Binary Codes for Tagging X-Ray Images via Deep De-Noising Autoencoders

Antonio Sze-To
Systems Design Engineering
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1
Email: hy2szeto@uwaterloo.ca

H.R. Tizhoosh
KIMIA Lab
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1
Email: tizhoosh@uwaterloo.ca

Andrew K.C. Wong
Systems Design Engineering
University of Waterloo
Waterloo, Ontario, Canada N2L 3G1
Email: akcwong@uwaterloo.ca

***Abstract*—A Content-Based Image Retrieval (CBIR) system which identifies similar medical images based on a query image can assist clinicians for more accurate diagnosis. The recent CBIR research trend favors the construction and use of binary codes to represent images. Deep architectures could learn the non-linear relationship among image pixels adaptively, allowing the automatic learning of high-level features from raw pixels. However, most of them require class labels, which are expensive to obtain, particularly for medical images. The methods which do not need class labels utilize a deep autoencoder for binary hashing, but the code construction involves a specific training algorithm and an ad-hoc regularization technique. In this study, we explored using a deep de-noising autoencoder (DDA), with a new unsupervised training scheme using only backpropagation and dropout, to hash images into binary codes. We conducted experiments on more than 14,000 x-ray images. By using class labels only for evaluating the retrieval results, we constructed a 16-bit DDA and a 512-bit DDA independently. Comparing to other unsupervised methods, we succeeded to obtain the lowest total error by using the 512-bit codes for retrieval via exhaustive search, and speed up 9.27 times with the use of the 16-bit codes while keeping a comparable total error. We found that our new training scheme could reduce the total retrieval error significantly by 21.9%. To further boost the image retrieval performance, we developed Radon Autoencoder Barcode (RABC) which are learned from the Radon projections of images using a de-noising autoencoder. Experimental results demonstrated its superior performance in retrieval when it was combined with DDA binary codes.**

I. Introduction

Retrieving similar medical images given a query image can be useful for more accurate diagnosis. A Content-Based Image Retrieval (CBIR) system, which identifies similar images based on an input image, is thus important for fields such as radiology and pathology. Building such a system is formidably difficult for two reasons. First, the similarity is hard to define for visual data. There is a semantic gap between low-level pixel values and high-level semantics [1]. Second, the recent advance in medical imaging devices has led to the production of a gigantic amount of image data. It has been reported that the Vanderbilt Medical Center had compiled over 100 million anonymized medical images in 18 months [2]. A sophisticated CBIR system must be able to search through this Big Medical Image Data efficiently in response to the user's query. Hence, for medical images, accuracy and speed are the two most critical criteria for performance of a CBIR system.

The recent trend in CBIR research is the construction and use of binary representations (codes) for image retrieval [3], because they offer several advantages over real-valued descriptors. For example, (1) they enable us to utilize the fast bitwise operation in hardware; (2) they are hashable, allowing the use of memory in exchange for fast retrieval; (3) they require much less storage.

There are many methods (as reviewed in Section II) to hash images into binary codes. Despite their availability, they mostly seek linear projections and thus cannot capture the underlying non-linearity inherent in the image data [4], [5]. Kernelized methods have this capability, but an appropriate kernel function, which may not exist, needs to be chosen [5].

Recently, there are increasing attempts [4], [6]–[9] to apply deep architectures (neural networks with at least 3 hidden layers), which could learn the non-linear relationship among image pixels, to hash images into binary codes. However, these methods mostly require class labels, yet obtaining class labels are expensive in the field of medical imaging. A recent study [2] labeled 2100 medical images in collaboration with a radiologist, while the unlabeled data involved was 1 million. A very small amount of labeled data is unlikely to be adequate for training deep architectures. It may lead to over-fitting if over-trained. From the practical perspective, unsupervised methods should therefore be explored. Semantic hashing [10], [11] is a recent work that builds a deep autoencoder [12] by stacking restricted boltzmann machines (RBMs) [13], to learn binary codes from documents [10] or images [11] for retrieval. However, building a deep autoencoder by stacking RBMs for binary hashing [11] is complicated since a specific RBM training algorithm and an ad-hoc regularization technique are needed. Recently, it is reported that a deep autoencoder built by stacking de-noising autoencoders, i.e. a deep de-noising autoencoder (DDA), has comparable performance in learning features for supervised classification task [14]. However, the unsupervised method using a DDA to hash images into binary codes has not been explored.

Deep architectures have been applied in medical image analysis [15]–[18]. There are also some methods [2], [19] applying deep architectures for medical image retrieval but they are not hashing the medical images into binary codes, except one recent study [1], based on supervised methods

using kernel. To our knowledge, no studies have used deep architectures to hash medical images into binary codes without class labels for retrieval.

Traditional methods seem to be unable to capture the underlying non-linearity inherent in images. Kernelized methods, on the other hand, are not adaptable. Most methods based on deep architectures require class labels. Obtaining class labels for medical images is a laborious task and not easily doable for large databases. The methods which do not need class labels train a deep autoencoder by adopting a specific training algorithm and an ad-hoc regularization technique. Until now no studies have used deep architectures to hash medical images into binary codes without class labels for retrieval. Hence, one objective of this study is to introduce a new unsupervised scheme for training a deep de-noising autoencoder (DDA) to hash images into binary codes using backpropagation and standard regularization techniques. As well, we will investigate the feasibility and performance of using a DDA to hash X-ray images into binary codes without class labels.

Indicated by our experimental results on the benchmark dataset IRMA from ImageCLEFmed09 [20], the new unsupervised training scheme introduced by us had a significant impact on the retrieval performance, decreasing the total error by 21.9%. Using the same Exhaustive Search strategy, the binary codes learned by our DDA were found to achieve the lowest total error on this dataset comparing to other unsupervised methods reported in the literature. Using the fast retrieval strategy, we required only 7.18ms to retrieval a image on a laptop, achieving a speed-up of 9.27x over the Exhaustive Search strategy with a comparable total error. The best unsupervised retrieval result on this dataset was achieved by the combined use of RABC and DDA binary codes.

The rest of this paper is organized as follows: The related work is reviewed in Section II. Our methodology is described in Section III. The experimental results and their analysis are shown in Section IV. The whole study is summarized and concluded in Section V.

II. Related work

A. Hashing Images into Binary Codes

The existing methods to hash images into binary codes can be categorized into two classes: data-independent and data-dependent (or learning-based) methods. Data-independent methods do not look into the data distribution when hashing images into binary codes. One popular method in this class is Locality Sensitive Hashing (LSH) [21]. It uses random projections to construct hash functions such that similar images would have the same hash codes with high probability. However, it works only with theoretical guarantees for some metric spaces, e.g. l_0, l_2 and Jaccard distances [4], [22].

Data-depending methods investigate the data distribution when hashing images into binary codes. They are also called learning-based methods. This class of method can be categorized into two sub-classes: unsupervised and supervised. For the unsupervised sub-class, no class label is required to learn the hash functions. For example, Spectral Hashing [23] is to obtain balanced binary codes by solving a spectral graph partitioning problem, but the input data is assumed to be uniformly distributed in R^d [3]. Kernelized Locality-Sensitive Hashing (KLSH) [24] formulates the random projections necessary for LSH in kernel space using a subset of samples, such that the underlying embedding of the data needs not to be explicitly known and computable [3], but an appropriate kernel is required and its scalability is questioned [5]. For the supervised sub-class, class labels are required to learn the hash functions. Labeled training data can be used for constructing a pairwise similarity matrix to learn the hashing functions. Binary Re-constructive Embedding (BRE) [25] is to minimize the deviation between the original Euclidean distances and the learned Hamming distances. Minimum Loss Hashing (MLH) [26] is to minimize the difference between the learned Hamming distance and the binary quantization error. Kernel-based Supervised Hashing (KSH) [27] map samples into binary codes whose Hamming distances are minimized on similar pairs and maximized on dissimilar pairs.

Nevertheless, these approaches require the use of class labels, which are expensive to acquire if the data is large. Note that some methods only require a subset of training data having class labels. However, they are prone to over-fitting when labeled data is small [22]. Semi-supervised hashing (SSH) [22] minimizes empirical error for the pairwise similarity in the training samples, regularized by maximizing the variance of the labeled and unlabeled data. But these methods mostly seek linear projections and thus cannot capture the non-linear structure of samples [4], [5]. Kernelized methods could capture the underlying non-linearity, but an appropriate kernel function, which may not exist, needs to be chosen [5].

B. Hashing by Deep Architectures

Recently, there are increasing attempts [4], [6]–[9] to use deep architecture to hash images into binary codes. Most of these methods require class labels. They utilize Convolutional Neural Network (CNN) [4], [8], [9] to conduct multi-class classification, followed by binarizing the activations of a fully-connected layer with a threshold and taking the binarized results as hash codes. However, as mentioned before, obtaining class labels is an expensive task for large datasets, and these methods are likely to overfit if over-trained on a few labeled data [22]. On the other hand, there are limited attempts in using deep architectures to hash images into binary codes without using labels for retrieval. Semantic hashing [10], [11] is a recent approach that builds a deep autoencoder [12] by stacking restricted Boltzmann machines (RBMs) [13], to learn binary codes from documents [10] or images [11] for retrieval. However, building a deep autoencoder by stacking RBMs for binary hashing [11] is complicated because 1) a specific RBM training algorithm called contrastive divergence is needed, and 2) An ad-hoc regularization technique, i.e. binarizing the forward activities in the coding layer [11], needs to be adopted. Recently, it has been reported that a deep autoencoder built by stacking de-noising autoencoders, i.e. a deep de-noising autoencoder (DDA), has a comparable

performance in learning features for supervised classification task [14]. To our knowledge, there are no studies on how to use a DDA to hash images into binary codes.

C. Hashing Medical Images into Binary Codes

There is a small number of studies [28]–[33] involving hashing medical images into binary codes, and to a lesser extent for image retrieval [28], [29], [33]. Rather than hashing the entire images into binary codes, the research community in medical image retrieval adopts localized methods such as obtaining binary descriptors from the images, assuming that some local regions are more important in medical images [32]. Local Binary Pattern (LBP) descriptor [34] has been most commonly used for similar purposes [28], [29], [33]. Recently, a new descriptor called "Radon Barcode" was proposed [33], [35], which obtains binary codes from a medical image by binarizing its Radon projections. It reported superior performance than LBP in X-ray images. We compare the binary codes learned by our method with these methods, and study if it is possible to learn binary codes from Radon projections.

III. Methodology

A. Problem Definition

Let $X = \{x_n\}_{n=1}^{N}$ be a set of N X-ray images, where each X-ray image x_n is reshaped from a two-dimensional matrix to a one-dimensional vector. Given X, the problem is to learn a mapping $F : X \rightarrow \{0,1\}^{N\times k}$, i.e. to map a X-ray image x_n to a k-bit binary code $b_n \in \{0,1\}^k$, such that the semantic similarity between the images is preserved in the binary codes.

B. Proposed Method

The overall idea of the proposed method is to use autoencoders to learn high-level features from X-ray images in an unsupervised manner. As well, we will use thresholding to binarize the high-level features into binary codes. How we use the binary codes for X-ray images retrieval is illustrated in Fig 1. Here we use a specific type of autoencoder, namely de-noising autoencoder [14], to reduce learning features from noises. We also stack multiple de-noising autoencoders into a deep architecture called deep de-noising autoencoder (DDA) [14] to enhance feature learning capability.

In general, to train a DDA for hashing X-ray images into binary codes, there are four steps:

Step 1: Image Pre-processing. All images are first resized to a small size (default: 32 $\times$ 32). As X-ray images are in grayscale, through dividing all intensity values by 255, they are normalized to be in $[0, 1]$.

Step 2: Unsupervised Layer-by-layer Training. (Algorithm 1) A DDA is constructed by first training each layer as an individual de-noising autoencoder by backpropagation, as reported in [14]. A dropout layer [36] is introduced after the input layer, such that for each training sample, a randomly chosen subset (default: 20%) of the inputs is set to zero. Note that this drop-out layer has no effects on testing.

Algorithm 1 layerTrain

Input: a set of training images X, a list of fan-in/fan-out of each layer in encoder L_{enc} {e.g. L_{enc} = [(1024,768),(768,512)]}
Output: a list of encoder weights W_{enc}, a list of decoder weights W_{dec} { e.g. W_{enc} = [$W_{enc}^{(1)}$, $W_{enc}^{(2)}$], W_{dec} = [$W_{dec}^{(1)}$, $W_{dec}^{(2)}$]}
{Constants}
epoch = 100, batchSize = 16, p = 0.2
{Initialization}
W_{enc} = [], W_{dec} = []
{Train layer by layer}
for i from 1 to size(L_{enc}) **do**
  encFanIn, encFanOut = $L_{enc}[i]$
  decFanOut, decFanIn = $L_{enc}[i]$
  {Set up a NN with a dropout layer, an input-hidden layer and a hidden-output layer, with weights initialized by the scheme in [37]}
  Q = setUpLayers([DropoutLayer(encFanIn, encFanIn, p), SigmoidLayer(encFanIn, encFanOut), SigmoidLayer(decFanIn, decFanOut)])
  {Train the network on X for epoch times}
  for e from 1 to epoch **do**
    trainNN(Q, X, batchSize) {Train NN by backpropagation with a mini-batch of batchSize via RMSProp [38]}
  end for
  {Add the weights at the end of the weight lists}
  append(W_{enc}, getEncoderWeights(Q));
  append(W_{dec}, getDecoderWeights(Q));
end for
{Return W_{enc}, W_{dec}}
return W_{enc}, W_{dec}

Step 3: Unsupervised Fine-Tuning with Dropout. (Algorithm 2) After unsupervised layer-by-layer training, the de-noising autoencoders are stacked one-by-one to construct a DDA. According to [10], the last layer of the decoder is turned into a softmax layer. We introduce one new change to the architecture to improve the hashing performance. A dropout [36] layer is added before the coding layer to regularize it, as shown in Fig. 2. A Dropout layer has been reported to be a way to add noise [14], by randomly setting the output of hidden units to be zero. Note that this dropout layer has no effects in testing. After these changes, the DDA is then trained by backpropagation.

Step 4: Decoder Removal After the training, the decoder in the DDA is removed. Then, the DDA becomes a binary hashing function for X-ray images. To hash an image into binary codes, a normalized image, as a one-dimensional real-valued vector is fed into the trained DDA. After passing through all the layers, a one-dimensional real-valued vector outputted by the last sigmoid layer of the trained DDA is obtained. A threshold (>0.5) is then applied on this real-valued

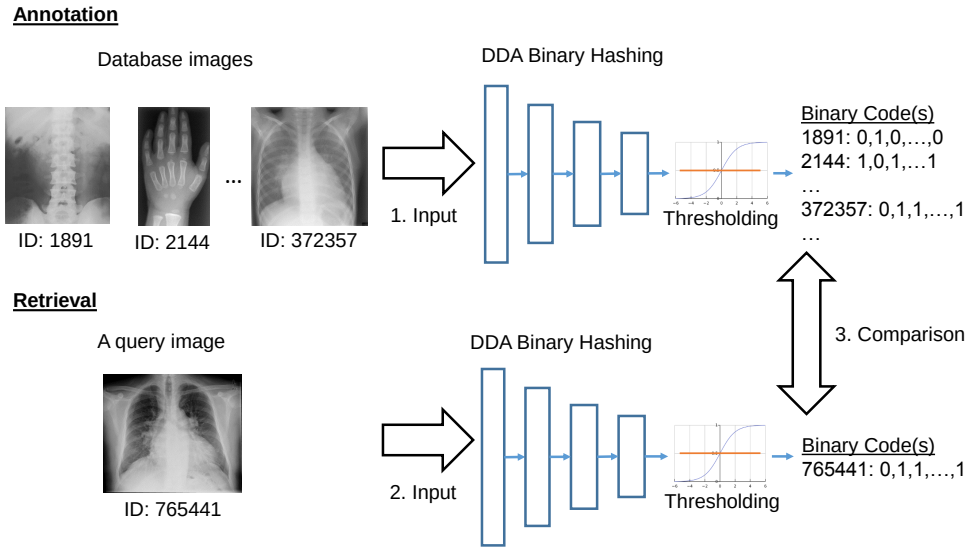

Fig. 1. Deep de-noising autoencoder (DDA). For all images, we use a trained DDA to annotate each of them with a k-bit binary code. Given a query image, we use the same DDA to annotate it with a k-bit binary code, followed by comparing this binary code with the binary codes of all database images.

vector to obtain a binary code vector.

C. X-ray Image Retrieval using Binary Codes

The retrieval of X-ray images is to identify the most similar images in the databases corresponding to a user's input X-ray image. It should be noted that the images in the databases and the users' input images are called training images and testing images, respectively. The following two X-ray image retrieval strategies are studied.

1) Exhaustive Search: After training a DDA, all training images are assigned with binary codes by it. For retrieval, the trained DDA is used for assigning a binary code to a given testing image. The binary code of the testing image is then compared with all binary codes of the training images. The ones with the least number of bit difference (shortest Hamming distance) are retrieved. The entire process is depicted in Fig 1. In our experiments, we only retrieved the first hit (the most similar image).

2) Semantic Hashing (SH): If this strategy is adopted, two DDAs are trained, where one is for hashing X-ray images into short binary codes (16 bits), and another is for hashing X-ray images into long binary codes (512 bits). All training images are assigned with both short and long binary codes. A hash table is then used for hashing the training images using the short binary codes as hash keys. Note that multiple X-ray images can share the same hash key.

In retrieval, the two DDAs are used to obtain both the short and long binary codes for a given test image. A number of bit flips, e.g. H, is then conducted on the short binary code of the test image to obtain a new short binary code. We repeat the process until all short binary codes with H bit differences are obtained. Using the short binary code of the test image with the short binary codes with H bit differences, a list of X-ray images are retrieved. The long binary code of the test image is then compared with those of the retrieved X-ray images. The ones with a small number of bit differences are retrieved. In our experiments, we only retrieved the first hit.

IV. Results

A. Dataset

To evaluate the retrieval performance, we used a benchmark dataset called IRMA (Image Retrieval Medical Applications) dataset as part of ImageCLEFmed09 initiative [20] which has 12677 images for training and 1733 images for testing, all images classified using IRMA codes. There are in total 193 classes. Each class is associated with a unique IRMA code. An IRMA code is mainly used for evaluating content-based medical image retrieval performance (hence not available in the real world). It is a string of 13 characters, where each of them is within the set of {0,...,9,a,...,z}. The 13 characters in an IRMA code are divided into four structures, in the following format: TTTT-DDD-AAA-BBB, where T, D, A and B mean technical, directional, anatomical and biological codes respectively. It should be noted that these IRMA codes are classified by professionals for benchmarking. In the real world, no medical images have associating IRMA codes.

B. Evaluation Metric

To evaluate the performance of image retrieval, we used the formula provided by ImageCLEFmed09 to compute the error between the IRMA codes of the testing image and the first hit retrieved training image. We then summed up the error for all testing images. The formula is provided as follows:

$$E_{Total} = \sum_{m=1}^{1733} \sum_{j=1}^{4} \sum_{i=1}^{l_j} \frac{1}{b_{l_j,i}} \frac{1}{i} \delta(I_{l_j,i}^m, \tilde{I}_{l_j,i}^m) \quad (1)$$

Here, m is an indicator to each image. j is an indicator of the structure of an IRMA code. l_j refers to the number of

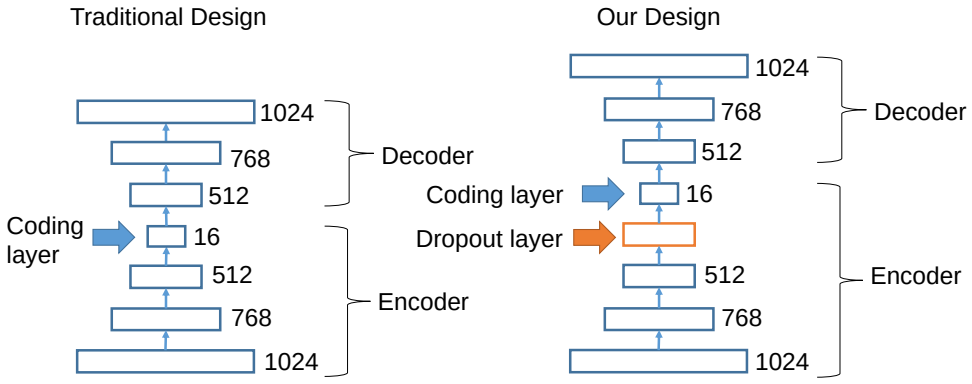

Fig. 2. This figure shows the difference between the traditional design (left) and our design (right) of a deep (de-noising) autoencoder (DDA) with the following architecture. We added a dropout layer before the coding layer to regularize it. A Dropout layer has been reported to be a way to add noise [14]. Note that the dropout layer is only effective in training and has no effects in testing. We trained this entire DDA by Methodology Step 3 Unsupervised Fine-tuning.

characters in each structure of an IRMA code. For example, in the IRMA code: 1121-4a0-914-700, $l_1 = 4$, $l_2 = 3$, $l_3 = 3$ and $l_4 = 3$. i is an indicator to a character in a particular structure. Here, $l_{2,2}$ refers to the character âĂŸaâĂŹ and $l_{4,1}$ refers to the character âĂŸ7âĂŹ. $b_{l_j,i}$ refers to the number of branches, i.e. number of possible characters, at the position i in the l_j^{th} structure in an IRMA code. I^m refers to the m^{th} testing image and $\tilde{I}^m$ refers to its top 1 retrieved image. $\delta(I^m_{l_j,i}, \tilde{I}^m_{l_j,i})$ compares a particular position in the IRMA code of the testing image and the retrieved image. It then outputs a value in {0, 1} according to the following rules.

$$\delta(I^m_{l_j,i}, \tilde{I}^m_{l_j,i}) = \begin{cases} 0, & I^m_{l_j,h} = \tilde{I}^m_{l_j,h} \forall h \leq i \\ 1, & I^m_{l_j,h} \neq \tilde{I}^m_{l_j,h} \exists h \leq i \end{cases} \quad (2)$$

We used the Python implementation of the above formula provided by ImageCLEFmed09 to compute the errors.

C. Implementation and Parameter Setting

The deep learning library Keras (http://keras.io/) with Theano backend [40] is adopted for implementation. The parameter setting is described as follows: The dropout parameter was set as 0.2, i.e. 20% of the inputs would be randomly set as zeroes. For both Methodology Step 2 and Step 3, the number of epochs and batch size were 100 and 16, respectively. The default loss function was binary cross-entropy. The default optimizer in Methodology Step 2 was RMSProp [38]. The default optimizer in Methodology Step 3 was RMSProp [38] for short codes and Adam [39] for long codes. These optimizers were used with default settings. All the experiments were run on a computer with 8.0 GB RAM, a i5-2410M-2.30GHz CPU (4 Cores) and a GT520M Graphics card. The neural networks were trained on the GPU. These settings were used in all experiments unless further specified.

D. Experiment Series 1

The objective of this experiment to investigate whether the binary codes tagged by a DDA on X-ray images can be used for image retrieval without using class labels. First, we pre-processed the training and testing images, according to Step 1 in the Methodology section. Second, according to Steps 2 and 3, we trained a DDA on the training images. Third, using Step 4, we used them to tag all training and testing images with binary codes. Afterward, we studied the X-ray image retrieval performance using these binary codes, according to an exhaustive search strategy mentioned in Section III-C1. We evaluated the performance using Equations 1 and 2. The results are shown in Table I. There are two types of DDA, where one is for short binary codes (16 bits) with the encoder architecture: 1024 inputs $\rightarrow$ 768 sigmoid neurons $\rightarrow$ 512 sigmoid neurons $\rightarrow$ Dropout $\rightarrow$ 16 sigmoid neurons and another is for long binary codes (512 bits) with the architecture: 1024 inputs $\rightarrow$ 768 sigmoid neurons $\rightarrow$ Dropout $\rightarrow$ 512 sigmoid neurons. Note that the dropout layer is only useful in training but not in testing. For comparison, the X-ray image retrieval errors of other binary codes such as Radon Barcode (RBC) [33], Local Binary Pattern (LBP) [34] and Local Radon Binary Pattern (LRBP) [41] on the same IRMA dataset using the Exhaustive Search strategy are listed from [33] for reference.

TABLE I
A COMPARISON OF THE IMAGE RETRIEVAL PERFORMANCE BETWEEN THE BINARY CODES LEARNED FROM DEEP DE-NOISING AUTOENCODER (DDA) AND OTHER METHODS FOR IRMA IMAGES.

Binary Code / Method	Label needed	Length of code	E_{Total}
TAUbiomed [42]	Yes	N/A	169.5
$DDA_{1024\rightarrow768\rightarrow512\rightarrow16}$	No	16	703.95
$DDA_{1024\rightarrow768\rightarrow512}$	**No**	512	**344.08**
RBC_4 [33]	No	512	476.62
RBC_8 [33]	No	1024	478.54
RBC_{16} [33]	No	2048	470.57
RBC_{32} [33]	No	4096	475.62
LBP [34]	No	7200	463.81
$LRBP_4$ [41]	No	7200	483.54
$LRBP_{32}$ [41]	No	7200	501.96

As shown in Table I, the 512-bit binary code learned by DDA has achieved a lower error total (E_{Total}) comparing to all the other binary codes including the latest developed

Algorithm 2 fineTune

Input: a set of training images X, a list of fan-in/fan-out of each layer in encoder L_{enc}, a list of encoder weights W_{enc}, a list of decoder weights W_{dec} {e.g. L_{enc} = [(1024,768),(768,512)], W_{enc} = [$W_{enc}^{(1)}$, $W_{enc}^{(2)}$], W_{dec} = [$W_{dec}^{(1)}$, $W_{dec}^{(2)}$]}
Output: a trained DDA Q
{Constants}
epoch = 100, batchSize = 16, p = 0.2
{Initialization}
encoderLayers = [], decoderLayers = [];
{Set up encoder layers}
for i from 1 to size(L_{enc}) **do**
  encFanIn, encFanOut = L_{enc}[i]
  if i == size(L_{enc}) **then**
    appendLayers(encoderLayers,
    [DropoutLayer(encFanIn, encFanIn, p)])
  end if
  appendLayers(encoderLayers,
  [SigmoidLayer(encFanIn, encFanout, $W_{enc}[i]$)])
end for
{Set up decoder layers}
for j from size(L_{enc}) downto 1 **do**
  decFanOut, decFanIn = L_{enc}[j]
  if j == size(L_{enc}) **then**
    appendLayers(decoderLayers,
    [SoftmaxLayer(decFanIn, decFanout, $W_{dec}[j]$)])
  else
    appendLayers(decoderLayers,
    [SigmoidLayer(decFanIn, decFanout, $W_{dec}[j]$)])
  end if
end for
{Set up the NN}
Q = setUpLayers(merge(encoderLayers, decoderLayers))
{Train the network on X for epoch times}
for e from 1 to epoch **do**
  trainNN(Q, X, batchSize) {Train NN by backpropagation with a mini-batch of batchSize via RMSProp [38] for short binary codes and Adam [39] for long binary codes}
end for
{Return a trained DDA}
return Q

Radon Barcode [33]. We also observe that even the length of binary codes has increased, the error total is still much higher compared to the 512-bit binary code. Note that the lowest error total achieved in this dataset is 169.5 by TAUbiomed [42] which requires class labels. No class information was used in ours and the rest in Table I.

E. Experiment Series 2

The objective of this experiment was to investigate whether Semantic Hashing (SH) (Section III-C2) can be applied on the X-ray images to speed up the retrieval process. We used the 16-bit and 512-bit DDA binary codes for hashing and re-ranking, respectively. The number of bit flips was to be 1, 2 and 3. For comparison, we also recorded the error total and retrieval time per image for a baseline Pearson Correlation method. Given a test image, the Pearson Correlation Coefficient was computed between the test image and every training image. The training image associated with the highest absolute Pearson Correlation Coefficient was retrieved. This baseline Pearson Correlation method was studied on both 32 $\times$ 32 and 64 $\times$ 64 images. The results are shown in Table II.

The Table II records the training time, error total and retrieval time per image for different methods. Note that the training time and the retrieval time per image were the averages of 20 independent runs of the full testing set (starting with the same random seed), along with the 95% confidence interval. As shown, Semantic Hashing with 2-bits flip has achieved a speedup of 9.27x over the Exhaustive Search Strategy via 512-bit binary codes learned from a DDA with a comparable total error, and a speed-up of 411.32x over the baseline Pearson Correlation method with a lower error total.

F. Experiment Series 3

The objective of this experiment was to investigate whether we can (1) learn binary codes from Radon projections [33] of X-ray images and (2) use them for boosting the image retrieval performance. We first computed the Radon projections of all training and test images with the size of 256 $\times$ 256 on 16 projections such that each image has a 4096-dimension real-valued vector. To obtain the binary code, it was proposed in [33] to use the median value of each projection to binarize the Radon projections, known as Radon Barcode (RBC) [33]. In this experiment, we constructed a de-noising autoencoder with the encoder architecture: 4096 inputs $\rightarrow$ Dropout $\rightarrow$ 2048 sigmoid neurons. We trained this de-noising autoencoder on the Radon projections using the default parameter setting, except that in Methodology Step 3 we set the epoch to be 2200. We name the binary code learned as Radon Autoencoder Barcode (RABC). We further define the bit difference between a test image I^m and a candidate image I^c as $\frac{d_{RABC2048}(I^m,I^c)}{2048} + \frac{d_{DDA512}(I^m,I^c)}{512}$, where the former is the normalized bit difference between the 2048-bit RABC of the testing and the candidate image, and the latter is the normalized bit difference between the 512-bit DDA binary code of the testing and the candidate image. The image retrieval results under the Exhaustive Search strategy are shown in Table III.

As shown, the error total achieved by RABC is significantly lower than that of RBC, even though the length of binary code of former is half of the latter. This shows that the binary codes learned from the Radon projections contain much compact information. The best image retrieval performance was achieved by the combined use of RABC and the 512-bit DDA binary codes learned in Experiment 1. This shows the effectiveness of RABC in improving image retrieval performance. A demonstration of binary codes is in Fig. 3.

TABLE II
A COMPARISON OF THE IMAGE RETRIEVAL PERFORMANCE BETWEEN EXHAUSTIVE SEARCH STRATEGY, SEMANTIC HASHING (SH) AND PEARSON CORRELATION FOR IRMA IMAGES.

Binary code / method	Length of code	Training time (s)	Retrieval time (ms) per image	E_{Total}
$DDA_{1024\rightarrow768\rightarrow512\rightarrow16}$	16	5420.09±10.65	50.93±0.86	703.95
$DDA_{1024\rightarrow768\rightarrow512}$	512	5084.72±6.13	66.58±1.22	344.08
SH (Number of Bit Flips: 1)	16; 512	10504.81±14.02	5.62±0.11	414.80
SH (Number of Bit Flips: 2)	16; 512	10504.81±14.02	**7.18±0.12**	**378.30**
SH (Number of Bit Flips: 3)	16; 512	10504.81±14.02	24.91±0.76	365.24
Pearson Correlation (32 × 32)	N/A	N/A	2311.60±37.89	399.68
Pearson Correlation (64 × 64)	N/A	N/A	2587.91±30.34	402.05

TABLE III
COMPARISON OF IMAGE RETRIEVAL PERFORMANCE FOR IRMA IMAGES.

Binary code / method	Length of code	Training time (s)	Retrieval time (ms) per image	E_{Total}
$DDA_{1024\rightarrow768\rightarrow512}$	512	5084.72±6.13	66.58±1.22	344.08
$RBC_{256\times256,16proj}$	4096	N/A	162.50±0.37	546.09
$RABC_{4096\rightarrow2048}$	2048	about 270,600	109.28±0.29	362.54
$DDA_{1024\rightarrow768\rightarrow512}$/512 + $RABC_{4096\rightarrow2048}$/2048	512;2048	about 275,690	233.30±2.88	**330.60**

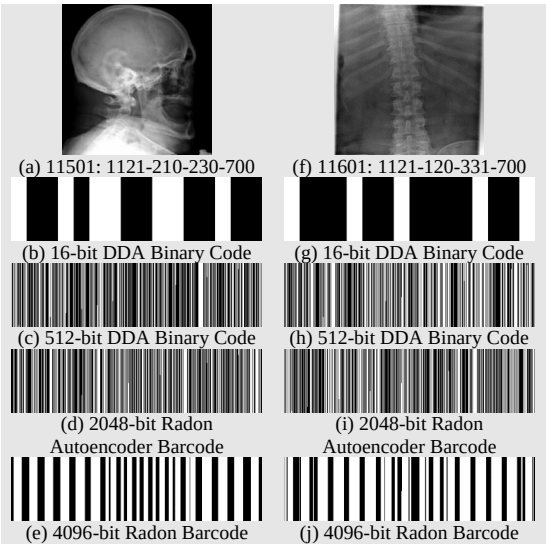

Fig. 3. This figure shows the IRMA code, 16-bit DDA Binary Code, 512-bit DDA Binary Code, 2048-bit Radon Autoencoder Barcode and 4096-bit Radon Barcode of image 11501 (a-e) and 11601 (f-j).

G. Experiment Series 4

The objective of this experiment was to investigate the effects brought by the dropout layer on the binary hashing performance of a DDA. In Methodology Step 3 (Unsupervised Fine-tuning), we introduced a change in the DDA architecture that is different from both the DDA mentioned in [14] and the very deep autoencoder mentioned in [11]. We added a dropout [36] layer before the coding layer, where 20% randomly chosen inputs to the coding layer would be set as zeros in training. We also used different optimizers (Adam [39] or RMSProp [38]) in Methodology Step 3 to identify the best training scheme.

We focused our experiment on the following DDA encoder architecture: 1024 inputs → 768 sigmoid neurons → Dropout → 512 sigmoid neurons. Based on this configuration, for each possible scenario, we started from the same set weights initialized by Methodology Step 2 and trained the DDA using Methodology Step 3. After learning the binary codes, we studied the image retrieval performance under the Exhaustive Search Strategy. The results are in Table IV.

TABLE IV
COMPARISON OF IMAGE RETRIEVAL PERFORMANCE

Configuration	Optimizer	E_{Total}
No Step 3	N/A	440.40
Step 3 without dropout	RMSProp	410.64
Step 3	RMSProp	**370.70**
Step 3 without dropout	Adam	561.47
Step 3	Adam	**344.08**

We observe that the dropout layer brought to the DDA has a significant improvement effect on the binary hashing performance. In Table III, the error total with no step 3 is 440.40. Using RMSProp as the optimizer, if we use Step 3 without dropout, the error is decreased by 29.76. If we use Step 3 (with dropout), the error is further decreased by 39.94. Using Adam as the optimizer, if we use Step 3 without dropout, the error is increased by 121.07, probably because of over-fitting. If we use Step 3 (with dropout), the error is decreased by 96.32, a reduction of total error by 21.9%. This indicates the importance of adding a dropout layer before the coding layer, independent of the optimizer used.

V. CONCLUSION

In this study, we explored the use of Deep De-noising Autoencoder (DDA) to hash X-ray images into binary codes without class labels. We introduced a new unsupervised training scheme by adding a dropout layer to the DDA architecture in the step of Unsupervised Fine-tuning. Conducting experiments on the benchmark dataset IRMA from ImageCLEFmed09, we observe that this is important for satisfactory binary hashing performance, reducing the total retrieval error by up to 21.9%. Moreover, we demonstrated an alternative in order to construct a deep (de-noising) autoencoder by stacking (de-noising) autoencoder directly for binary hashing, using only

backpropagation and dropout, simplifying the implementation. Furthermore, we developed Radon Autoencoder Barcode (RABC) and used it to improve retrieval performance. All these indicate the potential of our method for CBIR in practice, specifically for big image data like medical images whose labels are expensive to obtain.

REFERENCES

[1] X. Zhang, W. Liu, M. Dundar, S. Badve, and S. Zhang, "Towards large-scale histopathological image analysis: Hashing-based image retrieval," *Medical Imaging, IEEE Transactions on*, vol. 34, no. 2, pp. 496–506, 2015.

[2] J. E. Sklan, A. J. Plassard, D. Fabbri, and B. A. Landman, "Toward content based image retrieval with deep convolutional neural networks," in *SPIE Medical Imaging*. International Society for Optics and Photonics, 2015, pp. 94 172C–94 172C.

[3] K. Grauman and R. Fergus, "Learning binary hash codes for large-scale image search," in *Machine learning for computer vision*. Springer, 2013, pp. 49–87.

[4] J. Guo and J. Li, "Cnn based hashing for image retrieval," *arXiv:1509.01354*, 2015.

[5] V. Erin Liong, J. Lu, G. Wang, P. Moulin, and J. Zhou, "Deep hashing for compact binary codes learning," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2015, pp. 2475–2483.

[6] R. Xia, Y. Pan, H. Lai, C. Liu, and S. Yan, "Supervised hashing for image retrieval via image representation learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2014, pp. 2156–2162.

[7] W.-J. Li, S. Wang, and W.-C. Kang, "Feature learning based deep supervised hashing with pairwise labels," *arXiv:1511.03855*, 2015.

[8] K. Lin, H.-F. Yang, J.-H. Hsiao, and C.-S. Chen, "Deep learning of binary hash codes for fast image retrieval," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2015, pp. 27–35.

[9] R. Zhang, L. Lin, R. Zhang, W. Zuo, and L. Zhang, "Bit-scalable deep hashing with regularized similarity learning for image retrieval and person re-identification," *Image Processing, IEEE Transactions on*, vol. 24, no. 12, pp. 4766–4779, 2015.

[10] R. Salakhutdinov and G. Hinton, "Semantic hashing," *International Journal of Approximate Reasoning*, vol. 50, no. 7, pp. 969–978, 2009.

[11] A. Krizhevsky and G. E. Hinton, "Using very deep autoencoders for content-based image retrieval." in *ESANN*. Citeseer, 2011.

[12] G. E. Hinton and R. R. Salakhutdinov, "Reducing the dimensionality of data with neural networks," *Science*, vol. 313 (5786), pp. 504–507, 2006.

[13] G. Hinton, "A practical guide to training restricted boltzmann machines," *Momentum*, vol. 9, no. 1, p. 926, 2010.

[14] P. Vincent, H. Larochelle, I. Lajoie, Y. Bengio, and P.-A. Manzagol, "Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion," *The Journal of Machine Learning Research*, vol. 11, pp. 3371–3408, 2010.

[15] C. C. Tan and C. Eswaran, "Using autoencoders for mammogram compression," *Journal of medical systems*, vol. 35, no. 1, pp. 49–58, 2011.

[16] H.-C. Shin, M. R. Orton, D. J. Collins, S. J. Doran, and M. O. Leach, "Stacked autoencoders for unsupervised feature learning and multiple organ detection in a pilot study using 4d patient data," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 35, no. 8, pp. 1930–1943, 2013.

[17] N. Nayak, H. Chang, A. Borowsky, P. Spellman, and B. Parvin, "Classification of tumor histopathology via sparse feature learning," in *Biomedical Imaging (ISBI), 2013 IEEE 10th International Symposium on*. IEEE, 2013, pp. 410–413.

[18] Z. Camlica, H. Tizhoosh, and F. Khalvati, "Autoencoding the retrieval relevance of medical images," in *Image Processing Theory, Tools and Applications (IPTA), the fifth International Conference on*, 2015.

[19] S. Liu, W. Cai, Y. Song, S. Pujol, R. Kikinis, and D. Feng, "A bag of semantic words model for medical content-based retrieval," in *MICCAI Workshop on Medical Content-Based Retrieval for Clinical Decision Support*, 2013.

[20] T. Tommasi, B. Caputo, P. Welter, M. O. Güld, and T. M. Deserno, "Overview of the clef 2009 medical image annotation track," in *Multilingual Information Access Evaluation II. Multimedia Experiments*. Springer, 2010, pp. 85–93.

[21] A. Gionis, P. Indyk, R. Motwani *et al.*, "Similarity search in high dimensions via hashing," in *VLDB*, vol. 99, 1999, pp. 518–529.

[22] J. Wang, S. Kumar, and S.-F. Chang, "Semi-supervised hashing for large-scale search," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 34, no. 12, pp. 2393–2406, 2012.

[23] Y. Weiss, A. Torralba, and R. Fergus, "Spectral hashing," in *Advances in neural information processing systems*, 2009, pp. 1753–1760.

[24] B. Kulis and K. Grauman, "Kernelized locality-sensitive hashing for scalable image search," in *Computer Vision, 2009 IEEE 12th International Conference on*. IEEE, 2009, pp. 2130–2137.

[25] B. Kulis and T. Darrell, "Learning to hash with binary reconstructive embeddings," in *Advances in neural information processing systems*, 2009, pp. 1042–1050.

[26] M. Norouzi and D. M. Blei, "Minimal loss hashing for compact binary codes," in *Proceedings of the 28th international conference on machine learning (ICML-11)*, 2011, pp. 353–360.

[27] W. Liu, J. Wang, R. Ji, Y.-G. Jiang, and S.-F. Chang, "Supervised hashing with kernels," in *Computer Vision and Pattern Recognition (CVPR), 2012 IEEE Conference on*. IEEE, 2012, pp. 2074–2081.

[28] D. Unay, A. Ekin, and R. Jasinschi, "Medical image search and retrieval using local binary patterns and klt feature points," in *Image Processing, 2008. ICIP 2008. 15th IEEE International Conference on*. IEEE, 2008, pp. 997–1000.

[29] X.-c. Xu and Q. Zhang, "Medical image retrieval using local binary patterns with image euclidean distance (pp. 1–4)," in *International Conference on Information Engineering and Computer Science*, 2009.

[30] L. Nanni, A. Lumini, and S. Brahnam, "Local binary patterns variants as texture descriptors for medical image analysis," *Artificial intelligence in medicine*, vol. 49, no. 2, pp. 117–125, 2010.

[31] T. Kanumuri, M. Dewal, and R. Anand, "Progressive medical image coding using binary wavelet transforms," *Signal, Image and Video Processing*, vol. 8, no. 5, pp. 883–899, 2014.

[32] Z. Camlica, H. Tizhoosh, and F. Khalvati, "Medical image classification via svm using lbp features from saliency-based folded data," in *Machine Learning and Applications (ICMLA), The 14th International Conference on*, 2015.

[33] H. R. Tizhoosh, "Barcode annotations for medical image retrieval: A preliminary investigation," in *Image Processing (ICIP), 2015 IEEE International Conference on*, 2015, pp. 818 – 822.

[34] D. Huang, C. Shan, M. Ardabilian, Y. Wang, and L. Chen, "Local binary patterns and its application to facial image analysis: a survey," *Systems, Man, and Cybernetics, Part C: Applications and Reviews, IEEE Transactions on*, vol. 41, no. 6, pp. 765–781, 2011.

[35] H. Tizhoosh, M. Gangeh, and C. G. Tadayyon, H., "Tumour roi estimation in ultrasound images via radon barcodes in patients with locally advanced breast cancer," in *IEEE ISBI, Prague, Czech Republic*, 2016.

[36] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov, "Dropout: A simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[37] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *International conference on artificial intelligence and statistics*, 2010, pp. 249–256.

[38] T. Tieleman and G. Hinton, "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude," in *COURSERA: Neural Networks for Machine Learning*, 4 2012.

[39] D. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv:1412.6980*, 2014.

[40] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. Goodfellow, A. Bergeron, N. Bouchard, D. Warde-Farley, and Y. Bengio, "Theano: new features and speed improvements," *arXiv:1211.5590*, 2012.

[41] H. K. Galoogahi and T. Sim, "Face sketch recognition by local radon binary pattern: Lrbp," in *Image Processing (ICIP), 2012 19th IEEE International Conference on*. IEEE, 2012, pp. 1837–1840.

[42] U. Avni, J. Goldberger, and H. Greenspan, "Addressing the imageclef 2009 challenge using a patch-based visual words representation," in *Working Notes for the CLEF 2009 Workshop. The Cross-Language Evaluation Forum (CLEF), Corfu, Greece*, 2009.