Collective Bayesian Matrix factorization
Hashing for cross-modal retrieval
Loubna Karbil\textsuperscript{1} and Ahmad Sani\textsuperscript{2} and Imane Daoudi\textsuperscript{3}

\textsuperscript{1}National high school of Electricity and Mechanics, Hassan II University, Casablanca, Morocco
\textsuperscript{2}Department of Mathematics, Ibn Zohr University, Agadir, Morocco
\textsuperscript{3}National high school of Electricity and Mechanics, Hassan II University, Casablanca, Morocco

ABSTRACT

Matrix factorization hashing approaches have been widely applied in large scale cross-modality visual search due to their efficiency to preserve similarities among multimodal features. In this paper, we propose a novel cross-modality hashing technique based on Bayesian matrix factorization that factorizes all modalities into a shared latent semantic space using the Bayesian inference. To achieve better search performance, we measure the similarity using the cosine distance. Several experiments prove that the proposed method achieves better performance than many known methods on cross-modal retrieval applications.

KEYWORDS

Cross-modal retrieval, Matrix Factorization, Bayesian hashing, hash function, Multimodal hashing

I. INTRODUCTION

During recent years, Cross-modal retrieval becomes much attractive mainly with the unprecedented blow up of multimedia data on the internet and social networks. Image-text search has a big role in many fields such as object recognition, video surveillance and audio-text recognition. The task of cross modal retrieval is to submit (1) (2). Cross modal retrieval can return more comprehensive results with information from different modalities compared to unimodal retrieval methods (3) (4). The key concept of multimodal hashing is transforming high multimodal data into a set of a small length compact code that preserves the cross-modality similarities of the original data. So, the multimodal similarity distance between the original data can be computed using a convenient metric between obtained codes. Faced to limited storage resource and the semantic gap between the representation and semantic label, makes performing accurate cross-modal retrieval on large scale datasets very challenging. Many approaches have been proposed to address the cross-modality retrieval problem and can be grouped into two categories:

Unsupervised methods that can learn the cross-modal feature representation space by maximizing the correlation between pairwise data from different modalities such as Co-Regularized Hashing (CRH) (8) that preserves the inter-modality similarity. In fact, the objective function consists in projecting simultaneously data from 0 for good generalization. Cross-view hashing (CVH) proposed by Kumar and Udupa (5), which extends the spectral hashing (SH) (6), the method reduces the dissimilarity between hash codes to preserve the distance between similar multimodal data modal. Zhu et al. (7) in linear cross-modal hashing (LCMH) used anchor maps to create similar hash code between similar data point intra modality, without constructing similarity maps

The advantage of Inter-media hashing (IMH) (9) is the fact it considers the differences between multiple modalities. It starts by exploring the correlations within each single modality according to similarity graph. Then it makes the binary codes of the multimodal data points relevant. Multimodal latent binary embedding (MLBE) (10) uses latent variable models to learn the binary hash codes. Multi-View Spectral Hashing (MVSH) (11) creates binary hash codes, using multiview information. Composite Hashing with Multiple Information Sources...
(CHMIS) (12) integrates information from several different sources into the binary hashing codes by adjusting the weights on each individual source for maximizing the coding performance, and enables fast conversion from query examples to their binary hashing codes. (13) Propose a novel Latent Semantic Sparse Hashing (LSSH) to perform cross-modal similarity search by employing sparse coding to capture the salient structures of images and Matrix Factorization to learn the latent concepts from text. Semantic Topic Multimodal Hashing (STMH) (14) is developed by considering latent semantic information in coding procedure. It first discovers clustering patterns of texts and robust factorizes the matrix of images to obtain multiple semantic topics of texts and concepts of images. In the proposed RFDH model (15), binary codes are directly learned based on discrete matrix decomposition, so that the large quantization error caused by relaxation is avoided.

Supervised multimodal hashing methods can create hash codes with preserving semantic similarities between modalities by approximating the pairwise similarity matrices based on supervised information such as semantic correlation maximization (SCM) (16) which proposes to seamlessly integrate semantic labels into the hashing learning procedure for large-scale data modelling. Semantics-Preserving Hashing (SePH) method (17) proposed to transform semantic affinities of training data as supervised information into a probability distribution learned hash codes in Hamming space via minimizing the Kullback-Leibler divergence. Multimodal discriminative binary embedding (MBE) (10) focused on learning discriminative hash codes by formulating the hash function learning in terms of classification, where the binary codes generated is discriminative and exploiting, the label information to discover the shared structures inside heterogeneous data. Cross Modal Hashing (CMDH) (18) learned a set of shared binary codes from different modalities, to remove the modality effectively in cross-modal multimedia retrieval. Supervised robust discrete multimodal hashing (SRDMH) (19) incorporates full label information into the hash functions learning to preserve the similarity in the original space. Supervised hierarchical cross-modal hashing (HiCHNet) (20) exploits the hierarchical labels of instances by the pre-established label hierarchy and characterize each modality of the instance with a set of layer-wise hash representations. Recently, favourable performance can be achieved using deep supervised multimodal hashing methods thanks to using deep neural networks while constructing the hash code. Those algorithms can highly capture nonlinear correlations between multimodal data such as deep semantic multimodal hashing network (DSMHN) (21), Cross-modal hashing with semantic deep embedding (CMSDE) (22) and Pairwise Correlation Discrete Hashing (PCDH) (23). We can conclude that supervised methods needs usually supervised information of the training data set which makes performing accurate cross-modal retrieval on large scale datasets very challenging.

In the current work, we develop a new Bayesian specific modality Matrix factorization hashing method which improves matrix factorization hashing using a fully Bayesian treatment of the probabilistic matrix factorization. We summarize the main contributions of our algorithm as follows:

1. The use of probabilistic matrix factorization learning algorithm improves the initialization of projection matrix in stead of using random ones.

2. The representation of the training data is adapted by subtracting the error found between different unified representations of modalities.

3. To have higher performance, we use the cosine distance to measure similarity.

4. The fully Bayesian inference model used in our methods significantly increases the model’s predictive accuracy.

5. Experimental results on two real-world datasets show that our method significantly outperforms the state of the art multimodal methods in terms of both accuracy and scalability.

II. Related work

In (24), Collective Matrix Factorization Hashing (CMFH) learns unified hash codes by collective matrix factorization with latent factor model from different modalities as one instance. The CMFH+ in (25) improves the collaborative matrix factorization by adapting the representation of the training data with subtracting the
error found between different unified representations of modalities. (26) proposes supervised matrix factorization hashing with quantitative loss (SMFH-QL) which generates hash codes via the class label and use matrix factorization to design hash codes from original multimodal data. Modality-specific Matrix Factorization Hashing (MsMFH) (27) factorizes the original feature representations into individual latent semantic representations, and then align those representations using an orthogonal transformation. Joint and individual matrix factorization hashing (JIMFH) (28) method learns unified hash codes using collective matrix factorization, and uses individual matrix factorization to get individual hash codes.

In (29) they presented the Probabilistic Matrix Factorization (PMF) model which scales linearly with the number of observations and, more importantly, performs well on the large, sparse, and very imbalanced Netflix dataset. To improve collaborative filtering in Netflix dataset, (30) presented a Bayesian treatment of the Probabilistic Matrix Factorization (PMF) model in which model capacity is controlled automatically by integrating over all model parameters and hyper parameters trained using Markov chain Monte Carlo methods. (31) proposes an adaptive initialisation to the Bayesian matrix factorization which still suffers from the cold-start problem where predictions of ratings for new items or of new user’s preferences are required in Netflix dataset.

Inspired by those works, this paper focuses on leaning modality specific latent representation using Bayesian inference to solve collective matrix factorization between different modalities and learn hash function which is to the best our knowledge not done before.

III. Proposed method

In this work and for ease of presentation, here we describe the method problem with only two modalities (image modality and text modality), which can be easily extended to cases with more than two modalities.

III.1. Notation and problem formulation

Table 1 shows notations and their signification respectively used to study this paper. Our algorithm accepts paired image-text data for input and processed to a specific treatment using hash coding to retrieve at the end the most relevant texts given an image query and vice versa.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>O</td>
<td>set of multimodal objs</td>
</tr>
<tr>
<td>X₁, X₂</td>
<td>Image Modality, Text Modality</td>
</tr>
<tr>
<td>U₁, U₂</td>
<td>Basic matrices for matrix factorization</td>
</tr>
<tr>
<td>W₁, W₂</td>
<td>Hash functions for matrix factorization</td>
</tr>
<tr>
<td>V</td>
<td>Shared latent representation</td>
</tr>
<tr>
<td>N</td>
<td>number of training objects</td>
</tr>
<tr>
<td>d₁, d₂</td>
<td>Dimension of image, text modality dataset</td>
</tr>
<tr>
<td>k</td>
<td>Number of latent factors</td>
</tr>
</tbody>
</table>
III.2. Framework overview

We learn semantic features from data in the Collective Matrix Factorization Hashing (24) by using matrix factorization:

\[ X^{(t)} = U_t V_t, \quad \forall t \in \{1, 2\} \]  

(1)

Where \( U_t \in \mathbb{R}^{d_t \times k} \) and \( V_t \in \mathbb{R}^{k \times n} \). Each column vector \( v_t \) is a latent semantic representation of the \( t \)-th view data \( X^{(t)} \).

Assuming that the interconnected data should have the identical latent semantic representation, \( X^{(1)} \) and \( X^{(2)} \) are factorized together with the constraint \( V_1 = V_2 = V \), where \( v_i \) is a column vector that can be used as a semantic hash code for the object data \( V_{i,t} \), \( i \in \{1, 2, ..., n\} \) from database.

When we have a new input, we extend the method by learning hash functions like (1).

\[ V = W_t X^{(t)} \quad \forall t \in \{1, 2\} \]  

(2)

We combine in the overall objective function data matrix factorization (a), projection matrix (b) and regularization term (c):

\[ G = \lambda \|X^1 - U_1 V_1\|_F^2 + (1 - \lambda) \|X^2 - U_2 V_2\|_F^2 \\ + \delta \|V - W_1 X^1\|_F^2 + \|V - W_2 X^2\|_F^2 \\ + \gamma \|V\|_F^2 + \|U_1\|_F^2 + \|U_2\|_F^2 + \|W_1\|_F^2 + \|W_2\|_F^2 \]  

(3)

Where \( \lambda \) is the balance parameter, \( \delta \) and \( \gamma \) are trade-off parameters. Our method have the objective of minimizing \( G \) and solve \( U_1, U_2, V, W_1 \) and \( W_2 \) using the Bayesian inference.
III.3. Learning Hash Function

III.3.1. Bayesian inference

The graphical model representing the collective Bayesian matrix factorization is shown in Fig. 1. The prior distributions over $U_t$ and $V_t$ are assumed to be Gaussian. So the likelihood will be given like (30) by

$$P(U_t | \mu_{U_t}, A_{U_t}) = \prod_{i=1}^{N} N(U^i | \mu_{U_t}, A_{U_t}^{-1})$$  \hfill (4)

$$P(V_t | \mu_{V_t}, A_{V_t}) = \prod_{i=1}^{N} N(V^i | \mu_{V_t}, A_{V_t}^{-1})$$  \hfill (5)

With the constraint $V_1 = V_2$

We define the Gaussian-Wishart priors on the text and image hyperparameters $\theta_{U_t} = \{\mu_{U_t}, A_{U_t}\}$ and $\theta_{V_t} = \{\mu_{V_t}, A_{V_t}\}$:

$$P(\theta_{U_t} | \theta_0) = p(\mu_{U_t} | \mu_0, \beta_0 A_{U_t}^{-1}) W(A_{U_t} | W_0, v_0)$$  \hfill (6)

$$P(\theta_{V_t} | \theta_0) = p(\mu_{V_t} | \mu_0, \beta_0 A_{V_t}^{-1}) W(A_{V_t} | W_0, v_0)$$  \hfill (7)

Where $W$ is the Wishart distribution with $v_0$ degrees of freedom and:

$$W(\Lambda | W_0, v_0) \propto |\Lambda|^{\frac{(v_0-D-1)}{2}} \exp(-\frac{1}{2} \text{tr}(W_0^{-1}\Lambda))$$

For convenience we define $\theta_0 = \{\mu_0, v_0, W_0\}$

For ease of addressing the problem, we will use the Gibbs sampling (32) algorithm to resolve the equation (4) and (5).

Gibbs sampling (32) consists of looping through the latent variables, sampling each one from its distribution conditional on the current values of all other variables.

We used a conjugate priors for the parameters and hyperparameters in our collective Bayesian matrix factorisation model which made sampling the conditional distribution over different modalities becomes easy

$$P(U_t | X_t, \nu_t, \theta_{U_t}, \alpha) = N(U^i_t | \mu^i_t, [\Lambda^i_t]^{-1})$$  \hfill (8)

$$\sim \prod_{j=1}^{M} [N(X^j_t | U^j_t V^j_t, \alpha^{-1})]^H_j P(U_t | \mu_{U_t}, A_{U_t})$$

Where
\[ A_t^{i+} = A_{U_t} + \alpha \sum_{j=1}^{M} [V_j V_j^T]^{i,j} \]  
(9)

\[ \mu_t^{i+} = [A_t^{i+}]^{-1} \left( \alpha \sum_{j=1}^{M} [V_j X_j^{i,j}] + A_{U_t} \mu_{U_t} \right) \]  
(10)

Notice that the conditional distribution over the latent feature matrix \( U_t \) (idem for \( V_t \)) factorizes into the product of conditional distributions over the individual feature vectors

\[ p(U_t^i|X^t, V, \theta_{U_t}) = \prod_{i=1}^{N} p(U_t^i|X^t, V, \theta_{U_t}) \]

Which allows us to sample those conditional distribution in parallel and speed up the sampler, this make the algorithm less time consuming in large scale datasets.

The conditional distribution over modalities hyperparameters conditioned on features feature matrix \( U_t \) (idem for \( V_t \)) is given by the Wishart-Gaussian distribution:

\[ p(\mu_{U_t}, A_{U_t}|U_t, \theta_0) = \mathcal{N} \left( \mu_{0^*} \left( \mu_0^* \mu_0^* \right)^{-1} \right) \mathcal{W} (A_{0^*}, v_0^*) \]  
(11)

Where:

\[ \mu_0^* = \frac{\beta_0 \mu_0 + NH}{\beta_0 + N}, \quad \beta_0^* = \beta_0 + N, \quad v_0^* = v_0 + N, \]

\[ \left[ \mathcal{W}_0^* \right]^{-1} = \left[ \mathcal{W}_0 \right]^{-1} + NL + \frac{\beta_0 N}{\beta_0 + N} (\mu_0 - H)(\mu_0 - H)^T, \]

\[ H = \frac{1}{N} \sum_{i=1}^{N} U_t^i, \quad L = \frac{1}{N} \sum_{i=1}^{N} U_t^i U_t^i^T, \]

With the additional condition that \( V_1 = V_2 = V \).

III.3.2 The proposed algorithm

The solution of the optimization problem is done by sampling the Bayesian inference over the modalities matrix using the Gibbs sampling algorithm like in (30). We minimize the objective function using eight steps.

Our method relies on eight steps that are:

1. Initialize \( U_1^1, V_1^1 \) using the probabilistic matrix factorization in (29).
2. Sample the hyperparameters \( k \) times

\[ \theta_{U_t} \sim p(\theta_{U_t}|U_t^k, \theta_0) \quad \forall t \in \{1, 2\} \]  
(12)

\[ \theta_{V_t} \sim p(\theta_{V_t}|V_t^k, \theta_0) \quad \forall t \in \{1, 2\} \]  
(13)

3. For every sampling of the hyperparameters we calculate a new iteration of \( U_t \) matrix in parallel

\[ U_t^{k+1} \sim P(U_t^i|X^t, V_t^k, \theta_{U_t}^k) \quad \forall t \in \{1, 2\} \]  
(14)

4. For every sampling of the hyperparameters we calculate a new iteration of \( V_t \) matrix in parallel

\[ V_t^{k+1} \sim P(U_t^i|X^t, U_t^{k+1}, \theta_{V_t}^k) \quad \forall t \in \{1, 2\} \]  
(15)

5. Calculate \( U_t \) and \( V_t \) until \( |V_1 - V_2| < \varepsilon \) \quad \forall t \in \{1, 2\}

6. Measure the projection error matrix like:
\[ \text{Err}^t = X^t - U_t V \quad \forall \, t \in \{1, 2\} \tag{16} \]

7- Data representation of the training data is adjusted when we substruct the equation (16) from \( X^t \) like:

\[ X^t = X^t - \text{Err}^t \quad \forall \, t \in \{1, 2\} \tag{17} \]

8- Fixing \( V \), we calculate \( W_1 \) and \( W_2 \) using ascent method for linear triangular systems.

The algorithm is summarized in figure 2

**Algorithm 1 Initialization and Indexing**

**Input:**

Data matrix \( X^t \) where \( t = 1, 2 \).

**Output:**

Hash codes \( V \).

Projection matrix \( W_t \).

Data representation \( X^t \).

**Repeat**

1- Initialize model parameters \( U_t^1, V_t^1 \) where \( t = 1, 2 \).

2- For \( k = 1 \) to \( M \)
   
   - Sample the hyperparameters
     \[ \theta_{U_t}^k \sim p(\theta_{U_t}|U_t^k, \theta_0) \quad \forall \, t \in \{1, 2\} \]
     \[ \theta_{V_t}^k \sim p(\theta_{V_t}|V_t^k, \theta_0) \quad \forall \, t \in \{1, 2\} \]
   
   - For \( i = 1 \) to \( d_t \) where \( t = 1, 2 \)
     
     Calculate a new iteration of \( U_t \) matrix in parallel using equation 14
     
   - For \( i = 1 \) to \( N \)
     
     Calculate a new iteration of \( V_t \) matrix in parallel using equation 15

**While** \( |V_1 - V_2| < \varepsilon \)

3- Calculate \( W_t \), \( t = 1, 2 \).

4- Calculate \( \text{Err}^t \) using equation (16), \( t = 1, 2 \).

5- Calculate \( X^t \) using equation (17), \( t = 1, 2 \).

6- Return \( W_t, V, X^t \).

**Figure 3. Initialization and indexing algorithm.**

**III.3.3 Searching algorithm**

Our method uses the Bayesian inference to compute the latent semantic vectors \( V \). We consider each \( V_j \) as a hash code of the \( j \)-th data from the database.

We generate the hash code for a new query like:

\[ v_q = W_t \times (q^t - \sum \frac{x_i^t}{n}) \tag{18} \]

We calculate the similarity between two vectors \( X \) and \( Y \) using the cosine distance like:
\[ D_C(X, Y) = 1 - \frac{X^Y}{||X||||Y||} \]  

(19)

The searching algorithm is summarized in figure 3.

**Algorithm 2 Searching**

**Input**

\[ W^t, V, q^t \text{ and } X^t \quad t=1, 2. \]

**Output**

K-NN data points

1- Train \( q^t \) projection by equation (18), \( t=1, 2. \)

2- Measure the Cosine similarity between \( V \) and \( v_q \) using equation (19).

3- Retrieve the K-NN from the new \( X^t \).

Figure 4. Searching algorithm.

IV. Experiments

We realised experiments on two real world multimodal datasets, we are used to check the efficiency of our hashing method. All our experiments are carried out on a computer with Intel (R) CPU X7560@2.27 GHz and 22 GB RAM.

IV.1. Datasets

**NUS-WIDE** (33): is a multiclass dataset containing 269,648 annotated images from Flickr. Images are manually classified into 81 classes (an image can belong to more than one class) and represented as bags of 500 dimensions of SIFT (BoF, used as the first modality) and bags of 1000 dimensions of text tags. (Tags, used as second modality). We select 186,577 image-tag pairs that belong to the 10 largest concepts.

**Wiki** (34): is explored from featured Wikipedia articles. It consists of 2,866 documents which are image-text pairs and annotated with semantic tags of 10 classes. a 128-dimensional SIFT feature vector represents each image and each text is represented by a 10-dimensional topic vector.

IV.2. Experiments results

IV.2.1 Evaluation metrics

We evaluate the retrieval performance using the mean average precision (MAP) which is defined like:

\[ MAP = \frac{1}{L} \sum_{r=1}^{R} P(r) \times \text{rel}(r) \]  

(12)

Where \( \text{rel}(r) = \begin{cases} 1, & \text{if the rank is relevant} \\ 0, & \text{otherwise} \end{cases} \)

And \( R, P(r) \) and \( L \) denote respectively the number of retrieved results, the percentage of relevant results in the first \( r \) top-ranked retrieved matches and the number of ground-truth neighbours of a query.

We set \( v_0 = k \) and \( W_0 = k \times k \) identity matrix for both \( U_t \) and \( V_t \) then we choose for the convenience \( \mu_0 = 0 \) by symmetry. We loop the Gibbs sampling 2 times per experiment, we set the observation noise \( \alpha_1 = \alpha_2 = \alpha = 2. \)

IV.2.2 Rival Methods

Our method is compared on two cross modal retrieval tasks: querying, mutatis mutandis, image database by text sets. We evaluate our method with several known and recent multimodal hashing methods: CVH (5), SMH (35), CCA (36), SMFH, CMFH, CMFH+, LSSH, SMFH (26), SCM (16), DCH (23), LCMFH (7), EDSH (37), MSMFH (27). We carefully adjusted their parameters and reported the best of their results.
IV.4. Experiments results

IV.4.1 Results on Wiki

As usual, for the Wiki dataset 80% of the data is used as training set and 20% as the query set. We reported the MAP values for our method and for all rival methods on Table 2 under setting of 16, 32, 64 and 128 bits respectively. The Table 2 evaluates the proposed algorithm on wiki data set. This evaluation significantly shows that the CBMFH achieved better performance on this benchmark of both text to image and image to text sides, and the performance gain of the proposed method is more than 10%. CBMFH has done remarkable MAP. Compared to the rival methods on both views and where varying the code length, our method is more efficient especially in the text to image task which is due to the fact that the Bayesian matrix factorization can successfully find better latent semantic features from text.

Table 3. MAP results on WIKI dataset.

<table>
<thead>
<tr>
<th>Task (request)</th>
<th>Method</th>
<th>Wiki 16 bits</th>
<th>Wiki 32 bits</th>
<th>Wiki 64 bits</th>
<th>Wiki 128 bits</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Image To Text</td>
<td>CVH</td>
<td>0.2041</td>
<td>0.1604</td>
<td>0.1296</td>
<td>0.1308</td>
</tr>
<tr>
<td></td>
<td>SMH</td>
<td>0.2257</td>
<td>0.2459</td>
<td>0.249</td>
<td>0.2555</td>
</tr>
<tr>
<td></td>
<td>CCA</td>
<td>0.1578</td>
<td>0.1393</td>
<td>0.1282</td>
<td>0.1237</td>
</tr>
<tr>
<td></td>
<td>CMFH</td>
<td>0.2437</td>
<td>0.2523</td>
<td>0.2575</td>
<td>0.2611</td>
</tr>
<tr>
<td></td>
<td>CMFH+</td>
<td>0.2642</td>
<td>0.2675</td>
<td>0.2675</td>
<td>0.2675</td>
</tr>
<tr>
<td></td>
<td>LSSH</td>
<td>0.2297</td>
<td>0.2532</td>
<td>0.2406</td>
<td>0.2336</td>
</tr>
<tr>
<td></td>
<td>SMFH</td>
<td>0.2276</td>
<td>0.2514</td>
<td>0.2533</td>
<td>0.2564</td>
</tr>
<tr>
<td></td>
<td>SCMA</td>
<td>0.2474</td>
<td>0.2514</td>
<td>0.2533</td>
<td>0.2564</td>
</tr>
<tr>
<td></td>
<td>DCH</td>
<td>0.3423</td>
<td>0.3599</td>
<td>0.3806</td>
<td>0.3848</td>
</tr>
<tr>
<td></td>
<td>LCMFH</td>
<td>0.3114</td>
<td>0.3337</td>
<td>0.3455</td>
<td>0.3606</td>
</tr>
<tr>
<td></td>
<td>EDSH</td>
<td>0.3871</td>
<td>0.3942</td>
<td>0.3950</td>
<td>0.3956</td>
</tr>
<tr>
<td></td>
<td>MSMFH</td>
<td>0.3832</td>
<td>0.3953</td>
<td>0.4045</td>
<td>0.4050</td>
</tr>
<tr>
<td></td>
<td>Our</td>
<td><strong>0.4133</strong></td>
<td><strong>0.4135</strong></td>
<td><strong>0.4136</strong></td>
<td><strong>0.4136</strong></td>
</tr>
</tbody>
</table>

| Text To Image   | CVH    | 0.2962       | 0.1944       | 0.1337       | 0.1125        |
|                 | SMH    | 0.2341       | 0.241        | 0.2458       | 0.26          |
|                 | CCA    | 0.1755       | 0.1619       | 0.155        | 0.155         |
|                 | CMFH   | 0.6107       | 0.6282       | 0.6376       | 0.6471        |
|                 | CMFH+  | 0.6588       | 0.6558       | 0.6558       | 0.6558        |
|                 | LSSH   | 0.6134       | 0.6296       | 0.6349       | 0.6304        |
|                 | SMFH   | 0.5590       | 0.6473       | 0.6678       | 0.6617        |
|                 | SCMA   | 0.3819       | 0.4479       | 0.4418       | 0.4405        |
|                 | DCH    | 0.6999       | 0.7051       | 0.7117       | 0.7222        |
|                 | LCMFH  | 0.6969       | 0.7042       | 0.7094       | 0.7408        |
|                 | EDSH   | 0.5581       | 0.5661       | 0.5662       | 0.5732        |
|                 | MSMFH  | 0.5891       | 0.6082       | 0.6183       | 0.6221        |
|                 | Our    | **0.7525**   | **0.7528**   | **0.7531**   | **0.7532**    |

From results above, it is easy to see that our method has an almost the same MAP result over the variation of the code length which gives stability to our algorithm. When we retrieve the factorization error from data representation and we use the new representation to search relevant data which makes the performance of our algorithm more stable under the variation of the code length.

IV.4.2 Results on NUS-WIDE

Despite the cost in terms of training time of rival methods on NUS-WIDE, we evaluated and compared our algorithm to several recent methods in the literature. Better MAP result gains are obtained on the large scale NUS-WIDE, mainly in the Text to image retrieval. Table 3 shows that our method is more efficient in terms of search accuracy and precision. In addition, we can observe that our method remains invariant under the variation of the code length. This is an obvious consequence since we reinject the factorization error and we change the initial data representation. We can improve the research performance of the method by using cosine distance.
### Table 4. MAP results on NUS-WIDE dataset.

<table>
<thead>
<tr>
<th>Task (request)</th>
<th>Method</th>
<th>NUS-WIDE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>16 bits</td>
</tr>
<tr>
<td>Image To Text</td>
<td>CVH</td>
<td>0.5216</td>
</tr>
<tr>
<td></td>
<td>SMH</td>
<td>0.2918</td>
</tr>
<tr>
<td></td>
<td>CCA</td>
<td>0.3561</td>
</tr>
<tr>
<td></td>
<td>CMFH</td>
<td>0.3950</td>
</tr>
<tr>
<td></td>
<td>CMFH+</td>
<td>0.4054</td>
</tr>
<tr>
<td></td>
<td>LSSH</td>
<td>0.5047</td>
</tr>
<tr>
<td></td>
<td>SMFH</td>
<td>0.5671</td>
</tr>
<tr>
<td></td>
<td>SCM</td>
<td>0.5496</td>
</tr>
<tr>
<td></td>
<td>DCH</td>
<td>0.5686</td>
</tr>
<tr>
<td></td>
<td>LCMFH</td>
<td>0.6191</td>
</tr>
<tr>
<td></td>
<td>EDSH</td>
<td>0.5141</td>
</tr>
<tr>
<td></td>
<td>MSMFH</td>
<td>0.5521</td>
</tr>
<tr>
<td></td>
<td>Our</td>
<td>0.6561</td>
</tr>
<tr>
<td>Text To Image</td>
<td>CVH</td>
<td>0.5462</td>
</tr>
<tr>
<td></td>
<td>SMH</td>
<td>0.3552</td>
</tr>
<tr>
<td></td>
<td>CCA</td>
<td>0.3550</td>
</tr>
<tr>
<td></td>
<td>CMFH</td>
<td>0.4175</td>
</tr>
<tr>
<td></td>
<td>CMFH+</td>
<td>0.5163</td>
</tr>
<tr>
<td></td>
<td>LSSH</td>
<td>0.6201</td>
</tr>
<tr>
<td></td>
<td>SMFH</td>
<td>0.5843</td>
</tr>
<tr>
<td></td>
<td>SCM</td>
<td>0.5496</td>
</tr>
<tr>
<td></td>
<td>DCH</td>
<td>0.7376</td>
</tr>
<tr>
<td></td>
<td>LCMFH</td>
<td>0.7052</td>
</tr>
<tr>
<td></td>
<td>EDSH</td>
<td>0.6341</td>
</tr>
<tr>
<td></td>
<td>MSMFH</td>
<td>0.6792</td>
</tr>
<tr>
<td></td>
<td>Our</td>
<td>0.7934</td>
</tr>
</tbody>
</table>

For data with small code length, our method seems more efficient than the state of the art method. Despite that DCH achieves better performance on 128 bits code length, in general our method remains more accurate especially when the latent semantic representation is small enough.

#### IV.4.3 Scalability

This subsection is devoted to evaluate the scalability of different methods compared to ours. We start by experimenting the training time of different baseline methods on NUS-WIDE dataset when we vary the code length. Results are summarized in table 4. The benchmark is done over the methods of the state of the art that have good MAP performance: LSSH, DCH, CMFH, CMFH+ and LCMFH.
Table 4. Training time (in second) on NUS-WIDE dataset under the code length variation.

<table>
<thead>
<tr>
<th>Method</th>
<th>NUS-WIDE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>16 bits</td>
</tr>
<tr>
<td>LSSH</td>
<td>762.61</td>
</tr>
<tr>
<td>DCH</td>
<td>12.82</td>
</tr>
<tr>
<td>CMFH</td>
<td>545.95</td>
</tr>
<tr>
<td>CMFH+</td>
<td>550.01</td>
</tr>
<tr>
<td>LCMFH</td>
<td>5.44</td>
</tr>
<tr>
<td>Ours</td>
<td>18.21</td>
</tr>
</tbody>
</table>

From table 4 we can see that LSSH, CMFH and CMFH+ require much training time than DCH, LCMFH and our method. Additionally, the last three have a good training times despite that LCMFH achieved the best results than DCH and our method. However in summary, the proposed method shows the best retrieval performance in a competitive time.

V. Conclusion

We have proposed a new method relied on Collaborative Bayesian Matrix Factorization and have improved the precision by reducing the error between multimodal data and their hash code. The method exposed here is based on a probabilistic matrix factorization algorithm to initialize projection matrix. The similarity was measured using the cosine distance. Several experiments on two real datasets have proved that our method may significantly outperform and probably improve the state of the art methods in term of scalability and accuracy.

Figure 3 show the top K-Precision curves for all the baseline methods.
VI. RÉFÉRENCES


29- MNIH, Andriy et SALAKHUTDINOV, Russ R. Probabilistic matrix factorization.


http://www.svcl.ucsd.edu/projects/crossmodal/.


