SCRATCH: A Scalable Discrete Matrix Factorization Hashing for Cross-Modal Retrieval

Chuan-Xiang Li[1], Zhen-Duo Chen[1], Peng-Fei Zhang[1], Xin Luo[1], Liqiang Nie[2], Wei Zhang[3], Xin-Shun Xu[1*]
{chuanxiang.lee,chenzd.sdu,mima.zpf,luoxin.lxin,nieliqiang}@gmail.com,{davidzhang,xuxinshun}@sdu.edu.cn
[1]School of Software, Shandong University, Jinan 250101, China
[2]School of Computer Science and Technology, Shandong University, Qingdao 266237, China
[3]School of Control Science and Engineering, Shandong University, Jinan 250061, China

ABSTRACT

In recent years, many hashing methods have been proposed for the cross-modal retrieval task. However, there are still some issues that need to be further explored. For example, some of them relax the binary constraints to generate the hash codes, which may generate large quantization error. Although some discrete schemes have been proposed, most of them are time-consuming. In addition, most of the existing supervised hashing methods use an $n \times n$ similarity matrix during the optimization, making them unscalable. To address these issues, in this paper, we present a novel supervised cross-modal hashing method—Scalable disCRete mATrix faCtorization Hashing, SCRATCH for short. It leverages the collective matrix factorization on the kernelized features and the semantic embedding with labels to find a latent semantic space to preserve the intra- and inter-modality similarities. In addition, it incorporates the label matrix instead of the similarity matrix into the loss function. Based on the proposed loss function and the iterative optimization algorithm, it can learn the hash functions and binary codes simultaneously. Moreover, the binary codes can be generated discretely, reducing the quantization error generated by the relaxation scheme. Its time complexity is linear to the size of the dataset, making it scalable to large-scale datasets. Extensive experiments on three benchmark datasets, namely, Wiki, MIRFlickr-25K, and NUS-WIDE, have verified that our proposed SCRATCH model outperforms several state-of-the-art unsupervised and supervised hashing methods for cross-modal retrieval.

CCS CONCEPTS

• **Computing methodologies → Learning paradigms**; • **Information systems → Multimedia and multimodal retrieval**;

KEYWORDS

Cross-Modal Retrieval, Hashing, Matrix Factorization, Discrete Optimization

ACM Reference Format:
Chuan-Xiang Li, Zhen-Duo Chen, Peng-Fei Zhang, Xin Luo, Liqiang Nie, Wei Zhang, Xin-Shun Xu. 2018. SCRATCH: A Scalable Discrete Matrix Factorization Hashing for Cross-Modal Retrieval. In 2018 ACM Multimedia Conference (MM '18), October 22-26, 2018, Seoul, Republic of Korea. ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/3240508.3240547

1 INTRODUCTION

Approximate Nearest Neighbor (ANN) search plays a fundamental role in many fields, spanning from information retrieval and data mining to computer vision [12, 20, 22, 26, 31, 33, 35, 37, 44]. The most representative methods for ANN search are the hashing-based ones [23, 29, 30, 32, 36, 38, 43, 45, 47], having attracted much attention in the past several years. They map the data points into the compact low-dimensional representations in a Hamming space while preserving their similarities in the original feature space. As a result, the search becomes much faster by using XOR operation in the Hamming space. Inspired by this, various hashing methods based on different machine learning theories have been introduced, e.g., Spectral Hashing (SH) [39], Iterative Quantization (ITQ) [10], Supervised Hashing with Kernels (KSH) [19], K-means Hashing [11], and COSDISH [15]. More recently, some deep hashing models have emerged and obtained competitive performance, e.g., CNNH [40], Deep Visual-Semantic Hashing (DVSH) [2], Deep Cross-Modal Hashing (DCMH) [14], Adversarial Cross-Modal Retrieval (ACMR) [34] and Dual Deep Neural Networks Cross-Modal Hashing (DDCMH) [5].

More recently, the cross-modal retrieval [1, 4, 24, 25, 42, 48, 49] is gaining its momentum. Accordingly, many cross-modal hashing methods have been devised. They can be roughly divided into two categories: the unsupervised and supervised methods. Unsupervised ones learn the intra- and inter-modal similarities of the given data without supervised information. Typical examples in this category are Latent Semantic Sparse Hashing (LSSH) [50], Collective Matrix Factorization Hashing (CMFH) [8], Composite Correlation Quantization (CCQ) [21], and Fusion Similarity Hashing (FSH) [18]. To be more specifically, LSSH uses sparse coding and matrix factorization to learn the latent spaces and merges the learned latent features to generate the unified hash codes. CMFH leverages collective matrix factorization with a latent factor model from different views to learn the unified hash codes. CCQ finds the correlation-maximal mappings that transform different modalities into an isomorphic latent space, and learns composite quantizers

[*] Corresonding Author

MM '18, October 22–26, 2018, Seoul, Republic of Korea

ACM ISBN 978-1-4503-5665-7/18/10…$15.00
https://doi.org/10.1145/3240508.3240547

that convert the isomorphic latent features into compact binary codes. FSH first models the fusion similarity via constructing an undirected asymmetric graph among different modalities, and then a graph hashing scheme with an alternating optimization algorithm is introduced to learn the binary codes with embedding the fusion similarity.

By contrast, supervised ones try to model the intra- and inter-modal similarity by leveraging the supervised information, e.g., semantic labels/tags, to further exploit the information among different modalities. For example, Semantic Correlation Maximization (SCM) [46] can utilize the supervised information for training with the linear-time complexity by avoiding explicitly computing the pairwise similarity matrix, making it scalable to the large-scale datasets. Semantics Preserving Hashing (SePH) [16] treats semantic affinities as the supervised information, and then transforms them into a probability distribution, and approximates it with the to-be-learnt hash codes in a Hamming space by minimizing the Kullback-Leibler divergence. Discriminative Cross-modal Hashing (DCH) [41] learns modality-specific hash functions for generating unified binary codes, viewed as representative features of discriminative classification with class labels.

It is worth noting that several CMF-based hashing methods have been proposed, e.g., Collective Matrix Factorization Hashing (CMFH) [8], Supervised Collective Matrix Factorization Hashing (SCMFH) [9] and Supervised Matrix Factorization Hashing (SMFH) [17]. However, CMFH is an unsupervised one, which cannot use the label information. SCMFH and SMFH are actually the supervised extensions of CMFH. However, both SCMFH and SMFH employ the relaxation scheme, resulting in generating a large quantization loss and deteriorating the retrieval performance. Moreover, SMFH uses the Laplacian matrix that is not scalable to large-scale datasets. To tackle this issue, it employs a sampling technique, resulting in the information loss problem.

As aforementioned, many cross-modal hashing methods have been proposed; however, there are still some issues that need to be further considered: 1) Most of them relax the discrete constraints for easy optimization and then quantize the learnt real-valued solution to the binary hash codes, which may generate large quantization error and make the obtained hash codes unreliable. 2) Several methods, e.g., DCH [41], try to keep the discrete constraints and solve the optimization problem bit by bit via a discrete cyclic coordinate descent (DCC) method; however, the bit-wise optimization makes the training much time-consuming. And 3) some methods rely on an $n \times n$ pairwise similarity matrix to measure the similarity among instances, such as SePH and FSH. As to large-scale datasets, this will increase the computational complexity and the memory cost, resulting in these methods unscalable. To tackle this issue, a subset of training instances is usually adopted instead of utilizing all the samples.

To address the aforementioned challenges, in this paper, we present a novel supervised cross-modal hashing method, namely, Scalable disCRete mATrix faCtorization Hashing (SCRATCH). Based on the assumption that different modalities share the same latent space and keep the labels in the latent space, it utilizes Collective Matrix Factorization (CMF) on the kernelized features and semantic embedding with labels to find a latent semantic space whereby the intra- and inter-modality similarities are preserved. Thereafter, it learns the hash functions mapping the representations in the latent space into the binary codes. Moreover, based on a proposed alternating optimization algorithm, SCRATCH generates the binary codes discretely, reducing the quantization error generated in the relaxation schemes adopted by many existing methods. In addition, it does not use the similarity matrix during the optimization and the time complexity of the proposed optimization strategy is linear to the size of dataset, making the training of SCRATCH much efficient and scalable to large-scale datasets.

The contributions of SCRATCH are summarized as follows:

- A novel supervised cross-modal hashing method is proposed to efficiently capture the correlations between the heterogeneous data. By leveraging CMF and semantic embedding, it can make full use of the supervised information.
- An iterative optimization strategy is presented to solve the optimization problem in SCRATCH, and the time complexity is linear to the size of the given dataset, making SCRATCH scalable to large-scale datasets. Moreover, based on the optimization scheme, SCRATCH generates the binary codes discretely, reducing the quantization error.
- Extensive experiments are conducted on three benchmark datasets. The results demonstrate that SCRATCH outperforms several state-of-the-art hashing methods for the cross-modal retrieval.

The rest of this paper is organized as follows. Section 2 introduces our proposed SCRATCH model including its loss function, the optimization algorithm, and some further analysis, e.g., the extension to out-of-samples and the computational complexity. Section 3 presents the experimental results and the analyses, followed by conclusion and future work in Section 4.

2 PROPOSED METHOD

2.1 Notations

Suppose $\boldsymbol{O} = \{\mathbf{o}_1, \mathbf{o}_2, ..., \mathbf{o}_n\}$ is the training set with n instances, where $\mathbf{o}_i = \{\mathbf{x}_i^{(1)}, \mathbf{x}_i^{(2)}, ..., \mathbf{x}_i^{(m)}\}$ is the i-th instance and m is the number of modalities. Without losing generality, we further suppose that the instances are zero-centered in each modality, i.e., $\sum_{i=1}^{n} \mathbf{x}_i^{(j)} = \mathbf{0}, j = 1, ..., m$. Let us denote $\mathbf{L} \in \mathbb{R}^{l \times n}$ as the ground-truth label matrix, where l is the number of classes and $\mathbf{L}_{ki} = 1$ if $\mathbf{x}_i$ belongs to class k and 0 otherwise. $\mathbf{B} \in \{-1, 1\}^{r \times n}$ is the to-be-learnt binary code matrix. $\|\cdot\|_F$ denotes the Frobenius norm of a matrix. $sgn(\cdot)$ is an element-wise sign function defined as follows:

$$sgn(x) = \begin{cases} 1 & x > 0; \\ -1 & x \leq 0. \end{cases} \tag{1}$$

2.2 Scalable Discrete Matrix Factorization Hashing

CMF [7] techniques provide low-rank vectorial representations and remove the redundant information by approximating a data matrix $\mathbf{X} \in \mathbb{R}^{d \times n}$ with the outer product of two rank-r matrices $\mathbf{U} \in \mathbb{R}^{d \times r}$ and $\mathbf{V} \in \mathbb{R}^{r \times n}$, where n is the number of instances, d is the feature dimension and r is the number of latent factors.

Intuitively, different modalities of a multi-modal instance all describe the same instance, justifying that they should have the same semantics. Therefore, we can assume that different modalities also share the same latent space found by CMF. In the light of this, we have the following formulation for each modality:

$$\min_{\mathbf{U}_t,\mathbf{V}} \lambda_t \parallel \mathbf{X}^{(t)} - \mathbf{U}_t\mathbf{V} \parallel_F^2 + \gamma(\parallel \mathbf{U}_t \parallel_F^2 + \parallel \mathbf{V} \parallel_F^2), \tag{2}$$

where $\mathbf{X}^{(t)} \in \mathbb{R}^{d_t \times n}$ is the t-th modality of the training set, $\mathbf{U}_t \in \mathbb{R}^{d_t \times r}$, $\mathbf{V} \in \mathbb{R}^{r \times n}$, d_t is the feature dimension of the t-th modality, r is the number of latent factors, λ_t and γ are the balance parameters, $\Sigma_{t=1}^{m} \lambda_t = 1$, and each column vector $\mathbf{v}_i$ of $\mathbf{V}$ is a latent factor vector in the latent semantic space, which represents an instance via the extracted r latent topics.

In the original space, different modalities of one instance share the same semantic labels; therefore, the representations of one instance in the common latent found by CMF should contain the same labels. More specifically, we suppose that the common latent representations can be regressed to the corresponding class labels. In this way, the explicit semantic information can be embedded into the shared latent space to ensure that the learnt latent semantic information preserves the label consistency. To accomplish this, we define the following formulation:

$$\min_{\mathbf{G},\mathbf{V}} \alpha \parallel \mathbf{L} - \mathbf{G}\mathbf{V} \parallel_F^2 + \gamma \parallel \mathbf{G} \parallel_F^2, \tag{3}$$

where $\mathbf{L}$ is the ground-truth label matrix, $\mathbf{G} \in \mathbb{R}^{l \times r}$ is the projection matrix, l is the number of classes and α is a balance parameter.

To generate the binary codes from the latent semantic representations, we further define the following sub-optimization problem:

$$\begin{aligned} &\min_{\mathbf{B},\mathbf{R}} \parallel \mathbf{B} - \mathbf{R}\mathbf{V} \parallel_F^2, \\ &s.t. \quad \mathbf{B} \in \{-1,1\}^{r \times n}, \mathbf{R}\mathbf{R}^\top = \mathbf{I}, \end{aligned} \tag{4}$$

where $\mathbf{R} \in \mathbb{R}^{r \times r}$ is an orthogonal rotation matrix, $\mathbf{B}$ is the hash codes of all training instances, and r is the length of hash codes. It is worth noting that we introduce the random orthogonal rotation into the above problem. In this way, as shown in the optimization algorithm in Section 2.3, it can keep the optimization problem discrete during the training and generate the discrete binary codes directly. Therefore, it can avoid the large quantization error generated by the relaxation scheme adopted by some pioneer works.

In order to perform the cross-modal retrieval, SCRATCH further learns modality-specific functions to map different modalities of an instance into the latent semantic space. For this purpose, we further suppose that different modalities of an instance can be mapped to the same latent semantic representation. Under this assumption, we can reach the following formulation:

$$\min_{\mathbf{P}_t,\mathbf{V}} \mu \parallel \mathbf{V} - \mathbf{F}_t(\mathbf{X}^{(t)}) \parallel_F^2 + \gamma \parallel \mathbf{P}_t \parallel_F^2, \tag{5}$$

where $\mathbf{F}_t(\mathbf{X}^{(t)}) = \mathbf{P}_t\mathbf{X}^{(t)}$ is the mapping function for the t-th modality and μ is a balance parameter.

To capture the non-linear structure of different modalities, we further adopt the kernel features to replace the original features in the optimization problems, e.g., in Eqn. (2) & (5). Specifically, they can be rewritten as follows:

$$\min_{\mathbf{U}_t,\mathbf{V}} \lambda_t \parallel \boldsymbol{\phi}(\mathbf{X}^{(t)}) - \mathbf{U}_t\mathbf{V} \parallel_F^2 + \gamma(\parallel \mathbf{U}_t \parallel_F^2 + \parallel \mathbf{V} \parallel_F^2), \tag{6}$$

$$\min_{\mathbf{P}_t,\mathbf{V}} \mu \parallel \mathbf{V} - \mathbf{F}_t(\boldsymbol{\phi}(\mathbf{X}^{(t)})) \parallel_F^2 + \gamma \parallel \mathbf{P}_t \parallel_F^2, \tag{7}$$

where $\boldsymbol{\phi}(\mathbf{x})$ is the non-linear embedding of the instance $\mathbf{x}$. In this paper, we use the RBF kernel mapping, i.e., $\boldsymbol{\phi}_i(\mathbf{x}) = exp(\frac{-\|\mathbf{x}-\mathbf{a}_i\|_2^2}{2\sigma^2})$, where $\{\mathbf{a}_j\}_{j=1}^{q}$ are the randomly selected q anchor points from the training samples, and σ is the kernel width calculated via $\sigma = \frac{1}{nq} \sum_{i=1}^{n} \sum_{j=1}^{q} \parallel \mathbf{x}_i - \mathbf{a}_j \parallel_2$.

Combining Eqn. (3), (4), (6) and (7), we have the following overall objective function:

$$\begin{aligned} \min_{\mathbf{P}_t,\mathbf{U}_t,\mathbf{V}} & \sum_{t=1}^{m} \lambda_t \parallel \boldsymbol{\phi}(\mathbf{X}^{(t)}) - \mathbf{U}_t\mathbf{V} \parallel_F^2 \\ & + \mu \sum_{t=1}^{m} \parallel \mathbf{V} - \mathbf{P}_t\boldsymbol{\phi}(\mathbf{X}^{(t)}) \parallel_F^2 + \alpha \parallel \mathbf{L} - \mathbf{G}\mathbf{V} \parallel_F^2 \\ & + \parallel \mathbf{B} - \mathbf{R}\mathbf{V} \parallel_F^2 + \gamma Re(\mathbf{V},\mathbf{G},\mathbf{P}_t,\mathbf{U}_t), \\ & s.t. \quad \sum_{t=1}^{m} \lambda_t = 1, \mathbf{B} \in \{-1,1\}^{r \times n}, \mathbf{R}\mathbf{R}^\top = \mathbf{I}, \end{aligned} \tag{8}$$

where m is the number of modalities, λ_t, μ, α, γ are the tradeoff parameters, and $Re(\mathbf{V},\mathbf{G},\mathbf{P}_t,\mathbf{U}_t)$ is a regularization term to avoid overfitting, defined as the sum of the regularization terms in each sub-optimization problem as mentioned previously.

2.3 Optimization Algorithm

It can be proven that the optimization problem in Eqn. (8) is non-convex with multiple variables, $\mathbf{U}_t$, $\mathbf{P}_t$, $\mathbf{G}$, $\mathbf{V}$, $\mathbf{R}$, and $\mathbf{B}$. Fortunately, it is convex with respect to any one of these matrix variables while fixing the others. Therefore, to tackle the optimization problem, we propose an anternating optimization scheme as shown below.

Step 1: Fixing others and solving $\mathbf{U}_t$.

Fixing other variables, Eqn. (8) can be rewritten as:

$$\min_{\mathbf{U}_t} \lambda_t \parallel \boldsymbol{\phi}(\mathbf{X}^{(t)}) - \mathbf{U}_t\mathbf{V} \parallel_F^2 + \gamma \parallel \mathbf{U}_t \parallel_F^2 . \tag{9}$$

Setting the derivative of Eqn. (8) w.r.t $\mathbf{U}_t$ to zero, we can obtain $\mathbf{U}_t$ with a closed-form solution:

$$\mathbf{U}_t = \lambda_t\boldsymbol{\phi}(\mathbf{X}^{(t)})\mathbf{V}^\top(\lambda_t\mathbf{V}\mathbf{V}^\top + \gamma\mathbf{I})^{-1}. \tag{10}$$

Step 2: Fixing other variables, solving G.

Similar to the optimization of $\mathbf{U}_t$, when other variables are fixed, Eqn. (8) can be reformulated as:

$$\min_{\mathbf{G}} \alpha \parallel \mathbf{L} - \mathbf{G}\mathbf{V} \parallel_F^2 + \gamma \parallel \mathbf{G} \parallel_F^2 . \tag{11}$$

Setting the derivative of Eqn. (8) w.r.t $\mathbf{G}$ to zero, we can derive the analytical solution of $\mathbf{G}$ as follows:

$$\mathbf{G} = \alpha\mathbf{L}\mathbf{V}^\top(\alpha\mathbf{V}\mathbf{V}^\top + \gamma\mathbf{I})^{-1}, \tag{12}$$

where $\mathbf{I} \in \mathbb{R}^{r \times r}$ is the identity matrix.

Step 3: Fixing other variables, solving $\mathbf{P}_t$.

When other variables are fixed, Eqn. (8) becomes:

$$\min_{\mathbf{P}_t} \mu \parallel \mathbf{V} - \mathbf{P}_t\boldsymbol{\phi}(\mathbf{X}^{(t)}) \parallel_F^2 + \gamma \parallel \mathbf{P}_t \parallel_F^2 . \tag{13}$$

The solution can also be obtained by setting the derivative of Eqn. (8) w.r.t. $\mathbf{P}_t$ to zero:

$$\mathbf{P}_t = \mu\mathbf{V}\boldsymbol{\phi}(\mathbf{X}^{(t)})^\top\left(\mu\boldsymbol{\phi}(\mathbf{X}^{(t)})\boldsymbol{\phi}(\mathbf{X}^{(t)})^\top + \gamma\mathbf{I}\right)^{-1}. \tag{14}$$

Step 4: Fixing other variables, solving V.

Algorithm 1 Optimization algorithm in SCRATCH

```
Input: Training data matrices of $t$-th modality $\mathbf{X}^{(t)}$, parameters $\lambda_t, \mu, \alpha, \gamma$, hash code length $r$, and the total iterative number $c$.
Output: Hash code matrix $\mathbf{B}$, mapping matrix $\mathbf{P}_t$, $\mathbf{G}$, factorization matrix $\mathbf{U}_t$, rotation matrix $\mathbf{R}$, and latent representation $\mathbf{V}$.

Procedure:
1. Randomly initialize $\mathbf{B}, \mathbf{R}, \mathbf{V}, \mathbf{U}_t, \mathbf{G}$, and $\mathbf{P}_t$;
2. Embed $\mathbf{X}^{(t)}$ into the nonlinear space and obtain the kernel features $\mathbf{F}_t(\mathbf{X}^{(t)})$;
for $i = 1$ to $c$ do
  3. Fix $\mathbf{P}_t$, $\mathbf{G}$, $\mathbf{V}$, $\mathbf{R}$, and $\mathbf{B}$, update $\mathbf{U}_t$ using Eqn. (10);
  4. Fix $\mathbf{U}_t$, $\mathbf{P}_t$, $\mathbf{V}$, $\mathbf{R}$, and $\mathbf{B}$, update $\mathbf{G}$ using Eqn. (12);
  5. Fix $\mathbf{U}_t$, $\mathbf{G}$, $\mathbf{V}$, $\mathbf{R}$, and $\mathbf{B}$, update $\mathbf{P}_t$ using Eqn. (14);
  6. Fix $\mathbf{U}_t$, $\mathbf{G}$, $\mathbf{P}_t$, $\mathbf{R}$, and $\mathbf{B}$, update $\mathbf{V}$ using Eqn. (15);
  7. Fix $\mathbf{U}_t$, $\mathbf{G}$, $\mathbf{P}_t$, $\mathbf{V}$, and $\mathbf{B}$, update $\mathbf{R}$ via $\mathbf{R} = \mathbf{S}\widetilde{S}^\top$;
  8. Fix $\mathbf{U}_t$, $\mathbf{G}$, $\mathbf{P}_t$, $\mathbf{V}$, and $\mathbf{R}$, update $\mathbf{B}$ as $sgn(\mathbf{RV})$;
end for
return: $\mathbf{U}_t$, $\mathbf{P}_t$, $\mathbf{G}$, $\mathbf{V}$, $\mathbf{R}$, and $\mathbf{B}$.
```

By fixing other variable and setting the derivative of Eqn. (8) w.r.t $\mathbf{V}$ to zero, we have:

$$\begin{aligned}\mathbf{V} = &\Big(\sum_{t=1}^{m} \lambda_t \mathbf{U}_t^\top \mathbf{U}_t + \alpha \mathbf{G}^\top \mathbf{G} + \mathbf{R}^\top \mathbf{R} + (m\mu + \gamma)\mathbf{I}\Big)^{-1} \bullet \\ &\Big(\sum_{t=1}^{m} \big(\lambda_t \mathbf{U}_t^\top \boldsymbol{\phi}(\mathbf{X}^{(t)}) + \mu \mathbf{P}_t \boldsymbol{\phi}(\mathbf{X}^{(t)})\big) + \alpha \mathbf{G}^\top \mathbf{L} + \mathbf{R}^\top \mathbf{B}\Big).\end{aligned} \tag{15}$$

Step 5: Fixing other variables, solving R.

Fixing other variables, Eqn. (8) can be reformulated as follows:

$$\min_{\mathbf{R}} \parallel \mathbf{B} - \mathbf{RV} \parallel_F^2, \quad s.t. \;\; \mathbf{RR}^\top = \mathbf{I}. \tag{16}$$

Apparently, it is a classical Orthogonal Procrustes problem [28], which can be solved by leveraging Singular Value Decomposition (SVD). After the SVD operation, we can obtain $\mathbf{BV}^\top = \mathbf{S}\Omega\widetilde{S}^\top$, and then derive the solution of $\mathbf{R}$ as $\mathbf{R} = \mathbf{S}\widetilde{S}^\top$.

Step 6: Fixing other variables, solving B.

When other variables are fixed, Eqn. (8) is rewritten as follows:

$$\begin{aligned}&\min_{\mathbf{B}} \parallel \mathbf{B} - \mathbf{RV} \parallel_F^2, \\ &s.t. \;\; \mathbf{B} \in \{-1, 1\}^{r \times n}.\end{aligned} \tag{17}$$

The solution to the above sub-problem can be easily obtained by $\mathbf{B} = sgn(\mathbf{RV})$.

To obtain the final solution, we alternately update $\mathbf{U}_t$, $\mathbf{G}$, $\mathbf{P}_t$, $\mathbf{V}$, $\mathbf{R}$, and $\mathbf{B}$ according to the above steps until it converges. To demonstrate the above optimization scheme clearly, we summarize it in **Algorithm 1**.

2.4 Out-of-Sample Extension

For a new instance that is not in the training set, its binary code can be easily generated. For example, given a query instance with one of its modalities $\mathbf{x}^{(t)}$, its corresponding hash code can be obtained by:

$$\mathbf{b}^{(t)} = sgn\big(\mathbf{RF}_t(\boldsymbol{\phi}(\mathbf{x}^{(t)}))\big) = sgn\big(\mathbf{RP}_t \boldsymbol{\phi}(\mathbf{x}^{(t)})\big), \tag{18}$$

where $\boldsymbol{\phi}(\mathbf{x}^{(t)})$ is the non-linear embedding of $\mathbf{x}^{(t)}$.

2.5 Computational Complexity Analysis

In this section, we show that the time complexity of the optimization algorithm is linear to the size of the training set, i.e., n, making SCRATCH scalable to large-scale datasets. Specifically, the overall time complexity includes $O\big(c((qr + r^2)n + r^3 + qr^2)\big)$ for solving Eqn. (10), $O\big(c((lr + r^2)n + r^3 + lr^2)\big)$ for Eqn. (12), $O\big(c((qr + q^2)n + q^3 + rq^2)\big)$ for Eqn. (14), $O\big(c(mqr^2 + lr^2 + 2r^3 + r(l + r + mq)n)\big)$ for Eqn. (15), $O\big(c(r^2n + r^3)\big)$ for solving R, and $O\big(c(r^2 + r)n\big)$ for solving B, respectively, where q is the number of anchor points, r is the length of hash codes, l is the number of classes, m of modalities, and c of iterations. Since $r, l, q, m, c << n$, the overall computational complexity of the training stage is $O(n(r^2 + lr + q^2 + mqr)c)$, which is linear to the size of the training set, i.e., n.

3 EXPERIMENTS

To validate the effectiveness of the proposed cross-modal hashing method, we conducted extensive experiments on three widely-used benchmark datasets, i.e., Wiki [27], MIRFlickr-25K [13], and NUS-WIDE [6]. We also compared it with eight state-of-the-art shallow cross-modal hashing methods and one deep hashing method.

3.1 Datasets

Wiki: It contains 2,866 image-text pairs collected from the Wikipedia. Each instance is annotated with one of ten semantic classes. In addition, the visual modality of each instance is represented by a 128-dimensional bag-of-visual SIFT feature vector, and the textual one is represented by a 10-dimensional topic vector. On this dataset, we randomly selected 75% image-text pairs of the dataset as the training set, and the remaining 25% as the query set.

MIRFlickr-25K: It consists of 25,000 images collected from Flickr. Each image is annotated by some textual tags selected from 24 unique labels. We selected those points with at least 20 textual tags for our experiment and ultimately obtained 20,015 instances. Each image is described by a 512-dimensional GIST feature vector and the text for each point is represented as a 1,386-dimensional bag-of-words vector. We randomly selected 2,000 instances as the query set, and the remaining 18,015 instances as the training and retrieval set.

NUS-WIDE: It contains 269,648 images crawled from Flickr. Each instance is manually annotated with at least one of 81 provided labels. Considering some labels are scarce, we selected the top 10 most common concepts and the corresponding 186,577 images as the final dataset. Each image-text pair is annotated by at least 1 of 10 concepts. For each instance, the visual view is represented as a 500-dimensional bag-of-visual SIFT feature vector and the textual view is described by a 1,000-dimensional vector. We randomly selected 2,000 instances as the query set and the remaining 184,577 as the training and retrieval set.

3.2 Baselines and Metrics

We compared SCRATCH with eight state-of-the-art shallow cross-modal hashing methods, namely, LSSH [50], CMFH [8], SCM-seq [46], CCQ [21], SePH-km [8], SMFH [17], FSH [18] and DCH [41]. They can be divided into two categories: LSSH, CMFH, CCQ and

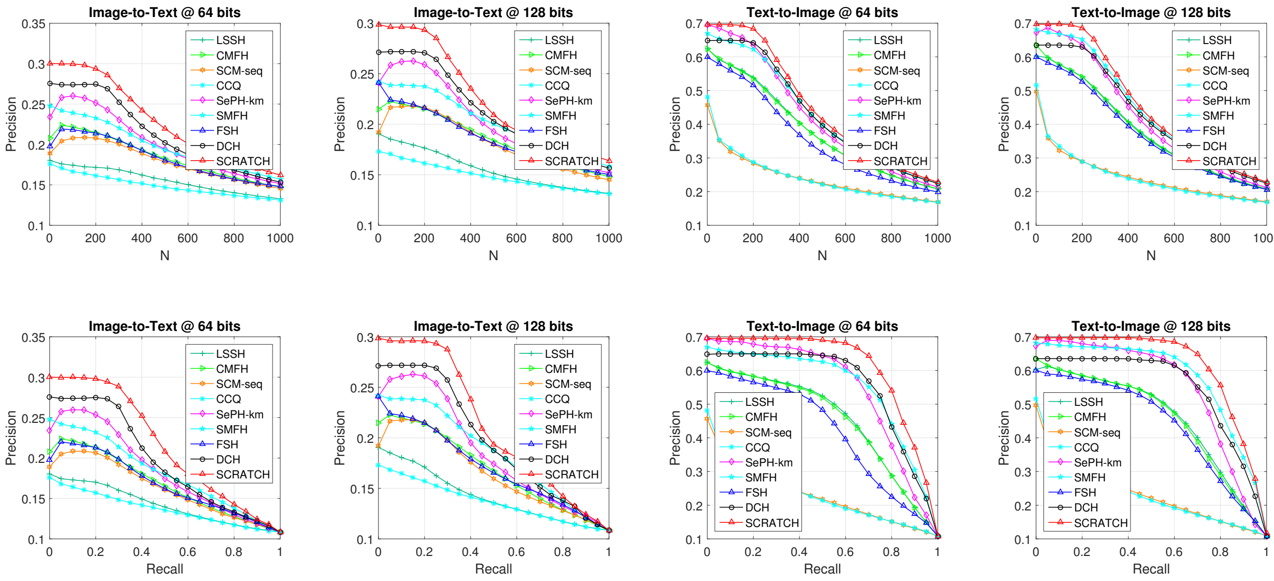

Figure 1: Top-N precision and Precision-Recall curves on Wiki with different code lengths.

FSH are unsupervised ones, while SCM-seq, SePH-km, SMFH and DCH are supervised ones. The source codes of all the baselines are kindly provided by the authors. SCMFH [9] is an supervised extension of CMFH; however, as reported in [9], its performance improvement over CMFH is not remarkable. In addition, its codes are not available. Therefore, it is not included in the baselines. We carefully tuned the parameters of these models and reported their best results. To tackle the high computational cost problem in SePH and FSH, following the strategy adopted in the literature [8] and [16], we randomly selected 5,000 instances to form the training set on NUS-WIDE. The parameters of SCRATCH were selected by a validation procedure, i.e., $\lambda_1 = \lambda_2 = 0.5, \mu = 1,000, \alpha = 500, \gamma = 5$. In addition, the iteration number c was set to 15.

The performance of all methods is evaluated by the widely-used Mean Average Precision (MAP). For a query $\mathbf{q}$, the average precision (AP) is defined as:

$$AP(q) = \frac{1}{L_q} \sum_{r=1}^{n} P_q(r)\delta_q(r), \tag{19}$$

where in the database, L_q is the number of ground-truth neighbors of query $\mathbf{q}$, n is the number of entities, $P_q(r)$ denotes the precision of the top r retrieved entities, and $\delta_q(r) = 1$ if the r-th retrieved entity is a ground-truth neighbour and $\delta_q(r) = 0$, otherwise. On Wiki, the ground-truth neighbors are defined as those having the same label, while on MIRFlickr-25K and NUS-WIDE, we define the ground-truth neighbors as those sharing at least one semantic label. The MAP is defined as:

$$MAP = \frac{1}{|Q|} \sum_{i=1}^{|Q|} AP(q_i), \tag{20}$$

where $|Q|$ is the size of the query set Q.

Table 1: Performance comparison on Wiki measured by MAP.

Task	Method	8 bits	16 bits	32 bits	64 bits	128 bits
Image to Text	LSSH	0.1831	0.2162	0.2164	0.2041	0.2084
	CMFH	0.2006	0.2145	0.2288	0.2360	0.2396
	SCM-seq	0.2125	0.2341	0.2410	0.2437	0.2541
	CCQ	0.1642	0.1675	0.1683	0.1682	0.1680
	SePH-km	0.2620	0.2796	0.2820	0.3076	0.3137
	SMFH	0.1673	0.2276	0.2470	0.2955	0.3133
	FSH	0.1992	0.2270	0.2433	0.2366	0.2463
	DCH	0.3111	0.3491	0.3589	0.3777	0.3791
	SCRATCH	**0.3185**	**0.3696**	**0.3874**	**0.4051**	**0.4001**
Text to Image	LSSH	0.4268	0.4990	0.5225	0.5287	0.5330
	CMFH	0.4434	0.4915	0.5252	0.5276	0.5347
	SCM-seq	0.2013	0.2257	0.2459	0.2480	0.2530
	CCQ	0.2094	0.2410	0.2518	0.2507	0.2543
	SePH-km	0.6065	0.6379	0.6451	0.6662	0.6706
	SMFH	0.3598	0.5242	0.5961	0.6608	0.6924
	FSH	0.4092	0.4864	0.5197	0.4961	0.5247
	DCH	0.6724	0.6815	0.7097	0.7216	0.7141
	SCRATCH	**0.7058**	**0.7471**	**0.7543**	**0.7654**	**0.7679**

To gain deep insights into SCRATCH and all baselines, we further plotted the top-N precision and precision-recall curves of the cases with 64 & 128 bits. In the experiments, we conducted two cross-modal retrieval tasks: 1) Image-to-Text using images as the query to search texts; 2) Text-to-Image using texts as the query to search images. For all these metrics, the larger the values are, the better the results are.

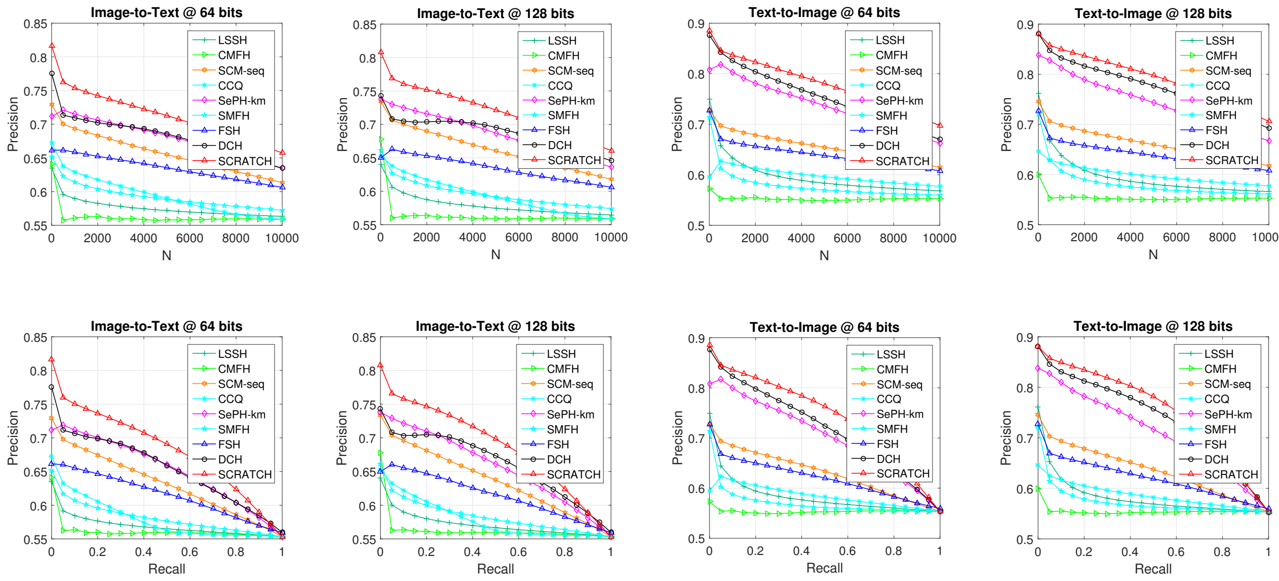

Figure 2: Top-N precision and Precision-Recall curves on MIRFlickr-25K with different code lengths.

3.3 Results and Analysis

3.3.1 Results on Wiki. The MAP values of SCRATCH and all baselines on Wiki are reported in Table 1, and the top-N precision and precision-recall curves of the cases with 64 & 128 bits on Wiki are plotted in Figure 1. From these results, we can observe the following points:

- SCRATCH significantly outperforms all of the baselines on both tasks with various code lengths.
- On Text-to-Image tasks, the MAP values of SCRATCH are much better than those of the baselines. The main reason may be that the matrix factorization can precisely find better latent topic concepts from the text than from the image.
- Most supervised methods, e.g., SCRATCH, DCH, and SePH-km, are superior to the unsupervised ones, e.g., CMFH and FSH, demonstrating the importance of utilizing the semantic information.
- SCRATCH performs much better than the baselines in cases where N is small, e.g., the Image-to-Text tasks. This demonstrates that SCRATCH can return more related samples when N is small, which is very important in a search task.

3.3.2 Results on MIRFlickr-25K. The MAP values of the Image-to-Text and Text-to-Image tasks on MIRFlickr-25K are summarized in Table 2. The top-N precision and precision-recall curves are plotted in Figure 2. From Table 2 and Figure 2, we have the following observations:

- SCRATCH outperforms all of the baselines on both tasks with various code lengths.
- The performance of SCRATCH boosts as the code length increases, verifying that longer hash codes can encode more information, and thus its performance can be promoted.

Table 2: Performance comparison on MIRFlickr-25K measured by MAP.

Task	Method	8 bits	16 bits	32 bits	64 bits	128 bits
Image to Text	LSSH	0.5698	0.5812	0.5811	0.5805	0.5800
	CMFH	0.5599	0.5687	0.5680	0.5685	0.5687
	SCM-seq	0.6235	0.6373	0.6478	0.6537	0.6611
	CCQ	0.5712	0.5885	0.5908	0.5924	0.5928
	SePH-km	0.6641	0.6685	0.6818	0.6830	0.6873
	SMFH	0.5587	0.5688	0.5917	0.5953	0.5961
	FSH	0.5911	0.6016	0.6149	0.6194	0.6242
	DCH	0.6659	0.6738	0.6859	0.6897	0.7030
	SCRATCH	**0.7092**	**0.7131**	**0.7222**	**0.7265**	**0.7346**
Text to Image	LSSH	0.5914	0.5917	0.5929	0.5926	0.5918
	CMFH	0.5615	0.5615	0.5606	0.5606	0.5608
	SCM-seq	0.6103	0.6206	0.6298	0.6372	0.6427
	CCQ	0.5902	0.5970	0.5992	0.6001	0.6001
	SePH-km	0.7033	0.7076	0.7212	0.7293	0.7348
	SMFH	0.5568	0.5586	0.5727	0.5841	0.5828
	FSH	0.5869	0.5979	0.6114	0.6186	0.6251
	DCH	0.7256	0.7511	0.7585	0.7681	0.7909
	SCRATCH	**0.7591**	**0.7762**	**0.7822**	**0.7978**	**0.8063**

- Generally speaking, the performance of most methods on the Text-to-Image tasks exceeds that on the Image-to-Text task. The possible reason is that the text can better describe the topic of the image-text pair than the image does.
- Compared with the results on Wiki, all methods obtain better results on MIRFlickr-25K. One possible reason is that different features were used in the two datasets.

3.3.3 Results on NUS-WIDE. The MAP values of all methods on NUS-WIDE are summarized in Table 3. The top-N precision and

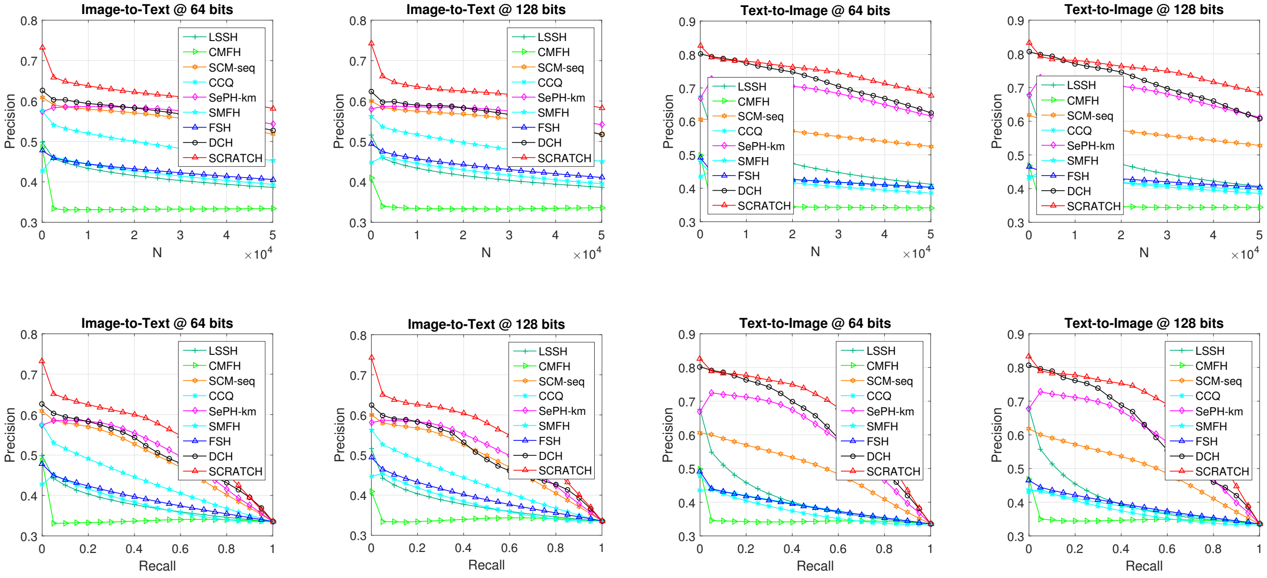

Figure 3: Top-N precision and Precision-Recall curves on NUS-WIDE with different code lengths.

precision-recall curves of the cases with 64 and 128 bits are plotted in Figure 3. From these results, we have the following observations, quite similar to those on Wiki and MIRFlickr-25K:

- SCRATCH outperforms the baselines in every case.
- The performance of SCRATCH keeps improving with the code length increasing.
- Most methods perform better on the Text-to-Image tasks than themselves on the Image-to-Text tasks, which is consistent with that on Wiki and MIRFlickr-25K.
- The supervised methods, e.g., SCRATCH, DCH and SePH-km, outperform the unsupervised ones, further verifying the effectiveness of the use of the semantic information.

In summary, SCRATCH achieves promising performance on the three benchmark datasets, demonstrating that embedding both the latent semantic information and the explicit semantic information of labels can generate more effective hash codes. Moreover, compared with other CMF-based hashing methods, e.g., CMFH and SMFH, SCRATCH yields significant improvements on performance on the three benchmark datasets, justifying the effectiveness of its loss function and optimization scheme.

3.3.4 Parameter Analysis. We conducted the empirical analysis of the parameter sensitivity on all of the datasets. The experiments were carried out by varying the value of one parameter while fixing the others. The parameters λ_1 and λ_2 control the weights of the matrix factorization between two modalities. In our experiments, we observed that they barely affect the performance; therefore, we empirically set $\lambda_1 = \lambda_2 = 0.5$. In addition, we also conducted experiments to demonstrate that SCRATCH can converge within several iterations. When varying the values of μ, α, γ, the MAP curves and the curves of the objective function in the case of 64 bits are reported in Figure 4, where "I-T" and "T-I" represent the Image-to-Text and Text-to-Image tasks, respectively. From these results, we can observe that:

Table 3: Performance comparison on NUS-WIDE measured by MAP.

Task	Method	8 bits	16 bits	32 bits	64 bits	128 bits
Image to Text	LSSH	0.3829	0.3885	0.3869	0.3911	0.3877
	CMFH	0.3406	0.3437	0.3399	0.3409	0.3440
	SCM-seq	0.5013	0.5120	0.5422	0.5488	0.5483
	CCQ	0.3902	0.3959	0.3929	0.4010	0.3952
	SePH-km	0.5256	0.5537	0.5627	0.5622	0.5698
	SMFH	0.3711	0.4006	0.4461	0.4593	0.4594
	FSH	0.3620	0.3732	0.3894	0.4014	0.4084
	DCH	0.5840	0.5808	0.5907	0.5932	0.5843
	SCRATCH	**0.6038**	**0.6207**	**0.6338**	**0.6459**	**0.6496**
Text to Image	LSSH	0.4075	0.4202	0.3444	0.4231	0.4175
	CMFH	0.3456	0.3498	0.3435	0.3486	0.3529
	SCM-seq	0.4709	0.4836	0.5067	0.5141	0.5161
	CCQ	0.3716	0.3740	0.3712	0.3734	0.3765
	SePH-km	0.6102	0.6407	0.6515	0.6608	0.6651
	SMFH	0.3631	0.3789	0.4046	0.4048	0.3997
	FSH	0.3623	0.3717	0.3835	0.3973	0.4007
	DCH	0.7106	0.7103	0.7098	0.7260	0.7223
	SCRATCH	**0.7210**	**0.7392**	**0.7549**	**0.7680**	**0.7755**

- μ and α indeed influence the performance of SCRATCH. SCRATCH obtains the best results when $\mu \approx 1,000$ and $\alpha \approx 500$.
- When μ and α are equal to zero, the performance of SCRATCH degrades significantly, indicating that it benefits much from the non-linear projection and the semantic embedding.
- γ controls the weight of the regularization term, and SCRATCH obtains the best results when $\gamma \approx 5$. It is worth noting that the

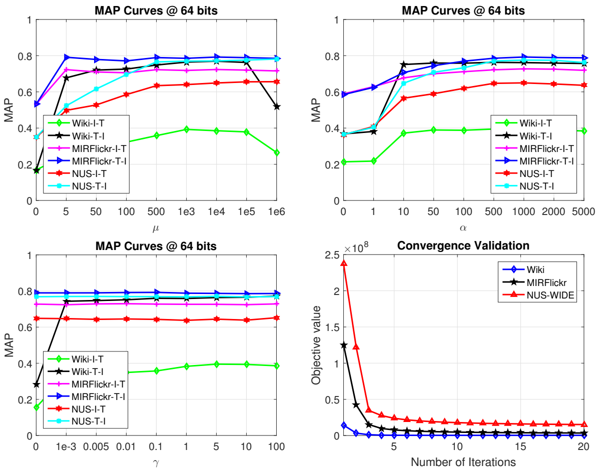

Figure 4: Parameter analysis (μ, α, γ and iterations).

performance of SCRATCH degrades significantly when γ is equal to zero on Wiki; however, there is no such a phenomenon on MIRFlickr-25K and NUS-WIDE. The possible reason is that Wiki is a small dataset, and SCRATCH is very easy to be overfitting on it. Therefore, the regularization term plays a critical role. However, the size of MIRFlickr-25K and NUS-WIDE is large enough to avoid the overfitting problem; correspondingly, the regularization is inessential to SCRATCH.

- SCRATCH converges within a few iterations on all of the datasets, further demonstrating the efficiency of the optimization scheme.

3.3.5 Time Cost Analysis. To verify the efficiency of our proposed SCRATCH model, we further compared the training time (in seconds) of all methods on MIRFlickr-25K. We varied the length of hash codes from 8 to 128. The results are summarized in Table 4. We can observe that the training time of SCRATCH does not significantly increase when the code length becomes longer. As CMFH is an unsupervised method, it is faster than SCRATCH. However, as shown in the previous sub-sections, its retrieval performance is much worse than that of SCRATCH. Therefore, considering both the retrieval performance and the training time, SCRATCH performs well on the benchmark datasets, exhibiting that SCRATCH is efficient and scalable to large-scale datasets.

3.3.6 Comparison with Deep Hashing. We further conducted experiments on MIRFlickr-25K to compare SCRATCH with a state-of-the-art deep cross-modal hashing method, i.e., DCMH [14]. Following [14], we first extracted the deep features of the image modality via the CNN-F network [3] using the same parameters in DCMH. Thereafter, SCRATCH works on the deep features of the image modality and the original text features. The MAP results are summarized in Table 5. We can observe that SCRATCH outperforms DCMH on both the Image-to-Text and Text-to-Image tasks; especially, the MAP values of SCRATCH on the Image-to-Text task are much better than those of DCMH. From these observations, we can easily draw the conclusion that SCRATCH is not an end-to-end deep model, yet it can outperform the state-of-the-art deep hashing method, i.e., DCMH. It further confirms the effectiveness of the loss function and the optimization scheme in SCRATCH.

Table 4: Training time comparison on MIRFlickr-25K (in seconds).

Method	8 bits	16 bits	32 bits	64 bits	128 bits
LSSH	34.41	29.89	29.67	32.44	38.17
CMFH	1.48	1.58	1.57	1.64	1.90
SCM-seq	8.84	12.65	22.90	42.85	77.33
CCQ	6.33	9.12	17.26	29.33	77.41
SePH-km	3328.20	3411.26	3576.82	3805.26	4451.09
SMFH	7.32	6.89	15.40	14.60	16.84
FSH	109.59	108.25	107.52	109.73	115.55
DCH	3.76	4.36	5.49	11.56	37.36
SCRATCH	1.59	1.59	1.88	2.44	3.68

Table 5: MAP comparison of SCRATCH and DCMH on MIRFlickr-25K.

Task	Method	8 bits	16 bits	32 bits	64 bits	128 bits
$I \rightarrow T$	DCMH	0.7276	0.7247	0.7435	0.7484	0.7571
	SCRATCH	**0.7587**	**0.7803**	**0.7978**	**0.8093**	**0.8230**
$T \rightarrow I$	DCMH	0.7453	0.7580	0.7692	0.7758	0.7819
	SCRATCH	**0.7532**	**0.7797**	**0.7913**	**0.7979**	**0.8160**

4 CONCLUSION AND FUTURE WORK

In this paper, we present a novel supervised cross-modal hashing method, i.e., Scalable disCRete mATrix faCtorization Hashing (SCRATCH). It learns a latent semantic space via collective matrix factorization and semantic embedding to preserve the intra- and inter-modality similarities. Based on the proposed loss function and the optimization scheme, it generates the binary codes discretely and avoids the large quantization error problem in the relaxation scheme. Moreover, it leverages the label matrix instead of the similarity matrix, much reducing its time cost. The time complexity of the proposed optimization strategy is linear to the size of the given dataset, ensuring its scalability to large-scale datasets. Extensive experimental results on three benchmark datasets demonstrate that SCRATCH outperforms several state-of-the-art hashing methods for cross-modal retrieval. Especially, when given deep features, it can even outperform the corresponding deep hashing model.

It is worth noting that SCRATCH is a shallow model since we intentionally shed light on the design of the loss function and the optimization scheme. As shown in subsection 3.3.6, given the deep features, SCRATCH can also outperform the deep hashing model. Inspired by this, we plan to incorporate the loss function into a deep network and construct an end-to-end deep hashing model.

5 ACKNOWLEDGEMENTS

This work was partially supported by the National Natural Science Foundation of China (61872428, 61573212, 61772310), the Project of Thousand Youth Talents 2016, and the Key Research and Development Program of Shandong Province (2016GGX101044).

REFERENCES

[1] Michael M Bronstein, Alexander M Bronstein, Fabrice Michel, and Nikos Paragios. 2010. Data fusion through cross-modality metric learning using similarity-sensitive hashing. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*. 3594–3601.

[2] Yue Cao, Mingsheng Long, Jianmin Wang, Qiang Yang, and Philip S Yu. 2016. Deep visual-semantic hashing for cross-modal retrieval. In *Proceedings of ACM International Conference on Knowledge Discovery and Data Mining*. 1445–1454.

[3] Ken Chatfield, Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. 2014. Return of the devil in the details: Delving deep into convolutional nets. In *Proceedings of British Machine Vision Conference*. 1–11.

[4] Fuhai Chen, Rongrong Ji, Jinsong Su, Donglin Cao, and Yue Gao. 2017. Predicting microblog sentiments via weakly supervised multi-modal deep learning. *IEEE Transactions on Multimedia* 20 (2017), 997–1007.

[5] Zhen-Duo Chen, Wan-Jin Yu, Chuan-Xiang Li, Liqiang Nie, and Xin-Shun Xu. 2018. Dual Deep Neural Networks Cross-Modal Hashing. In *Proceedings of AAAI Conference on Artificial Intelligence*. 274–281.

[6] Tat-Seng Chua, Jinhui Tang, Richang Hong, Haojie Li, Zhiping Luo, and Yantao Zheng. 2009. NUS-WIDE: A real-world web image database from national university of singapore. In *Proceedings of ACM International Conference on Image and Video Retrieval*. 48.

[7] Scott Deerwester, Susan T. Dumais, George W. Furnas, Thomas K. Landauer, and Richard Harshman. 1990. Indexing by latent semantic analysis. *Journal of the American Society for Information Science* 41, 6 (1990), 391–407.

[8] Guiguang Ding, Yuchen Guo, and Jile Zhou. 2014. Collective matrix factorization hashing for multimodal data. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*. 2075–2082.

[9] Guiguang Ding, Yuchen Guo, Jile Zhou, and Yue Gao. 2016. Large-scale cross-modality search via collective matrix factorization hashing. *IEEE Transactions on Image Processing* 25, 11 (2016), 5427–5440.

[10] Yunchao Gong and Svetlana Lazebnik. 2011. Iterative quantization: A procrustean approach to learning binary codes. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*. 817–824.

[11] Kaiming He, Fang Wen, and Jian Sun. 2013. K-means hashing: An affinity-preserving quantization method for learning binary compact codes. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*. 2938–2945.

[12] Geoffrey E Hinton and Sam T Roweis. 2003. Stochastic neighbor embedding. In *Proceedings of Advances in Neural Information Processing Systems*. 857–864.

[13] Mark J Huiskes and Michael S Lew. 2008. The MIR Flickr retrieval evaluation. In *Proceedings of ACM International Conference on Multimedia Information Retrieval*. 39–43.

[14] Qing-Yuan Jiang and Wu-Jun Li. 2017. Deep cross-modal hashing. In *Proceedings of IEEE Conference on Computer Vision and Pattern Recognition*. 3270–3278.

[15] Wang-Cheng Kang, Wu-Jun Li, and Zhi-Hua Zhou. 2016. Column sampling based discrete supervised hashing. In *Proceedings of AAAI Conference on Artificial Intelligence*. 1230–1236.

[16] Zijia Lin, Guiguang Ding, Mingqing Hu, and Jianmin Wang. 2015. Semantics-preserving hashing for cross-view retrieval. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*. 3864–3872.

[17] Hong Liu, Rongrong Ji, Yongjian Wu, and Gang Hua. 2016. Supervised matrix factorization for cross-modality hashing. In *Proceedings of International Joint Conference on Artificial Intelligence*. 1767–1773.

[18] Hong Liu, Rongrong Ji, Yongjian Wu, Feiyue Huang, and Baochang Zhang. 2017. Cross-modality binary code learning via fusion similarity hashing. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*. 7380–7388.

[19] Wei Liu, Jun Wang, Rongrong Ji, Yu-Gang Jiang, and Shih-Fu Chang. 2012. Supervised hashing with kernels. In *Proceedings of IEEE International Conference on Computer Vision and Pattern Recognition*. 2074–2081.

[20] Xianglong Liu, Cheng Deng, Bo Lang, Dacheng Tao, and Xuelong Li. 2016. Query-adaptive reciprocal hash tables for nearest neighbor search. *IEEE Transactions on Image Processing* 25, 2 (2016), 907–919.

[21] Mingsheng Long, Yue Cao, Jianmin Wang, and Philip S. Yu. 2016. Composite Correlation Quantization for Efficient Multimodal Retrieval. In *Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval*. 579–588.

[22] Xin Luo, Liqiang Nie, Xiangnan He, Ye Wu, Zhen-Duo Chen, and Xin-Shun Xu. 2018. Fast scalable supervised hashing. In *Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval*. 735–744.

[23] Xin Luo, Ye Wu, and Xin-Shun Xu. 2018. Scalable supervised discrete hashing for large-scale search. In *Proceedings of the World Wide Web Conference on World Wide Web*. 1603–1612.

[24] Xin Luo, Xiao-Ya Yin, Liqiang Nie, Xuemeng Song, Yongxin Wang, and Xin-Shun Xu. 2018. SDMCH: Supervised discrete manifold-embedded cross-modal hashing. In *Proceedings of International Joint Conference on Artificial Intelligence*. 2518–2524.

[25] Jonathan Masci, Michael M Bronstein, Alexander M Bronstein, and Jürgen Schmidhuber. 2014. Multimodal similarity-preserving hashing. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 36, 4 (2014), 824–830.

[26] Liqiang Nie, Shuicheng Yan, Meng Wang, Richang Hong, and Tat-Seng Chua. 2012. Harvesting visual concepts for image search with complex queries. In *Proceedings of ACM International Conference on Multimedia*. 59–68.

[27] Nikhil Rasiwasia, Jose Costa Pereira, Emanuele Coviello, Gabriel Doyle, Gert RG Lanckriet, Roger Levy, and Nuno Vasconcelos. 2010. A new approach to cross-modal multimedia retrieval. In *Proceedings of ACM International Conference on Multimedia*. 251–260.

[28] Peter H Schonemann. 1966. A generalized solution of the Orthogonal Procrustes problem. *Psychometrika* 31, 1 (1966), 1–10.

[29] Fumin Shen, Yang Yang, Li Liu, Wei Liu, Dacheng Tao, and Heng Tao Shen. 2017. Asymmetric binary coding for image search. *IEEE Transactions on Multimedia* 19, 9 (2017), 2022–2032.

[30] Jingkuan Song, Lianli Gao, Li Liu, Xiaofeng Zhu, and Nicu Sebe. 2018. Quantization-based hashing: a general framework for scalable image and video retrieval. *Pattern Recognition* 75 (2018), 175–187.

[31] Jingkuan Song, Yi Yang, Zi Huang, Heng Tao Shen, and Jiebo Luo. 2013. Effective Multiple Feature Hashing for Large-Scale Near-Duplicate Video Retrieval. *IEEE Transactions on Multimedia* 15, 8 (2013), 1997–2008.

[32] Jingkuan Song, Yi Yang, Xuelong Li, Zi Huang, and Yang Yang. 2014. Robust hashing with local models for approximate similarity search. *IEEE Transactions on Cybernetics* 44, 7 (2014), 1225–1236.

[33] Jinhui Tang, Zechao Li, Meng Wang, and Ruizhen Zhao. 2015. Neighborhood discriminant hashing for large-scale image retrieval. *IEEE Transactions on Image Processing* 24, 9 (2015), 2827–2840.

[34] Bokun Wang, Yang Yang, Xing Xu, Alan Hanjalic, and Heng Tao Shen. 2017. Adversarial cross-modal retrieval. In *Proceedings of AAAI Conference on Artificial Intelligence*. 154–162.

[35] Jian Wang, Xin-Shun Xu, Shanqing Guo, Lizhen Cui, and Xiao-Lin Wang. 2016. Linear unsupervised hashing for ANN search in Euclidean space. *Neurocomputing* 171 (2016), 283–292.

[36] Jingdong Wang, Ting Zhang, Nicu Sebe, Heng Tao Shen, et al. 2018. A survey on learning to hash. *IEEE Transactions on Pattern Analysis and Machine Intelligence* 40, 4 (2018), 769–790.

[37] Meng Wang, Weisheng Li, Dong Liu, Bingbing Ni, Jialie Shen, and Shuicheng Yan. 2015. Facilitating image search with a scalable and compact semantic mapping. *IEEE Transactions on Cybernetics* 45, 8 (2015), 1561–1574.

[38] Min Wang, Wengang Zhou, Qi Tian, Zhengjun Zha, and Houqiang Li. 2016. Linear distance preserving pseudo-supervised and unsupervised hashing. In *Proceedings of ACM International Conference on Multimedia*. 1257–1266.

[39] Yair Weiss, Antonio Torralba, and Rob Fergus. 2009. Spectral hashing. In *Proceedings of Advances in Neural Information Processing Systems*. 1753–1760.

[40] Rongkai Xia, Yan Pan, Hanjiang Lai, Cong Liu, and Shuicheng Yan. 2014. Supervised hashing for image retrieval via image representation learning. In *Proceedings of AAAI Conference on Artificial Intelligence*. 2156–2162.

[41] Xing Xu, Fumin Shen, Yang Yang, Heng Tao Shen, and Xuelong Li. 2017. Learning discriminative binary codes for large-scale cross-modal retrieval. *IEEE Transactions on Image Processing* 26, 5 (2017), 2494–2507.

[42] Xin-Shun Xu. 2016. Dictionary Learning Based Hashing for Cross-Modal Retrieval. In *Proceedings of ACM International Conference on Multimedia*. 177–181.

[43] Ting-Kun Yan, Xin-Shun Xu, Shanqing Guo, Zi Huang, and Xiaolin Wang. 2016. Supervised Robust Discrete Multimodal Hashing for Cross-Media Retrieval. In *Proceedings of ACM International Conference on Information and Knowledge Management*. 1271–1280.

[44] Rui Yang, Yuliang Shi, and Xin-Shun Xu. 2017. Discrete Multi-view Hashing for Effective Image Retrieval. In *Proceedings of ACM International Conference on Multimedia Retrieval*. 175–183.

[45] Yang Yang, Yadan Luo, Weilun Chen, Fumin Shen, Jie Shao, and Heng Tao Shen. 2016. Zero-shot hashing via transferring supervised knowledge. In *Proceedings of ACM International Conference on Multimedia*. 1286–1295.

[46] Dongqing Zhang and Wu-Jun Li. 2014. Large-scale supervised multimodal hashing with semantic correlation maximization.. In *Proceedings of AAAI Conference on Artificial Intelligence*. 2177–2183.

[47] Hanwang Zhang, Meng Wang, Richang Hong, and Tat-Seng Chua. 2016. Play and rewind: Optimizing binary representations of videos by self-supervised temporal hashing. In *Proceedings of ACM International Conference on Multimedia*. 781–790.

[48] Peng-Fei Zhang, Chuan-Xiang Li, Meng-Yuan Liu, Liqiang Nie, and Xin-Shun Xu. 2017. Semi-relaxation supervised hashing for cross-modal retrieval. In *Proceedings of ACM International Conference on Multimedia*. 1762–1770.

[49] Yi Zhen and Dit-Yan Yeung. 2012. A probabilistic model for multimodal hash function learning. In *Proceedings of ACM International Conference on Knowledge Discovery and Data Mining*. 940–948.

[50] Jile Zhou, Guiguang Ding, and Yuchen Guo. 2014. Latent semantic sparse hashing for cross-modal similarity search. In *Proceedings of International ACM SIGIR Conference on Research and Development in Information Retrieval*. 415–424.