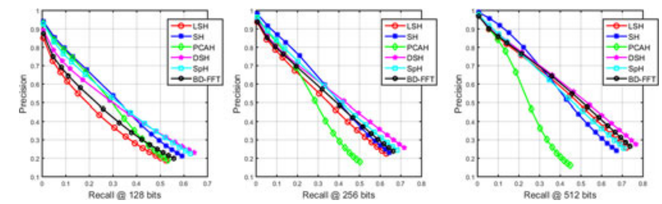

Fig. 10. Retrieval performance with hash codes compared with state-of-the-art methods for Corel-10 K dataset.

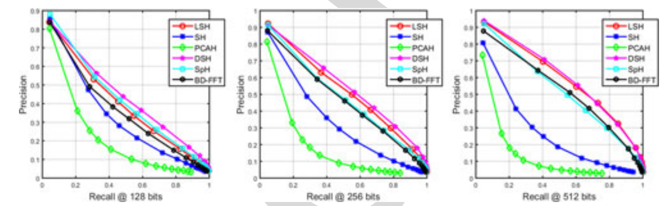

Fig. 11. Retrieval performance with hash codes compared with state-of-the-art methods for holiday dataset.

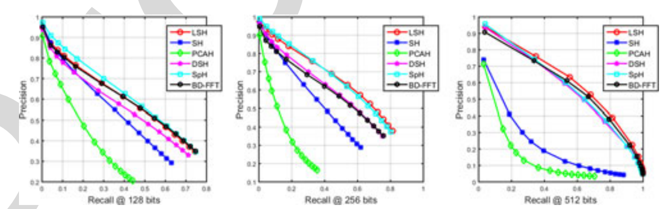

Fig. 12. Retrieval performance with hash codes compared with state-of-the-art methods for IRMA-2009 dataset.

low recall rates, BD-FFT performed similar to DSH. The performance of BD-FFT improved significantly with 512 bit codes where it outperformed LSH and PCAH at low recalls and LSH, PCAH, SH, and SpH at all recall rates above 0.35.

In Holiday dataset, BD-FFT performed better than PCAH and SH at 128 and 256 bit codes (see Fig. 11). At 512 bits, it significantly outperformed PCAH, SH, and yielded slightly better precision scores than SpH at most recall settings. However, the performance of LSH and DSH was relatively better for this dataset. In IRMA-2009 dataset, BD-FFT yielded better results than PCAH, SH, DSH, and LSH at 128 bit codes. Only SpH performed slightly better than our method. With 256 bit codes, BD-FFT scored better than PCAH and SH, however it performed slightly poor than the rest of the methods. Increasing the hash code length to 512 bits resulted in much better performance of our method, surpassing SpH, SH, PCAH, and DSH for recall rates above 0.4 as shown in Fig. 12.

In the VeRI dataset, BD-FFT significantly outperformed PCAH, SH, and DSH in all experiments. With 512 bits, it performed better than LSH at high recalls and reached the performance of SpH (see Fig. 13). Similarly in SOP dataset, BD-FFT outperformed PCAH and SH at 128, 256, and 512 bit codes. However the other methods LSH, SpH, and DSH performed much better at low recall rates as shown in Fig. 14. This is the most challenging dataset and that is why its precision scores are much lower than the other datasets.

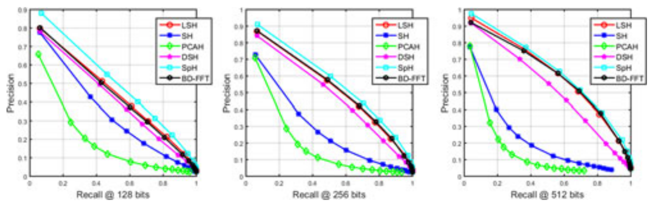

Fig. 13. Retrieval performance with hash codes compared with state-of-the-art methods for VeRI dataset.

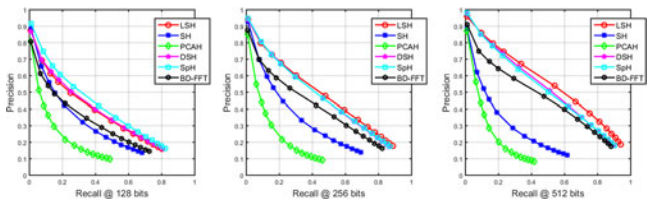

Fig. 14. Retrieval performance with hash codes compared with state-of-the-art methods for SOP dataset.

In most of the datasets, BD-FFT outperformed majority of the methods, and achieved impressive performance especially with 256 and 512 bit hash codes. Moreover, the proposed method yields more significant performance gains than the other competing methods when size of the hash code increases. Keeping in view the simplicity of our method, these results are very promising. From these results, we can conclude that the proposed method is capable of transforming high dimensional deep features to compact binary codes of any length. We recommend hash codes of length 256 or 512 bits to be used for image indexing and retrieval in large datasets. Though higher length codes can also be generated in the same efficient manner, which may yield performance improvements in most cases.

E. Qualitative Retrieval Performance Using the Proposed Hash Codes

In this experiment, randomly chosen query images were used to retrieve top-ranked images from each of the five datasets using hash codes generated with the proposed BD-FFT method having 512-bit length. Results of two queries have been shown for each dataset in terms of top 20 retrieved images in Fig. 15. Results reveal that the proposed hash codes is capable of retrieving relevant images at top ranks despite the huge volume and diversity within these datasets, particularly IRMA-2009, Stanford Online Products, and VeRI. The proposed hash codes can effectively represent deep features, allowing almost the same retrieval results as the raw features. The top two queries were taken from Corel-10 K dataset where all relevant images have been retrieved at top ranks. The next two rows contain results from Holiday dataset where the first query image had three other relevant images in the dataset, which have been successfully retrieved at top ranks. It is important to note here, that the rest of the images, though irrelevant, resemble the query image in visual appearance. Similar is the case with the other query where the images at ranks 1, 2, 3, and 5, have been correctly retrieved. The other images are also visually similar to the query image. In the third pair of queries, visually similar images have

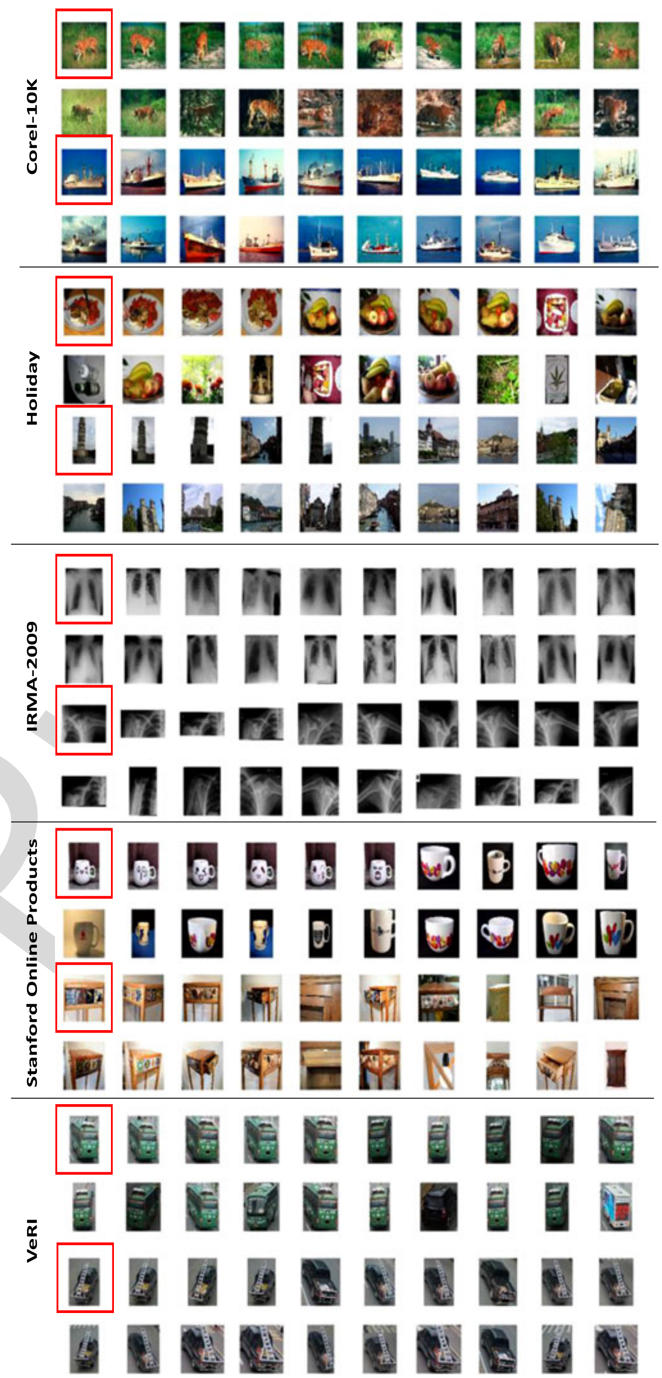

Fig. 15. Retrieval results using BD-FFT based 512-bit hash codes.

TABLE I
TRAINING TIME REQUIRED (IN SECONDS) FOR VARIOUS HASHING METHODS

Method	Training Time (20000 × 4096 features) 512-bits
LSH	0.03
SH	20.6
PCAH	19.7
DSH	30.2
SpH	252.1
BD-FFT	0.00

TABLE II
TIME REQUIRED (IN SECONDS) FOR TRANSFORMING FEATURES TO HASH CODES USING VARIOUS METHODS

Method	Feature Size								
	10000 × 4096			20000 × 4096			200000 × 4096		
	128-bit	256-bit	512-bit	128-bit	256-bit	512-bit	128-bit	256-bit	512-bit
LSH	0.30	0.31	0.40	0.60	0.61	0.62	1.68	2.92	5.50
SH	1.10	4.47	16.18	2.20	8.32	33.3	24.76	84.86	341.9
PCAH	0.07	0.13	0.27	0.16	0.33	0.55	1.53	2.78	5.49
DSH	0.08	0.14	0.29	0.16	0.34	0.59	2.18	3.09	6.2
SpH	0.22	0.33	0.63	0.47	0.71	1.44	0.51	0.98	1.82
BD-FFT (CPU)		0.55			1.2			13.9	
BD-FFT (GPU)		**0.02**			**0.041**			**0.43**	

TABLE III
STORAGE SPACE REQUIREMENTS FOR 1 MILLION IMAGES WITH DEEP FEATURES AND PROPOSED HASH CODES

Features	Storage required (MB)	Storage required (GB)	Retrieval performance % of original features
Raw (4096 deep features)	31250	30.51758	100
512-bit	**61.03516**	**0.059605**	**97.02**
256-bit	**30.51758**	**0.029802**	**92.09**
128-bit	15.25879	0.014901	86.10
64-bit	7.629395	0.007451	64.25
32-bit	3.814697	0.003725	40.31

been successfully retrieved at top ranks for both queries. The last two pairs of queries are from the most challenging datasets SOP, and VeRI. Despite the challenging nature and large size of these datasets, the proposed codes were able to retrieve the relevant images at top ranks. These results show the promising performance of the proposed codes. With sufficiently sized codes, almost the same retrieval results can be achieved with the proposed method.

F. Efficiency Analysis

In this section, we evaluate efficiency of the proposed scheme in terms of training time, hash code computation time, and storage requirements for the varying length hash codes. We aim to provide an insight into how efficient the proposed method is, compared to other similar approaches. In Table I, we listed the training times for various competing methods when 20 000 features having 4096-dimensions were used for training the hashing functions. The training time mentioned in seconds, reveal that the LSH method is the quickest to train and takes only 0.03 s. The SH and PCAH methods take around 20 s, whereas, DSH require 30.2 s. The most computationally expensive method was found to be SpH which took 252.1 s to train for generating 512-bit hash codes. Though some of these methods are quite fast to train, they would require retraining when the hash code size gets changed. Further, the data-dependent methods like SH and SpH require to be trained each time when utilized for a different kind of dataset. Contrary to these methods, the proposed method do not require any training and can be used to directly transform deep features into binary hash codes of any length. Further, using specialized hardware (GPU), the proposed method can be executed in parallel, yielding very high speeds for transforming features to hash codes. These characteristics make its implementation in real applications very easy. The proposed method can be easily implemented to transform the indexed features to binary codes which would allow efficiently locating similar images using ANN schemes.

Table II lists the hash code computation times for varying length codes using deep features. We used three test sets, having 10 K, 20 K, and 200 K vectors of 4096-d to evaluate the conversion efficiency. Hash codes of 128, 256, and 512-bits were obtained using different hashing methods and the conversion times were recorded. The average conversion times reported in Table II reveal that majority of the methods including LSH, PCAH, DSH, and SpH are very efficient when shorter length hash codes are generated. The slowest method SH required 1.10 s to convert 10 K features to 128-bit hash codes, however it took 341.9 s to convert 200 K features to 512-bit codes. In comparison, most of the hashing methods are more efficient than the proposed method on a CPU, which require 0.55, 1.2, and 13.9 s to convert 10 K, 20 K, and 200 K features into 128, 256, and 512 bit hash codes, respectively. However, the advantage of the proposed method over other methods is that it can be easily computed on a GPU which yield significant gains in efficiency, reducing the computation times to 0.0002, 0.041, and 0.43 s for 128, 256, and 512-bits, respectively. If the proposed method is implemented on a GPU, it can compute hash codes significantly faster than all the other competing methods. This characteristic also favors our method for implementation in practical applications.

In Table III, we show the amount of storage required for 1 Million images when the raw features are stored to index images. We also show the amount of storage required to index 1M images with 32, 64, 128, 256, and 512-bit codes. In addition, we also report the relative image retrieval performance to the original deep features for each code. With 32-bit codes, we would require only 3.8 MB storage to index the images, however we would only get 40.3% retrieval performance. Hash codes greater than or equal to 128-bits, yield considerable retrieval performance as well as saves storage space. The recommended setting is to generate 256 or 512 bit codes for representing images because they would respectively yield 92% and 97% relative retrieval performance as compared to the original features. Further, these hash codes reduce the storage requirements of the index file from 30.5 GB to only 30 or 61 MB, which allow them to be easily fit into memory. This would significantly improve retrieval efficiency for large scale datasets.

V. Conclusion and Future Work

In this paper, we presented an efficient method to directly transform deep features into compact hash codes with locality sensitivity property. These hash codes allow efficient retrieval from large scale datasets utilizing ANN search procedures. The proposed hash code conversion method require two steps. First, salient deep features are selected using the proposed feature selection algorithm, which analyzes the deep features and selects features with higher diversity than a certain threshold. We analyzed deep features and found that these features are highly redundant and a significant number of these features can be ignored without any loss in retrieval performance. Through experiments, we determined 1816 features out of 4096 to represent images. In the second step, we computed the FFT of these selected features and binarized the top-n frequencies using mean frequency as the threshold. The parameter n determined the desired length of the hash code. The main idea behind the proposed method is to represent the selected deep feature as a signal and the FFT is used to approximate the feature vector in the frequency domain. The computed hash codes have significant representational capability with 128, 256, and 512 bit codes, where the 512 bit codes yield almost the same retrieval accuracy as the original deep features.

An essential characteristic of the proposed hashing method is that it is completely data-independent and does not require any training. Hash codes of any length can be directly computed very efficiently. The implementation and operational simplicity of the proposed scheme makes it very convenient to be implemented in real-world applications. Further, GPU based acceleration of the proposed method can substantially improve overall efficiency of the retrieval system of large scale datasets. In this work, we showed that the proposed method yield comparable performance to the state-of-the-art for codes above 256 bits, however its performance with smaller codes is relatively weak. Further, the proposed method performs well for deep features, however, it may not perform well for sparse features and further study is needed to improve its performance for any type of features.

In future, we plan to study the effects of deep features on its frequency spectrum and devise more effective ways of capturing information in deep features into the compact binary representations. Further, we will also evaluate wavelet based methods to construct high performance short codes so that the retrieval efficiency could be further enhanced.

References

[1] P. Louridas and C. Ebert, "Embedded analytics and statistics for big data," *IEEE Softw.*, vol. 30, no. 6, pp. 33–39, Nov./Dec. 2013.

[2] J. Lloret, M. Garcia, M. Atenas, and A. Canovas, "A QoE management system to improve the IPTV network," *Int. J. Commun. Syst.*, vol. 24, pp. 118–138, 2011.

[3] B. Thomee *et al.*, "YFCC100M: The new data in multimedia research," *Commun. ACM*, vol. 59, pp. 64–73, 2016.

[4] M. R. Robertson, "300+ hours of video uploaded to youtube every minute," *ReelSEO*, vol. 21, Nov. 2014.

[5] J. Ahmad, M. Sajjad, S. Rho, and S. W. Baik, "Multi-scale local structure patterns histogram for describing visual contents in social image retrieval systems," *Multimedia Tools Appl.*, vol. 75, pp. 12669–12692, 2016.

[6] J. Ahmad, M. Sajjad, I. Mehmood, S. Rho, and S. W. Baik, "Saliency-weighted graphs for efficient visual content description and their applications in real-time image retrieval systems," *J. Real-Time Image Process.*, vol. 13, pp. 431–447, Sep. 2017.

[7] A. W. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 22, no. 12, pp. 1349–1380, Dec. 2000.

[8] M. Garcia, A. Canovas, M. Edo, and J. Lloret, "A QoE management system for ubiquitous IPTV devices," in *Proc. 3rd Int. Conf. Mobile Ubiquitous Comput., Syst., Service Technol.*, 2009, pp. 147–152.

[9] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *Int. J. Comput. Vis.*, vol. 60, pp. 91–110, 2004.

[10] S. Lazebnik, C. Schmid, and J. Ponce, "Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2006, pp. 2169–2178.

[11] H. Jégou, M. Douze, and C. Schmid, "Improving bag-of-features for large scale image search," *Int. J. Comput. Vis.*, vol. 87, pp. 316–336, 2010.

[12] M. Douze, A. Ramisa, and C. Schmid, "Combining attributes and fisher vectors for efficient image retrieval," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2011, pp. 745–752.

[13] H. Jégou, M. Douze, C. Schmid, and P. Pérez, "Aggregating local descriptors into a compact image representation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2010, pp. 3304–3311.

[14] A. Oliva and A. Torralba, "Modeling the shape of the scene: A holistic representation of the spatial envelope," *Int. J. Comput. Vis.*, vol. 42, pp. 145–175, 2001.

[15] J. Wu and J. M. Rehg, "CENTRIST: A visual descriptor for scene categorization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 8, pp. 1489–1501, Aug. 2011.

[16] J. Ahmad, M. Sajjad, I. Mehmood, and S. W. Baik, "SiNC: Saliency-injected neural codes for representation and efficient retrieval of medical radiographs," *PloS One*, vol. 12, 2017, Art. no. e0181707.

[17] A. Razavian, H. Azizpour, J. Sullivan, and S. Carlsson, "CNN features off-the-shelf: An astounding baseline for recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2014, pp. 806–813.

[18] H. Azizpour, A. Razavian, J. Sullivan, A. Maki, and S. Carlsson, "From generic to specific deep representations for visual recognition," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2015, pp. 36–45.

[19] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2014, pp. 580–587.

[20] J. Ahmad, K. Muhammad, and S. W. Baik, "Data augmentation-assisted deep learning of hand-drawn partially colored sketches for visual search," *PloS One*, vol. 12, 2017, Art. no. e0183838.

[21] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," in *Proc. Eur. Conf. Comput. Vis.*, 2014, pp. 584–599.

[22] A. Babenko and V. Lempitsky, "Aggregating local deep features for image retrieval," in *Proc. IEEE Int. Conf. Comput. Vis.*, 2015, pp. 1269–1277.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Proc. 25th Int. Conf. Inf. Process. Syst.*, 2012, pp. 1097–1105.

[24] J. Wang, W. Liu, S. Kumar, and S.-F. Chang, "Learning to hash for indexing big data—A survey," *Proc. IEEE*, vol. 104, no. 1, pp. 34–57, Jan. 2016.

[25] A. Gionis, P. Indyk, and R. Motwani, "Similarity search in high dimensions via hashing," in *Proc. 25th Int. Conf. Very Large Data Bases*, 1999, pp. 518–529.

[26] M. Datar, N. Immorlica, P. Indyk, and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proc. 20th Annu. Symp. Comput. Geom.*, 2004, pp. 253–262.

[27] X. Yu, S. Zhang, B. Liu, L. Zhong, and D. Metaxas, "Large scale medical image search via unsupervised PCA hashing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2013, pp. 393–398.

[28] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Proc. Adv. Neural Inf. Process. Syst*., 2009, pp. 1753–1760.

[29] J.-P. Heo, Y. Lee, J. He, S.-F. Chang, and S.-E. Yoon, "Spherical hashing," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2012, pp. 2957–2964.

[30] Z. Jin, C. Li, Y. Lin, and D. Cai, "Density sensitive hashing," *IEEE Trans. Cybern.*, vol. 44, no. 8, pp. 1362–1371, Aug. 2014.

[31] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2009, pp. 248–255.

[32] K. Lin, H.-F. Yang, J.-H. Hsiao, and C.-S. Chen, "Deep learning of binary hash codes for fast image retrieval," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. Workshops*, 2015, pp. 27–35.

[33] X.-J. Wang, L. Zhang, F. Jing, and W.-Y. Ma, "Annosearch: Image auto-annotation by search," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, 2006, pp. 1483–1490.

[34] M. Slaney and M. Casey, "Locality-sensitive hashing for finding nearest neighbors [lecture notes]," *IEEE Signal Process. Mag.*, vol. 25, no. 2, pp. 128–131, Mar. 2008.

[35] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representations*, 2015.

[36] A. B. Valdez, M. H. Papesh, D. M. Treiman, K. A. Smith, S. D. Goldinger, and P. N. Steinmetz, "Distributed representation of visual objects by single neurons in the human brain," *J. Neurosci.*, vol. 35, pp. 5180–5186, 2015.

[37] G. Chandrashekar and F. Sahin, "A survey on feature selection methods," *Comput. Elect. Eng.*, vol. 40, pp. 16–28, 2014.

[38] A. Babenko, A. Slesarev, A. Chigorin, and V. Lempitsky, "Neural codes for image retrieval," in *Proc. 13th Eur. Conf. Comput. Vis*., 2014, pp. 584–599.

[39] J. Ahmad, K. Muhammad, and S. W. Baik, "Medical image retrieval with compact binary codes generated in frequency domain using highly reactive convolutional features," *J. Med. Syst.*, vol. 42, p. 24, Dec. 19, 2017.

[40] "MATLAB," 2016a ed: MathWorks, 2016.

[41] H. Jegou, M. Douze, and C. Schmid, "Hamming embedding and weak geometric consistency for large scale image search," in *Proc. 10th Eur., Conf. Comput. Vis*., 2008, pp. 304–317.

[42] H. Müller *et al.*, "Overview of the CLEF 2009 medical image retrieval track," in *Proc. Workshop Cross-Lang. Eval. Forum Eur. Lang.*, 2009, pp. 72–84.

[43] X. Liu, W. Liu, H. Ma, and H. Fu, "Large-scale vehicle re-identification in urban surveillance videos," in *Proc. IEEE Int. Conf. Multimedia Expo.*, 2016, pp. 1–6.

[44] H. Oh Song, Y. Xiang, S. Jegelka, and S. Savarese, "Deep metric learning via lifted structured feature embedding," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, 2016, pp. 4004–4012.

Jamil Ahmad (S'16) received the BCS degree in computer science from the University of Peshawar, Peshawar, Pakistan, in 2008 with distinction. He received the Master's degree in computer science with specialization in image processing from Islamia College, Peshawar, Pakistan, in 2014. He is currently working toward the Ph.D. degree in digital contents at Sejong University, Seoul, South Korea.

He is also a Regular Faculty Member with the Department of Computer Science, Islamia College Peshawar, Peshawar, Pakistan. He has published several articles in these areas in reputed journals, including *Journal of Real-Time Image Processing*, *Multimedia Tools and Applications*, *Journal of Visual Communication and Image Representation*, *PLOS One*, *Journal of Oral and Maxillofacial Surgery*, *Computers and Electrical Engineering*, *SpringerPlus*, *Journal of Sensors*, and KSII *Transactions on Internet and Information Systems*. He is also an Active Reviewer for *IET Image Processing*, *Engineering Applications of Artificial Intelligence*, *KSII Transactions on Internet and Information Systems*, *Multimedia Tools and Applications*, IEEE TRANSACTIONS ON IMAGE PROCESSING, and IEEE TRANSACTIONS ON CYBERNETICS. His research interests include deep learning, medical image analysis, content-based multimedia retrieval, and computer vision.

Khan Muhammad (S'16) received the Bachelor's degree in computer science from the Islamia College Peshawar, Peshawar, Pakistan, in 2014, with a focus on information security. He is currently working toward the M.S. degree leading to Ph.D. degree in digital contents at Sejong University, Seoul, South Korea.

Since 2015, he has been a Research Associate with the Intelligent Media Laboratory. He has authored more than 40 papers in peer-reviewed international journals and conferences, such as IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, *Future Generation Computer Systems*, *Neurocomputing*, the IEEE ACCESS, the *Journal of Medical Systems*, *Biomedical Signal Processing and Control*, *Multimedia Tools and Applications, Pervasive and Mobile Computing*, *SpringerPlus*, *KSII Transactions on Internet and Information Systems*, MITA 2015, PlatCon 2016, FIT 2016, and ICNGC 2017. His research interests include image and video processing, information security, image and video steganography, video summarization, diagnostic hysteroscopy, wireless capsule endoscopy, computer vision, deep learning, and video surveillance.

Jaime Lloret (M'07–SM'10) received the B.Sc. and M.Sc. degrees in physics from University of Valencia, Valencia, Spain, in 1997, the B.Sc. and M.Sc. degrees in electronic engineering from University of Valencia, Valencia, Spain, in 2003 and the Ph.D. (Dr. Ing.) degree in telecommunication engineering from the Polytechnic University of Valencia, Valencia, Spain, in 2006.

He is currently an Associate Professor with the Polytechnic University of Valencia, Valencia, Spain. He is the Chair of the Integrated Management Coastal Research Institute and the Head of the "Active and collaborative techniques and use of technologic resources in the education" Innovation Group. He has authored 22 book chapters and has more than 400 research papers published in national and international conferences and journals. He is an Editor-in-Chief of the "Ad Hoc and Sensor Wireless Networks" (with ISI Thomson Impact Factor) and "Networks Protocols and Algorithms". He leaded many national and international projects. He is currently the Chair of the working group of the Standard IEEE 1907.1. He has been General Chair (or Co-Chair) of 38 International workshops and conferences. He is an IARIA Fellow.

Sung Wook Baik (M'16) received the B.S degree in computer science from Seoul National University, Seoul, South Korea, in 1987, the M.S. degree in computer science from Northern Illinois University, Dekalb, Dekalb, IL, USA, in 1992, and the Ph.D. degree in information technology engineering from George Mason University, Fairfax, VA, USA, in 1999.

From 1997 to 2002, he was a Senior Scientist of the Intelligent Systems Group with Datamat Systems Research Inc. In 2002, he joined the faculty of the College of Electronics and Information Engineering, Sejong University, Seoul, South Korea, where he is currently a Full Professor and Dean of Digital Contents. He is also the Head of Intelligent Media Laboratory, Sejong University, Seoul, South Korea. His research interests include computer vision, multimedia, pattern recognition, machine learning, data mining, virtual reality, and computer games.