Redundancy-resistant Generative Hashing for Image Retrieval

Changying Du\textsuperscript{1}, Xingyu Xie\textsuperscript{2}, Changde Du\textsuperscript{3}, Hao Wang\textsuperscript{1}\textsuperscript{*}
\textsuperscript{1} 360 Search Lab, Beijing 100015, China
\textsuperscript{2} College of Automation, Nanjing University of Aeronautics and Astronautics, Nanjing, China
\textsuperscript{3} Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
ducytict@gmail.com, nuaaxing@gmail.com, duchangde2016@ia.ac.cn, cashenry@126.com

Abstract

By optimizing probability distributions over discrete latent codes, Stochastic Generative Hashing (SGH) bypasses the critical and intractable binary constraints on hash codes. While encouraging results were reported, SGH still suffers from the deficient usage of latent codes, i.e., there often exist many uninformative latent dimensions in the code space, a disadvantage inherited from its auto-encoding variational framework. Motivated by the fact that code redundancy usually is severer when more complex decoder network is used, in this paper, we propose a constrained deep generative architecture to simplify the decoder for data reconstruction. Specifically, our new framework forces the latent hashing codes to not only reconstruct data through the generative network but also retain minimal squared L2 difference to the last real-valued network hidden layer. Furthermore, during posterior inference, we propose to regularize the standard auto-encoding objective with an additional term that explicitly accounts for the negative redundancy degree of latent code dimensions. We interpret such modifications as Bayesian posterior or regularization and design an adversarial strategy to optimize the generative, the variational, and the redundancy-resistingant parameters. Empirical results show that our new method can significantly boost the quality of learned codes and achieve state-of-the-art performance for image retrieval.

1 Introduction

Image retrieval aims to retrieve relevant images to a given query image from a very large image corpus. Due to the huge amount of computation and storage cost required for similarity evaluation in the original feature space, the core problem in this area has been how to re-represent each image using sufficiently compact codes while preserving the original similarity structure as more as possible.

Learning to hash (LTH) [Wang et al., 2017] is a popular kind of techniques developed for the aforementioned approx-

\*Corresponding author

imate nearest neighbor search problem. By using binary hash codes to represent the original data, the storage cost can be dramatically reduced. Furthermore, search with such binary representations can be efficiently conducted because (1) we can perform Hamming distance computation that is supported via POPCNT on modern CPU/GPU; and (2) we can achieve a constant or sub-linear time complexity for search by using hash codes to construct an index [Kong and Li, 2012].

Since the expressiveness of the learned hash codes can directly affect the final retrieval performance, improving code quality by capturing preferred properties of the hash function is a long-standing topic in LTH studies. Supervised hashing [Xia et al., 2014], semi-supervised hashing [Wang et al., 2012; Zhu et al., 2016], and cross-modal hashing [Jiang and Li, 2016] achieve this goal by assuming the images are labeled or tagged, which requires many human efforts or even be unrealistic for modern applications. In contrast, unsupervised hashing methods do not utilize any semantic relations explicitly, and thus are more challenging to be effective in practice. Representative work along this line include spectral hashing [Weiss et al., 2008], isotropic hashing [Kong and Li, 2012], iterative quantization [Gong et al., 2013], K-means hashing [He et al., 2013], discrete graph hashing [Liu et al., 2014], spherical hashing [Heo et al., 2015], graph hashing [Jiang and Li, 2015], binary autoencoder (BAE) hashing [Carreira-Perpinán and Raziperchikolaei, 2015], stochastic generative hashing (SGH) [Dai et al., 2017], etc. Among them, SGH and BAE enjoy the theoretical advantage of minimum description length (MDL) owing to their auto-encoding architecture. This property is critical to achieve not only fast but also accurate retrieval. Besides, due to its probabilistic nature, SGH avoids commonly used relaxation of the binary constraints (which often leads to inferior results) by optimizing probability distributions over discrete hash codes.

The apparent benefits claimed by SGH are inherited from the auto-encoding variational inference framework [Kingma and Welling, 2014; Rezende et al., 2014]. We note, however, that the MDL principle may be unsatisfied due to the prior constraints in this framework. In fact, recent studies show that such a variational autoencoder (VAE) framework suffers from the deficient usage of latent codes, i.e., there often exist many uninformative latent dimensions in the code space [Sønderby et al., 2016], especially when the decoder network is complex. This phenomenon is caused by the variational
pruning [Hoffman, 2017; Yeung et al., 2017], a problem occurs when many latent units collapse early in training before they learned a useful representation.

In this paper, we design a new deep generative architecture which forces the latent hashing codes to not only reconstruct data through the generative (decoder) network but also retain minimal squared L2 difference to the output of the last real-valued decoder hidden layer. We show that such a constrained deep structure has the effect of simplifying the decoder while providing sufficiently good generation/reconstruction quality. A simplified decoder network together with a deep inference (encoder) network are expected to alleviate variational pruning and hence code redundancy. Furthermore, during posterior inference, we propose to regularize the standard auto-encoding objective with an additional term that explicitly accounts for the negative redundancy degree of latent code dimensions. We interpret such modifications as Bayesian posterior regularization and design an adversarial strategy to optimize the generative, the variational, and the redundancy-resisting parameters. Experimental results show that our redundancy-resistant generative hashing framework can significantly boost the quality of learned codes and achieve state-of-the-art retrieval performance on image data sets.

## 2 Model

As a powerful unsupervised representation learning framework, VAE [Kingma and Welling, 2014; Rezende et al., 2014] defines a latent variable generative model as follows

\[
p(\mathbf{Z}) = \prod_{i=1}^{N} p(z_i), \quad p(\mathbf{X}|\mathbf{Z}) = \prod_{i=1}^{N} p(x_i|z_i)
\]

where \(\theta\) denotes the likelihood is parameterized by a decoder Deep Neural Network (DNN), and \(\mathbf{Z} \in \mathbb{R}^{K \times N}\) and \(\mathbf{X} \in \mathbb{R}^{D \times N}\) denote the latents and the observed data, respectively. Specifying \(q_{\phi}(\mathbf{Z}|\mathbf{X}) = \prod_{i=1}^{N} q_{\phi}(z_i|x_i)\) as the DNN-parameterized inference model (encoder), VAE seeks to maximize the evidence lower bound (ELBO) w.r.t. \(\phi\) and \(\theta\)

\[
\sum_{i=1}^{N} \log p(x_i) \geq \sum_{i=1}^{N} \mathbb{E}_{q_{\phi}(z_i|x_i)}[\log p(x_i|z_i)]
- \sum_{i=1}^{N} D_{KL}(q_{\phi}(z_i|x_i)||p(z_i)) = \text{ELBO}.
\]

### 2.1 Stochastic Generative Hashing

As an instance of VAE, SGH [Dai et al., 2017] uses a linear Gaussian likelihood and imposes the Bernoulli prior on \(\mathbf{Z}\):

\[
p(\mathbf{Z}) = \prod_{i=1}^{N} \prod_{j=1}^{K} \rho_j^{z_{ij}}(1-\rho_j)^{1-z_{ij}},
\]

where \(\rho = [\rho_j]_{j=1}^{K} \in [0, 1]^{K}\). Correspondingly, the inference model is parameterized as

\[
q_{\phi}(\mathbf{Z}|\mathbf{X}) = \prod_{i=1}^{N} \prod_{j=1}^{K} (\phi_j(x_i))^{z_{ij}}(1-\phi_j(x_i))^{1-z_{ij}},
\]

where \(\phi_j(x_i) = q_{\phi}(z_{ij} = 1|x_i)\), and \(\phi_j\) represents a scalar-valued linear or deep nonlinear transformation.

### 2.2 Redundancy-resistant Generative Hashing

To ensure efficient similarity search, we assume the latent hash code \(\mathbf{z}\) of each data to be binary, thus our Redundancy-resistant SGH (R-SGH) model shares the same Bernoulli prior \(p(\mathbf{Z})\) as in (1). Compared to SGH, one distinct feature of R-SGH lies in the new deep generative architecture for likelihood computation. Suppose the decoder DNN (generative network) has \(M+1\) layers, where 1) the first layer is the hash code layer, which has prior \(p(\mathbf{Z})\) and is the latent variable of our Bayesian model; 2) the last layer is the data reconstruction layer, whose output is used to form the mean and covariance parameters of our Gaussian likelihood on data; and 3) the \(M\)-th layer is the last real-valued decoder hidden layer, whose output is denoted by \(\mathbf{H}^{(M)} = [\mathbf{h}_i^{(M)}]_{i=1}^{N}\). Our idea is to force \(\mathbf{Z}\) to not only reconstruct data through the decoder network but also retain minimal squared L2 difference to the output of the \(M\)-th decoding layer. Thus we propose to add the regularization term \(-\|\mathbf{Z} - \mathbf{H}^{(M)}\|_F^2\) into the VAE objective, where \(\|\cdot\|_F\) denotes the Frobenius norm of a matrix, and we assume \(\mathbf{z}\) and \(\mathbf{h}^{(M)}\) have the same dimension.

Consider the extreme case that the above regularization term is perfectly optimized (\(\mathbf{Z}\) exactly equals to \(\mathbf{H}^{(M)}\)), i.e., the hash code layer is identical to the last decoder hidden layer, then our decoder model is equivalent to a single layer network. Thus, such a constrained deep structure has the effect of simplifying the decoder while providing sufficiently good generation/reconstruction quality. As studied in RNN-based variational language models [Yang et al., 2017], where the flexible LSTM decoder in VAE does not make effective use of the latent representation, a simplified decoder network together with a deep unconstrained inference (encoder) network are expected to alleviate variational pruning and hence code redundancy. For our R-SGH, we assume that the inference model has the same form as in (2), and \(\phi\) is implemented by an encoder DNN (which transforms the input data into Bernoulli parameter for hash code through \(M\) hidden layers) with mirrored layer structure as the decoder.

Furthermore, during posterior inference, we propose to regularize the standard auto-encoding variational objective with an additional term that explicitly accounts for the negative redundancy degree of latent code dimensions. The key insight is that, if a latent dimension can be approximated by the effective use of the latent representation, a simplified decoder network together with a deep unconstrained inference (encoder) network are expected to alleviate variational pruning and hence code redundancy. For our R-SGH, we assume that the inference model has the same form as in (2), and \(\phi\) is implemented by an encoder DNN (which transforms the input data into Bernoulli parameter for hash code through \(M\) hidden layers) with mirrored layer structure as the decoder.

Before we design specific adversarial algorithm to optimize \(\theta, \phi\) and \(\mathbf{A}\) alternately, we first interpret the formulation in (3) as Bayesian posterior regularization [Zhu et al.,...
2014]. We first define an auxiliary function
\[ l(Z; \eta, \delta, A) = \exp \{ \eta \|Z - H(M)\|^2_F - \delta \|Z^T - Z^T A\|^2_F \}, \]
then given \( A \) the formulation in (3) can be rewritten as
\[ \max_{\theta, \phi} ELBO - \mathbb{E}_{q_{\phi}(Z;X)}[\log l(Z;\eta, \delta, A)]. \]  
Solving problem (4), we can get the posterior distribution
\[ q_{\phi}(Z|X) = \frac{p_{\theta}(X|Z)p(Z|l(Z;\eta, \delta, A))}{p(X)}. \]
As a direct way to impose constraints or incorporate knowledge in Bayesian models, Bayesian posterior regularization is more natural and general than specially designed priors.

### 2.3 Adversarial Training

The VAE parameters \( \theta, \phi \) aim to maximize (3), while the redundancy-resisting parameter \( A \) plays an adversarial role to minimize it. To optimize them in turn, we have to formulate each sub-problem separately.

#### Optimizing \( \theta \)

Given \( \phi \) and \( A \), problem (3) simplifies to
\[ \max_{\theta} \sum_{i=1}^{N} \mathbb{E}_{q_{\phi}(Z|X)} \log p_{\theta}(X_i|Z_i), \]  
which is straightforward to solve utilizing the reparameterization trick introduced in [Dai et al., 2017]. Specifically, to deal with the discrete variational distribution, we use the Doubly Stochastic Neuron (DSN) that achieves the hash code variable by \( z_{ij} = f(\phi_j(x_i), \xi_j, \varepsilon_j) \), where \( x_i \) is a vector with its \( i \)-th element being the \( z_{ij} \).

\[ f(e, \xi_j, \varepsilon_j) = \begin{cases} 1, & \text{if } e > \xi_j, \\ 1_{e > \varepsilon_j}, & \text{if } e = \xi_j, \\ 0, & \text{if } e < \xi_j. \end{cases} \]

Hence, the objective in (5) can be rewritten as
\[ \max_{\theta} \sum_{i=1}^{N} \mathbb{E}_{\xi_i \varepsilon} \log p_{\theta}(X_i|f(\phi_j(x_i), \xi, \varepsilon)), \]  
where \( x_i = f(\phi_j(x_i), \xi, \varepsilon) = [f(\phi_j(x_i), \xi_j, \varepsilon_j)]_{j=1}^{K} \). Then it is similar as in the vanilla VAE to derive a Monte Carlo estimator for the gradient of the objective in (6) w.r.t. \( \theta \).

#### Optimizing \( \phi \)

Given \( \theta \) and \( A \), problem (3) simplifies to
\[ \max_{\phi} \mathbb{E}_{q_{\phi}(Z|X)} [\delta \|Z^T - Z^T A\|^2_F - \eta \|Z - H[M+1]\|^2_F] + ELBO, \]  
which can also be solved using the reparameterization trick. However, we empirically note that such direct treatment often leads to poor performance since a optimum found this way is easily dominated by the first term. The underlying reason is that the ELBO is upper-bounded while the first term can be arbitrarily large. To this end, we introduce the \( \varepsilon \)-insensitive idea, and reformulate the problem as
\[ \max_{\phi} \text{ELBO} + \delta \cdot \mathbb{E}_{q_{\phi}(Z|X)} \left[ \min_{k=1}^{K} \frac{\|p_{k}\|^2_N}{N} - \varepsilon, 0 \right] - \eta \cdot \mathbb{E}_{q_{\phi}(Z|X)} \|Z - H[M+1]\|^2_F, \]  
where \( p_k \) is the \( k \)-th column of \( R = Z^T - Z^T A \). Following [Dai et al., 2017], we employ the distributional derivative of DSN to derive the gradient of the objective in (8) w.r.t. \( \phi \).

### 2.4 Complexity Analysis

Using the gradient estimators of \( \{\theta, \phi\} \) and the analytical solution of \( A \), the computational complexity of a single joint update of the parameters \( \{\theta, \phi, A\} \) for the proposed model is \( O(NC_{nn} + K^3) \), where \( N \) is the batch size during our mini-batch training, and \( C_{nn} \) is the cost of evaluating the encoder/decoder networks. After training, all parameters are frozen, and we are interested in encoding a query image to its hash code, a procedure of the same complexity \( O(C_{nn}) \) as other deep hashing methods.

### 3 Experiments

We evaluate the proposed method on two computer vision tasks: 1) Image generation/reconstruction on MNIST [Oliiva and Torralba, 2001]; 2) Image retrieval on CIFAR-10 [Krizhevsky, 2009] and Caltech-256 [Griffith et al., 2007].

#### 3.1 Experimental Setup

**Compared Methods**

- (1) Spectral Hashing (SH) [Weiss et al., 2008];
- (2) Iterative Quantization (ITQ) [Gong et al., 2013];
- (3) PCA Hashing (PCAH), which projects the input via PCA and performs binarization according to the sign of each dimension;
- (4) Stochastic Generative Hashing (SGH) [Dai et al., 2017]; and
- (5) Deep-SGH, a variant of SGH that adopts DNN as the encoder and decoder.
Parameter Settings For the compared methods, we use the implementations provided by their authors (Deep-SGH is implemented directly based on SGH) and set the parameters according to their original papers. Without explicit statement, 1) for our R-SGH, the prior parameter $\rho_j$ is set to 0.5 for any $j \in \{1, ..., K\}$, the threshold parameter $\epsilon$ is set to 0.05, and both $\delta$ and $\eta$ are set to 0.01; and 2) for R-SGH and Deep-SGH, the encoder and decoder network structures are set as $[D-K-K-K]$ and $[K-K-K-D]$ respectively, where $D$ and $K$ are the dimensions of input data and hash code respectively. Note, the 4-layer encoder and decoder share the same hash code layer, and all hidden layers have the same dimension.

Evaluation Metrics (1) L2 reconstruction error; (2) precision at $S$: the percentage of true relevant instances among top $S$ retrieved ones; and (3) recall at $S$: the percentage of true relevant instances that are retrieved as the top $S$ similar ones; (4) mean average precision (mAP), which computes the area under the entire precision-recall curve and evaluates the overall retrieval performance of different hashing algorithms.

Consistently improve the overall retrieval performance. Second, R-SGH keeps a greater growth of mAP than that of SGH when increasing number of hash bits are used, this is because the redundancy-resistant terms in R-SGH can effectively activate the uninformative bit in the generative model. Third, R-SGH outperforms the other compared methods by a large margin when 128 bits of hash code are used. We attribute this to the fact that higher dimensional latent representations come with higher information redundancy. Finally, we note that SGH and ITQ achieve better recall and precision performances when 32 bits hash code are used. This is due to very little information redundancy contained in the hash code when the number of hash bits is small. In this case, the redundancy-resistant regularization influences the reconstruction procedure and may degrade the performance. Fortunately, if low redundancy is expected in advance, we can prevent the negative impact by setting a small threshold parameter $\epsilon$.

Figure 1: Reconstruction error vs. the number of iterating epochs.

<table>
<thead>
<tr>
<th>Method</th>
<th>32 bits</th>
<th>64 bits</th>
<th>128 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITQ</td>
<td>22.34</td>
<td>27.45</td>
<td>29.28</td>
</tr>
<tr>
<td>SH</td>
<td>6.97</td>
<td>12.23</td>
<td>19.39</td>
</tr>
<tr>
<td>PCAH</td>
<td>7.79</td>
<td>11.33</td>
<td>15.72</td>
</tr>
<tr>
<td>SGH</td>
<td>23.86</td>
<td>30.56</td>
<td>35.61</td>
</tr>
<tr>
<td>Deep-SGH</td>
<td>21.96</td>
<td>29.98</td>
<td>34.77</td>
</tr>
<tr>
<td>R-SGH</td>
<td>24.66</td>
<td>33.62</td>
<td>44.12</td>
</tr>
</tbody>
</table>

Table 1: mAP (%) results on the CIFAR-10 dataset.

Image Retrieval on CIFAR-10 The CIFAR-10 dataset contains 60,000 color images from 10 object classes. Each original image is of size $32 \times 32$ and is represented as a 512-D GIST feature vector [Oliva and Torralba, 2001]. Following the same setting as in [Erin Liong et al., 2015], we randomly sample 1000 samples, 100 per class, as the query data, and use the remaining 590,000 images as the gallery set.

The parameter $\delta$ in R-SGH is set to 0.1 on this dataset. For all considered methods, we repeat the experiments 20 trials and take the averages as the final results. Table 1 lists the mAP results of different hashing methods. Figure 3 shows the recall and precision curves as functions of the retrieved number of instances under different number of hash bits.

First, the clear advantages of our R-SGH over the other compared unsupervised hashing methods can be easily observed in terms of mAP performance, which demonstrate that our redundancy-resistant generative hashing framework can consistently improve the overall retrieval performance. Second, R-SGH keeps a greater growth of mAP than that of SGH when increasing number of hash bits are used, this is because the redundancy-resistant terms in R-SGH can effectively activate the uninformative bit in the generative model. Third, R-SGH outperforms the other compared methods by a large margin when 128 bits of hash code are used. We attribute this to the fact that higher dimensional latent representations come with higher information redundancy. Finally, we note that SGH and ITQ achieve better recall and precision performances when 32 bits hash code are used. This is due to very little information redundancy contained in the hash code when the number of hash bits is small. In this case, the redundancy-resistant regularization influences the reconstruction procedure and may degrade the performance. Fortunately, if low redundancy is expected in advance, we can prevent the negative impact by setting a small threshold parameter $\epsilon$.

Image Retrieval on Caltech-256 The Caltech-256 dataset consists of 29,780 images associated with 256 object categories. We randomly choose 1000 images as the query set and the rest of the dataset is regarded as the training set. We use the VGG network (VGG-fc7 [Szegedy et al., 2015]) pre-trained on ILSVRC [Russakovsky et al., 2015] as the feature extractor, which transforms each image into a 1024-dimensional vector.

We followed the same settings as in the CIFAR-10 experiment and used the same DNN structures. Table 2 displays the mAP performance of different hashing methods on Caltech-256 dataset. Figure 4 shows the precision-recall curves for different hashing methods under 32, 64 and 128 bits, respectively. We can see that R-SGH significantly outperforms the other compared methods on this dataset, even under 32 bits hash code. It is probably due to the well known fact that the CNN features their self have some information redundan-
Figure 2: Illustration of MNIST images re-generated by R-SGH and SGH with 64 bit hash code. The hash codes of R-SGH contain richer information for reconstruction owing to their low redundancy merit.

![Figure 2: Illustration of MNIST images re-generated by R-SGH and SGH with 64 bit hash code.](image1.png)

Figure 3: Recall and precision curves on CIFAR-10 dataset. The advantage of R-SGH is more evident when more hash bits are used, which verifies our conjecture that higher dimensional latent representations come with higher information redundancy.

![Figure 3: Recall and precision curves on CIFAR-10 dataset.](image2.png)

3.3 Sensitivity Analysis

For sensitivity analysis of the proposed model, we use the Caltech-256 data set to evaluate the influences of regularization parameters $\delta$ and $\eta$. Here, the number of hash bits for each image is fixed to 128. When $\eta$ is fixed to 0.01, Figure 5 (a) shows the mAP performance of R-SGH with varying $\delta$. When $\delta$ is fixed to 0.01, Figure 5 (b) shows the mAP performance of R-SGH with varying $\eta$.

It is worth to mention that R-SGH seems to be more sensitive to $\delta$. Unsuitable $\delta$ can degrade the mAP performance severely, while $\eta$ has little impact on the mAP in a relatively large range, i.e., $[0.002, 0.07]$, as shown in Figure 5 (b). Both terms we proposed improves the discriminative power of the learned hash codes.

![Figure 3: Recall and precision curves on CIFAR-10 dataset.](image3.png)

Table 2: mAP (%) results on the Caltech-256 dataset.

<table>
<thead>
<tr>
<th>Method</th>
<th>32 bits</th>
<th>64 bits</th>
<th>128 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>ITQ</td>
<td>50.12</td>
<td>68.36</td>
<td>76.88</td>
</tr>
<tr>
<td>SH</td>
<td>41.64</td>
<td>52.32</td>
<td>62.93</td>
</tr>
<tr>
<td>PCAH</td>
<td>43.62</td>
<td>55.65</td>
<td>66.02</td>
</tr>
<tr>
<td>SGH</td>
<td>47.12</td>
<td>71.09</td>
<td>78.61</td>
</tr>
<tr>
<td>Deep-SGH</td>
<td>51.39</td>
<td>70.34</td>
<td>79.39</td>
</tr>
<tr>
<td>R-SGH</td>
<td><strong>59.02</strong></td>
<td><strong>74.18</strong></td>
<td><strong>84.96</strong></td>
</tr>
</tbody>
</table>

Table 2: mAP (%) results on the Caltech-256 dataset.
Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence (IJCAI-18)

Figure 4: Precision-recall curves on Caltech-256 dataset under 32, 64 and 128 bits hash code, respectively. The proposed R-SGH significantly outperforms the other compared methods even under 32 bits hash code, which may be due to the well known fact that CNN features their self have information redundancy.

Figure 5: (a) mAP vs. $\delta$ curve with fixed $\eta = 0.01$. (b) mAP vs. $\eta$ curve with fixed $\delta = 0.01$.

4 Related Work

Learning-based hashing is an effective solution to accelerate similarity search by designing compact binary codes in a low-dimensional space. To enhance the expressiveness of fixed-length hash codes, there already exist some hash learning methods that pursue the independence (or low redundancy) between the hash bits. For example, the iterative quantization method [Gong et al., 2013] maximizes the variance of each binary bit to keep the non-relevance and minimizes the binarization loss to obtain a high performance for image retrieval. Later, Liong et.al [Erin Liong et al., 2015] proposed to use a DNN to learn hash codes with three objectives: (1) the loss between the real-valued feature descriptor and the learned binary codes is minimized; (2) binary codes distribute evenly on each bit; and (3) different bits are as independent as possible. They pursue the independence between hash bits by constraining the weight matrix in the encoders to be high rank. A more recent work named DeepBit [Lin et al., 2016] learns compact binary code based on similar rules, i.e., minimal loss quantization, uncorrelated and evenly distributed bits. Bypassing quantization, our redundancy-resistant generative hashing method clearly differs from these methods.

Recent studies of VAEs for representation learning have revealed the decoupling phenomenon of latent code dimensionality from expressive power. Caused by the variational pruning [Hoffman, 2017; Yeung et al., 2017], this phenomenon can be seen as a virtue of automatic relevance determination if efficiency is not a problem (but it is not the case for hashing-based similarity search), but also as a problem when many units collapse early in training before they learned a useful representation. [Sønderby et al., 2016] observed that such units remain uninformative for the rest of the training, presumably trapped in a local minima or saddle point, with the optimization algorithm unable to re-activate them. In [Bowman et al., 2015; Sønderby et al., 2016], the authors used warm-up strategy to alleviate it by initializing training using the reconstruction error only. As more principled approaches, [Hoffman, 2017] proposed to learn a shared shearing matrix to rotate the variational distribution, and [Tomczak and Welling, 2017] proposed a “Variational Mixture of Posteriors” prior for the latent codes. The most similar strategy to ours is to restrict the power of the decoder by word drop-out [Bowman et al., 2015], lossy coding [Chen et al., 2016] and dilated CNN [Yang et al., 2017]. Nevertheless, our adversarial learning framework is more general than these works.

5 Conclusion

We designed a new deep generative architecture which forces the latent hashing codes to not only reconstruct data through the generative network but also retain minimal difference to the last real-valued hidden layer of the network. Furthermore, during posterior inference, we proposed to regularize the standard auto-encoding variational objective with an additional term that explicitly accounts for the negative redundancy degree of latent code dimensions. We designed an adversarial strategy to optimize the generative, the variational, and the redundancy-resisting parameters. Experimental results show that our redundancy-resistant generative hashing framework can significantly boost the quality of learned codes and achieve state-of-the-art retrieval performance on image data.

Acknowledgements

This work was supported by the National Natural Science Foundation of China (No. 61602449, 61672501, 61603372).
References


