IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received December 7, 2020, accepted December 24, 2020, date of publication January 12, 2021, date of current version January 26, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3051190

Optimizing Non-Differentiable Metrics for Hashing

YIWEN WEI[1], DAYONG TIAN[2], JIAO SHI[2], (Member, IEEE), AND YU LEI[2], (Member, IEEE)
[1]School of Optoeletronics and Physics, Xidian University, Xi'an 710071, China
[2]School of Electronics and Information, Northwestern Polytechnical University, Xi'an 710072, China

Corresponding author: Dayong Tian (dayong.tian@nwpu.edu.cn)

This work was supported in part by the National Science Foundation of China under Grant 61806166 and Grant 61801362, in part by the Natural Science Foundation of Shaanxi Province under Grant 2020JQ-197, and in part by the Fundamental Research Funds for the Central Universities, Northwestern Polytechnical University (NPU) under Grant G2018KY0303.

ABSTRACT Image hashing embeds the image to binary codes which can boost the efficiency of approximately nearest neighbors search. F-measure is a widely-used metric for evaluating the performance of hashing methods. However, it is non-differentiable and hence it has not been used as an object function for hashing. Heuristic algorithms, e.g. evolutionary computation and particle swarm optimization (PSO), are good at optimizing non-differentiable objectives, while they are inefficient in very high-dimensional variables which are commonly used in hashing models. To address this contradict, we propose a scheme to bridge hashing methods and F-measure objective using PSO. The hashing methods are used to generate real-valued codes for images and then the parameters of quantization procedure are optimized by PSO. Our scheme can incorporate a wide range of hashing methods, heuristic optimization algorithms and non-differentiable metrics. Experimental results demonstrate that our scheme can be used to further improve the performance of existing hashing methods.

INDEX TERMS Image hashing, approximately nearest neighbor search, particle swarm optimization, F-measure.

I. INTRODUCTION

Due to the large amount of images available on Internet, hashing that embeds images to binary codes has attracted a lot of interests. As digital computers handle binary codes much more efficiently than any other types of numbers, hashing can boost the speed of approximately nearest neighbor search.

Classical hashing methods are generally modeled as optimization problems whose object function are differentiable so as to iterative gradient-based algorithms can be used. To construct a differentiable objective, traditional hashing methods commonly generate real-valued codes as an intermediate and then adopt a quantization method to generate the final binary hashing codes. Although F-measure is a widely-used metric for evaluating hashing methods, it has not been used as an objective in hashing methods because it is non-differentiable. However, directly maximizing F-measure is an intuitive way to improve the performance of classical hashing methods.

Although optimizing non-differetiable metrics have attracted interests in classification problem, they have never been used as objective in hashing problem. Recent works on optimizing non-differentiable metrics for classification problems focus on continuous approximation of F-measure [1] and learning surrogate losses [2]. They all need ground-truth labels in training data sets. However, there are no ground-truth hashing codes. Hence, these methods cannot be directly used for hashing problem.

Heuristic algorithms are good at non-differentiable objectives, but they lack of ability in handling high-dimensional variables. For a simple hashing method, such as ITerative Quantization [3], the dimension of variables is $l \times l$, where l is the code length. For a medium length, say 64 bits, the dimension of variable is 4096, which is difficult to optimize only using heuristic algorithms.

To solve this dilemma, we propose a scheme to bridge traditional hashing methods and non-differentiable objectives. Our scheme consists of two stages, as shown in Fig. 1. First, we generate real-valued codes using traditional hashing methods. Then, we calculate the parameters of the proposed quantization method by maximizing F-measure using Particle Swarm Optimization (PSO). Our scheme is available for a wide range of hashing methods,

The associate editor coordinating the review of this manuscript and approving it for publication was Ran Cheng.

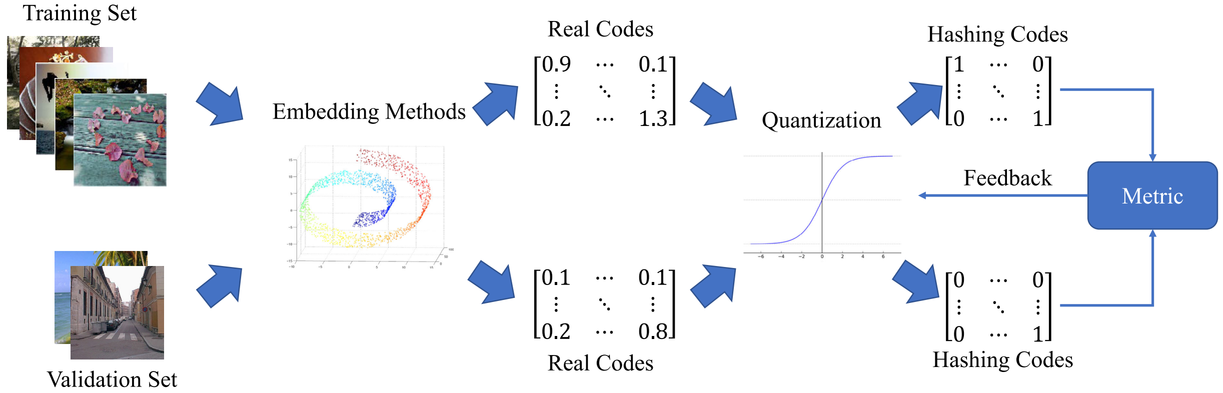

FIGURE 1. The scheme of our proposed method. We randomly select a subset from training set as validation set. We adopt a traditional hashing method without quantization as an embedding method to generate real codes. We propose a parameterized quantization method to generate binary hashing codes. The parameters are learned by optimizing the metric, such as F-measure.

heuristic optimization algorithms and non-differentiable metrics.

The main contribution of our work is a scheme that can optimize non-differentiable metrics for hashing problem. By directly optimizing the metrics, the proposed method can avoid the impacts of correlated hashing bits and defects of methods that try to de-correlate hashing bits, such as orthogonality (details in Subsection III-E).

This paper is organized as following. The related works are reviewed in Section II. The proposed method is described in Section III. The experimental results are reported in Section IV. The conclusive remarks are given in Section V.

II. RELATED WORKS

In this section, we will briefly review the representative hashing methods and efforts have been made for optimizing non-differentiable metrics.

A. HASHING METHODS

Hashing methods can be divided into two categories based on whether they depend on data. Locality-sensitive hashing (LSH) [4] is a well-known data-independent hashing method. To handle nonlinear data structure, kernels are adopted in LSH [5], [6].

The key idea of data-dependent hashing methods is maximizing the correlation between structures of data and hashing codes. Directly maximizing the correlation need to compute affinity matrix which requires to compute mutual distances between any pairs of data. For large scale dataset, it becomes intractable. Spectral Hashing (SH) [7] solves a relaxed mathematical problem to avoid computing affinity matrix.

Anchor Graph Hashing (AGH) [8] uses anchor points to construct a sparse matrix to approximate the affinity matrix. Discrete Graph Hashing follows this idea and project hashing code matrix to orthogonal and balanced solution space to de-correlate hashing bits.

Minimizing quantization errors is a promising way to generate hashing codes due to its computation efficiency. ITerative Quantization (ITQ) [3] rotates the principal components by an orthogonal matrix to minimize the quantization errors. The variances of each principal component are different, which means the importance of each principal component is different. IsoH [9] balances the principal components by dividing their corresponding variances. Besides Principal Component Analysis (PCA), Linear Discriminant Analysis can be also used [10]. These methods pre-compute the projections of original data. Neighborhood Discriminant Hashing [11] computes the projections of original data during the optimization procedure. Nonlinear embedding methods are also used to handle the nonlinear data structure. Inductive Manifold Hashing (IMH) [12], [13] learns a nonlinear manifold on a small subset and inductively insert the remaining data.

Orthogonality and balance constraints are expected to be good regularizations for hashing codes [7]. However, they are difficult to be fulfilled because the binary code matrix cannot be directly optimized by gradient-based algorithms. DGH projects code matrix to orthogonal and balanced space in each iteration. By setting a parameter to infinity, it can generate orthogonal and balanced code matrix. Nevertheless, it is impractical to set the parameter to infinity. Methods based on minimizing quantization errors apply orthogonality and balanced regularizations on the intermediate real matrices of hashing code matrices. However, it has been proven that quantization will break the orthogonality and balance except for some extremely ideal cases [14].

Matrix factorization is also widely used in hashing methods. Ding *et al.* [15] use collective matrix factorization for multimodal hashing. Lu *et al.* [16] use matrix decomposition to extract latent semantic features for generating discriminative binary codes. Liu *et al.* [17] notice the sparsity of data structure and propose an adaptively sparse matrix factorization for hashing.

Although the performances of the above-mentioned shallow hashing methods can be improved by extracting features using deep neural networks, deep hashing models can achieve better performances. Deep transfer hashing (DTH) [18] substitutes the principal coefficients and orthogonal rotation matrix in ITQ with a deep neural network. Deep binary descriptors (DeepBit) [19] uses VGGNet [20] to extract the features of images and learns the hashing codes with a combined object function of quantization loss, balanced regularization and rotation invariant objective. Stochastic generative hashing (SGH) [21] learns hashing codes by minimum description length principle so as to maximally compress the dataset as well as regenerate outputs from the codes. Semantic structure-based unsupervised hashing (SSDH) [22] uses two half Gaussian distributions to estimate pairwise cosine distances of data points and assign any two data points with obviously smaller distance as semantically similar pair. A pairwise loss function to preserve this semantic structure are used to train the neural network. DistillHash [23] learns confidence similarity signals first to "supervise" the subsequent hashing code generating. Lu *et al.* [24] integrate the quantization process and ranking process into a unified architecture. Shen *et al.* [25] found that graphs built from original data introduce biased prior knowledge of data relevance and therefore they propose a twin-bottleneck autoencoder to trace the code-driven similarity graph.

B. OPTIMIZING NON-DIFFERENTIABLE METRICS

In classification problem, different thresholding strategies usually lead to different precision and recall. To comprehensively evaluate the performance of classifiers, metrics like mean average precision (MAP), F-measure are widely-used. Because they are non-differentiable, they are rarely used as object functions. Recently, directly optimizing these metrics attracts interests in machine learning community.

Pioneer works on maximizing F-measure focus on empirical utility maximization and decision-theoretic approach [26]. Later on, optimal thresholding of classifiers are used to maximize F-measure [27], [28]. Parambath *et al.* [29] use cost-sensitive classification to maximizing F-measure. Recent works try to optimize the tight bounds of F-measure [30] and use continuous and differentiable approximation of F-measure [1], [31].

Another promising way to optimizing non-differentiable metrics is relaxed surrogates. Eban *et al.* [32] define relaxation forms of building blocks of a confusion matrix, e.g. true positives, true negatives, etc and combine the building block relaxation to create a final surrogate for area under curve (AUC) metric. Berman *et al.* [33] use Lovasz softmax loss to approximate the Jaccard index metric. Grabocka *et al.* [2] use surrogate neural network to approximate non-differentiable metrics. Their method can be used to optimize many metrics, e.g. AUC, Jaccard Index, F-measure, etc.

All the above-mentioned methods focus on classification problem where ground-truth labels are available. For example, the computation of the continuous approximation of F-measure proposed in [1] use both of ground-truth labels and outputs of classifiers. To learn the surrogate neural network in [2], the ground-truth labels are also required in the objective function. Nevertheless, there are no ground-truth hashing codes for any data and hence we cannot estimate F-measure in such ways.

III. METHODOLOGY

Let us define some notations. For simplicity in mathematical description, we treat a datum as a vector and all the data in a dataset forms up a data matrix $\mathbf{X} \in \mathbb{R}^{n\times d}$, where n is the number of data and d is the dimension. Although image data are usually three-dimensional tensors, they can be flatten to vectors. For example, an $128 \times 128 \times 3$ image can be flatten to an 1×41952 row vector. $\mathbf{Y} \in \mathbb{R}^{n\times l}$ is the real-valued codes generated by traditional hashing methods, where l is the code length. $\mathbf{B} \in \{-1, 1\}^{n\times l}$ is the binary hashing code matrix. F-measure is defined as:

$$\beta \cdot \frac{precision \cdot recall}{precision + recall}, \tag{1}$$

where β is a positive constant. Without losing generality, we set β as 2 in our paper. The *precision* is defined as:

$$\frac{retrieved \quad true \quad positives}{number \quad of \quad all \quad retrieved \quad items}, \tag{2}$$

and the *recall* is defined as:

$$\frac{retrieved \quad true \quad positives}{number \quad of \quad all \quad true \quad neighbors}. \tag{3}$$

precision and *recall* are used to evaluate the retrieval performance in two different views. *precision* focuses on the accuracy of retrieved results, while *recall* focuses on how many true neighbors are retrieved. F-measure is a balanced metric combining *precision* and *recall*.

A hashing method can be seen as a function that maps images to binary codes:

$$f : \mathbf{X} \rightarrow \mathbf{B} \tag{4}$$

A. GENERATING REAL-VALUED CODES

As an example, ITQ is used for generating real-valued codes. ITQ is modeled as a minimization problem:

$$\begin{aligned} &\underset{\mathbf{B},\mathbf{R}}{argmin}\ \|\mathbf{B} - \mathbf{XWR}\|_F^2 \\ &s.t. \quad \mathbf{R}^\top\mathbf{R} = \mathbf{I}, \mathbf{B} \in \{-1, 1\}^{n\times l}, \end{aligned} \tag{5}$$

where $\mathbf{W}$ is principal component coefficients corresponding to the top l variances and $\mathbf{I}$ is the identity matrix. Eq. (5) is minimized by iteratively updating $\mathbf{B}$ and $\mathbf{R}$. $\mathbf{B}$ is updated by

$$\mathbf{B} = sign(\mathbf{XWR}). \tag{6}$$

To update $\mathbf{R}$, the singular value decomposition (SVD) is used, i.e. $\mathbf{B}^\top\mathbf{XW} = \mathbf{U}\Sigma\mathbf{V}^\top$. Then, $\mathbf{R}$ is updated by

$$\mathbf{R} = \mathbf{V}^\top\mathbf{U}. \tag{7}$$

sign() function in Eq. (6) acts as a quantization step. We use $\mathbf{Y}$ to represent the real-valued codes generated by hashing methods, i.e. $\mathbf{Y} = \mathbf{XWR}$ in Eq. (5). It is defined as

$$sign(x) = \begin{cases} 1, & if\ x > 0 \\ 0, & if\ x = 0 \\ -1, & if\ x < 0 \end{cases}. \tag{8}$$

Algorithm 1 Algorithm for Calculating F-Measure

```
Require: $\mathbf{B}_v, \mathbf{B}_d, \mathbf{S}$.
Ensure: F-measure.
1: Calculate the mutual Hamming distance between each pair of data in $\mathbf{B}_v$ and $\mathbf{B}_d$ to make up matrix $\mathbf{D} \in \{0, 1, \ldots, l\}^{(n-m)\times m}$.
2: Select those pairs whose Hamming distances are no greater than the preset Hamming radius, say 2.
3: Calculating *precision* and *recall* using Eq. (2) and Eq. (3), respectively.
4: Calculating F-measure using Eq. (1).
```

We substitute *sign* by

$$Q(x_i) = \begin{cases} 1, & \text{if } a_i x_i + c_i \geq 0 \\ -1, & \text{if } a_i x_i + c_i < 0 \end{cases}, \tag{9}$$

where x_i is the i-th element of vector $\mathbf{x}$ of length l and $i = \{1, 2, \ldots, l\}$. a_i and c_i are the i-th element of vector $\mathbf{a}$ and $\mathbf{c}$ which are two variables to be optimized by PSO.

B. F-MEASURE AS OBJECT FUNCTION

We randomly select m data points from the training dataset as a validation set. The F-measure is calculated using these m data points as queries which make up $\mathbf{X}_v \in \mathbb{R}^{m\times d}$ and corresponding $\mathbf{Y}_v \in \mathbb{R}^{m\times l}$ and the remaining $n - m$ data points as database which is denoted as $\mathbf{X}_d^{(n-m)\times l}$ and corresponding $\mathbf{Y}_d^{(n-m)\times l}$. First, the real-valued codes $\mathbf{Y}_d$ and $\mathbf{Y}_v$ are quantized by Eq. (9) and get binary code matrix $\mathbf{B}_d \in \{-1, 1\}^{(n-m)\times l}$ and $\mathbf{B}_v \in \{-1, 1\}^{m\times l}$. Second, the F-measure is calculated by **Algorithm 1**.

In **Algorithm 1**, $\mathbf{S} \in \{0, 1\}^{(n-m)\times m}$ is the groundtruth matrix. If the i-th data point in the database is the true neighbor of the j-th data point in the query set, then $S_{ij} = 1$, otherwise $S_{ij} = 0$. $\mathbf{S}$ can be calculated in two different ways. If the labels are available, then $\mathbf{S}$ can be calculated directly by matching the labels of two data points. On the other hand, if the labels are unavailable, we can define the top $p\%$ neighbors searched by Euclidean distances in raw data are true neighbors. The mutual Hamming distance matrix $\mathbf{D}$ can be efficiently calculated by

$$\begin{cases} (1 - \mathbf{B}'_d)\mathbf{B}'^{\top}_v + \mathbf{B}'_d(1 - \mathbf{B}'_v)^{\top} \\ \mathbf{B}'_d = \dfrac{\mathbf{B}_d + 1}{2}, \mathbf{B}'_v = \dfrac{\mathbf{B}_v + 1}{2} \end{cases} \tag{10}$$

C. OPTIMIZATION

Let $\mathbf{t} = \{a_1, a_2, \ldots, a_i, \ldots, a_l, c_1, c_2, \ldots, c_i, \ldots, c_l\}$ be a $2l$-dimensional variable. To find $\mathbf{t}$ that maximize F-measure, we use PSO as the optimization algorithm. PSO updates the variable $\mathbf{t}$ and a auxiliary variable $\mathbf{v}$ called velocity by

$$\begin{cases} \mathbf{v}^{k+1} = \mathbf{v}^k + \alpha_1 r_1(\mathbf{p}_{best} - \mathbf{t}) + \alpha_2 r_2(\mathbf{g}_{best} - \mathbf{t}) \\ \mathbf{t}^{k+1} = \mathbf{t}^k + \mathbf{v}^{k+1} \end{cases}, \tag{11}$$

Algorithm 2 Overall Scheme

```
Require: $\mathbf{X}$, the maximum iteration $K$
Ensure: $\mathbf{B}$
    repeat
        1. Split $\mathbf{X}$ to database $\mathbf{X}_d$ and query set $\mathbf{X}_v$.
        2. Using a traditional hashing method trained on $\mathbf{X}_d$ to generate real-valued code matrix $\mathbf{Y}_d$ and $\mathbf{Y}_v$.
        3. Calculate F-measure using Algorithm 1.
        4. Find the current best solution $\mathbf{p}_{best}$ and the best solution ever emerged $\mathbf{g}_{best}$.
        5. Update $\mathbf{T}$ using Eq. (12).
        6. $k = k + 1$
    until $k = K$
```

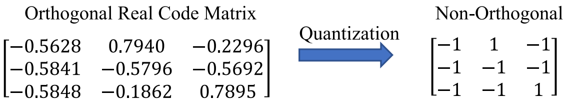

FIGURE 2. How quantization breaks orthogonality. The real code matrix is orthogonal. After it is quantized by function, the resulting binary code matrix is not orthogonal.

where α_1 and α_2 are two constants, r_1 and r_2 are two random numbers sampled from a certain probabilistic distribution, $\mathbf{p}_{best}$ is the best solution in current iteration (the k-th iteration), $\mathbf{g}_{best}$ is the best solution ever emerged in the previous k iterations. PSO will randomly initialize q variables and each variable is updated using Eq. (11). Let $\mathbf{T} \in \mathbb{R}^{q\times 2l}$ be the matrix of which each row corresponds to an instance of variable $\mathbf{t}$. Let $\mathbf{V} \in \mathbb{R}^{q\times 2l}$ be the matrix of which each row is the corresponding velocity of an instance of $\mathbf{t}$. PSO can be written in matrix form:

$$\begin{cases} \mathbf{V}^{k+1} = \mathbf{V}^{k+1} + \alpha_1 r_1(\mathbf{1}^{q\times 1}\mathbf{p}_{best} - \mathbf{T}) + \alpha_2 r_2(\mathbf{1}^{q\times 1}\mathbf{g}_{best} - \mathbf{T}) \\ \mathbf{T}^{k+1} = \mathbf{T}^k + \mathbf{V}^{k+1} \end{cases}. \tag{12}$$

Our overall scheme is shown in **Algorithm 2**.

D. IMPLEMENTATION DETAILS

Parameter setting and initialization: In Eq. (12), α_1 and α_2 are set as 2, and r_1 and r_2 are sampled from uniform distribution $U(0, 1)$. The maximum iteration $K = 500$. We use 10% data points as validation set, i.e. $m = 10\%n$. $\mathbf{T}$ are randomly initialized using samples from normal distribution $N(0, 0.1)$. $\mathbf{V}$ is initialized as a zero matrix.

E. WHY DOES IT WORKS?

As the code length increases, the correlation between hashing code bits may degrade the retrieval performance. Researchers generally add an orthogonality constraint on the code matrix to handle this problem. However, most hashing methods apply the orthogonality constraint on the real code matrix to avoid solving an NP-hard problem. Even though one can get an orthogonal real code matrix, the quantization will completely break the orthogonality (Fig. 2).

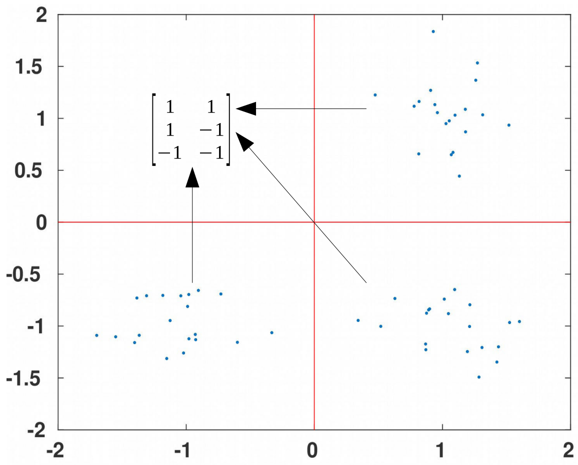

FIGURE 3. Orthogonality is not optimal all the time. The optimal quantization for data in the figure results in a non-orthogonal code matrix. In this case, the F-measure is maximal, i.e. equal to 1. It is impossible to find an orthogonal code matrix to achieve the same performance.

On the other hand, even an orthogonal hashing code matrix may have poor retrieval performance (Fig. 3). By directly maximizing F-measure, our method can avoid the impact from orthogonality as well as the correlation among hashing code bits.

IV. EXPERIMENTAL RESULTS

Our methods are evaluated on three benchmarks, CIFAR10, MIRFlickr and NUSWIDE. Two kinds of experiments were conducted, *hashing lookup* and *Hamming ranking*. The *hashing lookup* experiments are evaluated by F-measure, while the Hamming ranking experiments are evaluated by mean average precision (MAP).

A. DATASETS

CIFAR10: consists of 50,000 training images and 10,000 testing images. They are belonging to 10 classes. The ground-truth neighbors for a query are defined as those in the same category.

MIRFlickr: is comprised of 25,000 images each of which are annotated with at least one of 24 labels. 2,000 images are randomly selected as queries and the remaining 23,000 images are used as retrieval set. Ground-truth neighbors are defined as those sharing at least one label.

NUS-WIDE: contains 269,648 images. 81 concepts are provided for the entire dataset. 10 most common concepts are selected for labels. Hence, 186,577 images are left. 5% of the 186,577 images are used as queries and the remaining images are used as training set. Ground-truth neighbors for a query are defined as those sharing at least one label.

For all images, we use VGG16 [20] to extract 4,096-dimensional features for shallow hashing methods.

B. BASELINES

We incorporate five state-of-the-art traditional hashing methods to our scheme, i.e. ITQ [3], GHS [34], IMH [12], SGH [21] and SSDH [22]. ITQ, GHS and IMH are shallow hashing methods, while SGH and SSDH adopts deep neural networks. We evaluate the performance improvement brought by our scheme on these five methods.

C. EVALUATION

The *Hamming ranking* experiments are evaluated by MAP. The average precision (AP) is defined as:

$$AP = \frac{1}{n}\sum_{r=1}^{R} P(r)\delta(r), \tag{13}$$

where R is the radius of Hamming distance, P(r) is the precision of the top r retrieved images and $\delta(r) = 1$ if the r-th retrieved image is a true neighbor, otherwise $\delta(r) = 0$. MAP is the mean of APs for all queries. The maximum of MAP is 1 or 100%. The closer the MAP to 1, the better the performance.

The *hashing lookup* experiments are evaluated by F-measure. The Hamming radius is set to 2 for all experiments. That is, the hashing codes whose distances to a query are equal or less than 2 are retrieved to estimate F-measure.

D. RESULTS AND DISCUSSION

The MAP results are given in Table 1. In Table 1, we use $*$ to represent the proposed modification on original hashing methods. Results in Table 1 demonstrate that our scheme can improve the performance of the original hashing methods. Our scheme works better on deep hashing methods, i.e. SGH and SSDH. A possible explanation is that our method is equivalent to add a full-connected layer on the top of the deep neural networks. The ground-truth neighbors are determined by the labels and the information of ground-truth neighbors are incorporated in computing F-measure. That is, the label information is implicitly incorporated to the original unsupervised hashing methods. However, for shallow methods, even though we implicitly incorporate label information, the improvement on performance is relatively incremental because the features are extracted by the pre-trained VGG16. The label information are not used to fine-tune the weights of VGG16. For deep methods, the label information used to refining all the weights so that the performance are greatly improved.

The F-measure is shown in Fig. 4. It can be seen that the F-measure is greatly improved by the proposed scheme, especially on long-bit experiments. For traditional hashing methods, the main reason of low F-measure values in long-bit experiments is low recall values. As our objective function is F-measure, the F-measure declines slowly as the code bit increases.

E. ABLATION STUDIES

In this subsection, we tested the effects of α_1 and α_2 parameters in the PSO algorithm. It is conventionally to set α_1 and α_2 as 2. α_1 controls the weights of local searches, while α_2 controls the weights of global searches. For an extreme case, i.e. $\alpha_1 = 2$ and $\alpha = 0$, each particle becomes an

TABLE 1. MAP results on Wiki, MIRFlickr and NUS-WIDE datasets.

Methods	CIFAR10				MIRFlickr				NUS-WIDE			
	8 bits	16 bits	32 bits	64 bits	8 bits	16 bits	32 bits	64 bits	8 bits	16 bits	32 bits	64 bits
ITQ	0.1942	0.2086	0.2151	0.2188	0.6192	0.6318	0.6346	0.6477	0.5283	0.5323	0.5319	0.5424
ITQ*	**0.2205**	**0.2409**	**0.2518**	**0.2522**	**0.6334**	**0.6333**	**0.6399**	**0.6449**	**0.5542**	**0.5624**	**0.5762**	**0.4923**
GHS	0.2033	0.2145	0.2288	0.2296	0.6245	0.6430	0.6419	0.6531	0.5341	0.5396	0.5401	0.5523
GHS*	**0.2101**	**0.2178**	**0.2352**	**0.2364**	**0.6429**	**0.6610**	**0.6635**	**0.6706**	**0.5401**	**0.5507**	**0.5582**	**0.5670**
IMH	0.1639	0.1732	0.1780	0.1822	0.5756	0.5768	0.5825	0.5879	0.5138	0.5127	0.5210	0.5287
IMH*	**0.1967**	**0.2054**	**0.2214**	**0.2250**	**0.6173**	**0.6297**	**0.6305**	**0.6418**	**0.5266**	**0.5351**	**0.5413**	**0.5492**
SGH	0.1795	0.1827	0.1889	0.1904	0.6162	0.6283	0.6253	0.6206	0.4936	0.4829	0.4865	0.4975
SGH*	**0.2444**	**0.2453**	**0.2467**	**0.2495**	**0.6564**	**0.6656**	**0.6675**	**0.6690**	**0.5767**	**0.5852**	**0.5869**	**0.5847**
SSDH	0.2568	0.2560	0.2587	0.2601	0.6621	0.6733	0.6732	0.6771	0.6231	0.6294	0.6321	0.6485
SSDH*	**0.2856**	**0.2860**	**0.2864**	**0.2870**	**0.6922**	**0.6923**	**0.6918**	**0.6993**	**0.6731**	**0.6810**	**0.6817**	**0.6832**

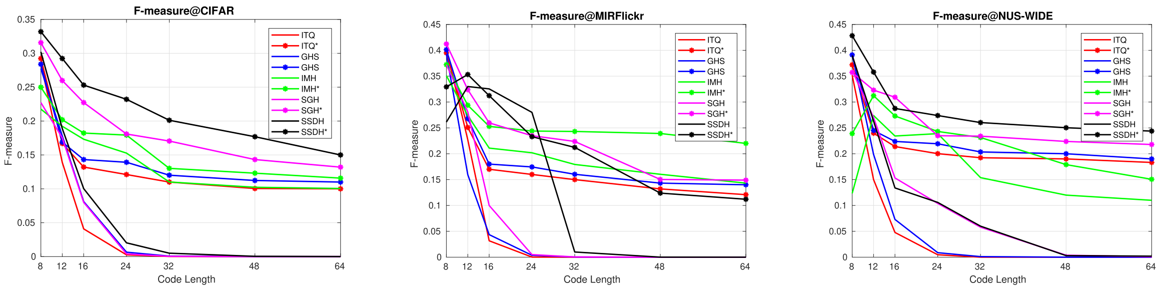

FIGURE 4. Results of Hash Lookup Experiments on CIFAR, MIRFlickr and NUS-WIDE datasets.

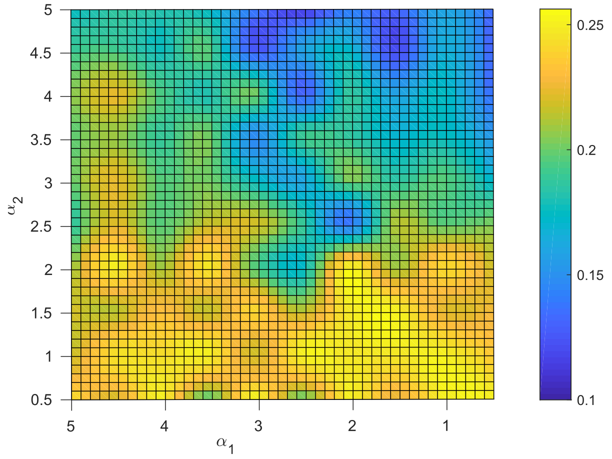

FIGURE 5. Average MAP results of ITQ* on CIFAR10.

independent searcher. On the other hand, when $\alpha_1 = 0$ and $\alpha_2 = 2$, all the particles will be attracted to one point and they will completely lose independence. We tested the $ITQ*$ on CIFAR10 dataset by setting α_1 and α_2 in a range from 0.5 to 5 by step 0.5. For each setting, we did hamming ranking experiments for 20 times to calculate the mean of MAP. The final MAP results are interpolated by cubic spline to look smoother. The results are shown in Fig. 5. It is safe to increase α_1. However, when increasing α_2, the MAP decreases dramatically. The reason is that a larger α_2 leads to higher dependence among particles so that they tend to move to one point and this point certainly is not the best one. PSO itself relies on multiple particles. Even though these particles are completely independent searchers, PSO still can choose the best result from these searchers. However, if all particles move to one point, PSO will have no choice.

V. CONCLUSION

In this paper, we proposed a scheme to further improve the performance of traditional hashing methods by directly maximizing F-measure. Particle Swarm Optimization (PSO) is used as an exemplary algorithm to maximize the non-differentiable objective. The proposed scheme can incorporate a wide range of hashing methods, heuristic optimization algorithms and non-differentiable metrics. Experimental results on three widely used benchmarks demonstrated that our scheme could further improve the performance of traditional hashing methods.

REFERENCES

[1] N. Brukhim and A. Globerson, “Predict and constrain: Modeling cardinality in deep structured prediction,” in *Proc. 35th Int. Conf. Mach. Learn.*, in Proceedings of Machine Learning Research, vol. 80. J. Dy and A. Krause, Eds. Stockholm Sweden: PMLR, Jul. 2018, pp. 659–667.

[2] J. Grabocka, R. Scholz, and L. Schmidt-Thieme, “Learning surrogate losses,” Tech. Rep., 2019.

[3] Y. Gong and S. Lazebnik, “Iterative quantization: A procrustean approach to learning binary codes,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 817–824.

[4] A. Andoni and P. Indyk, “Near-optimal hashing algorithms for approximate nearest neighbor in high dimensions,” *Commun. ACM*, vol. 51, no. 1, pp. 117–122, Jan. 2008.

[5] B. Kulis and K. Grauman, “Kernelized locality-sensitive hashing,” *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 6, pp. 1092–1104, Jun. 2012.

[6] M. S. Charikar, “Similarity estimation techniques from rounding algorithms,” in *Proc. 34th Annu. ACM Symp. Theory Comput. (STOC)*, 2002, pp. 380–388.

[7] Y. Weiss, A. Torralba, and R. Fergus, “Spectral hashing,” in *Proc. Adv. Neural Inf. Process. Syst.*, 2008, pp. 1753–1760.

[8] W. Liu, J. Wang, and S.-F. Chang, "Hashing with graphs," in *Proc. Int. Conf. Mach. Learn.*, 2011.

[9] W. Kong and W.-J. Li, "Isotropic hashing," in *Proc. Adv. Neural Inf. Process. Syst.*, 2012, pp. 1646–1654.

[10] C. Strecha, A. M. Bronstein, M. M. Bronstein, and P. Fua, "LDAHash: Improved matching with smaller descriptors," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 1, pp. 66–78, Jan. 2012.

[11] J. Tang, Z. Li, M. Wang, and R. Zhao, "Neighborhood discriminant hashing for large-scale image retrieval," *IEEE Trans. Image Process.*, vol. 24, no. 9, pp. 2827–2840, Sep. 2015.

[12] F. Shen, C. Shen, Q. Shi, A. van den Hengel, and Z. Tang, "Inductive hashing on manifolds," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2013, pp. 1562–1569.

[13] F. Shen, C. Shen, Q. Shi, A. van den Hengel, Z. Tang, and H. T. Shen, "Hashing on nonlinear manifolds," *IEEE Trans. Image Process.*, vol. 24, no. 6, pp. 1839–1851, Jun. 2015.

[14] D. Tian, Y. Wei, and D. Zhou, "Learning decorrelated hashing codes with label relaxation for multimodal retrieval," *IEEE Access*, vol. 8, pp. 79260–79272, 2020.

[15] G. Ding, Y. Guo, and J. Zhou, "Collective matrix factorization hashing for multimodal data," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2014, pp. 2083–2090.

[16] X. Lu, X. Zheng, and X. Li, "Latent semantic minimal hashing for image retrieval," *IEEE Trans. Image Process.*, vol. 26, no. 1, pp. 355–368, Jan. 2017.

[17] H. Liu, X. Li, S. Zhang, and Q. Tian, "Adaptive hashing with sparse matrix factorization," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 10, pp. 4318–4329, Oct. 2020.

[18] J. T. Zhou, H. Zhao, X. Peng, M. Fang, Z. Qin, and R. S. M. Goh, "Transfer hashing: From shallow to deep," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 29, no. 12, pp. 6191–6201, Dec. 2018.

[19] K. Lin, J. Lu, C.-S. Chen, and J. Zhou, "Learning compact binary descriptors with unsupervised deep neural networks," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2016, pp. 1183–1192.

[20] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," Tech. Rep., 2014.

[21] B. Dai, R. Guo, S. Kumar, N. He, and L. Song, "Stochastic generative hashing," in *Proc. 34th Int. Conf. Mach. Learn. (ICML)*, vol. 70, 2017, pp. 913–922.

[22] E. Yang, C. Deng, T. Liu, W. Liu, and D. Tao, "Semantic structure-based unsupervised deep hashing," in *Proc. 27th Int. Joint Conf. Artif. Intell.*, AAAI Press, Jul. 2018, pp. 1064–1070.

[23] E. Yang, T. Liu, C. Deng, W. Liu, and D. Tao, "Distillhash: Unsupervised deep hashing by distilling data pairs," Tech. Rep., 2019.

[24] X. Lu, Y. Chen, and X. Li, "Discrete deep hashing with ranking optimization for image retrieval," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 31, no. 6, pp. 2052–2063, Jun. 2020.

[25] Y. Shen, J. Qin, J. Chen, M. Yu, L. Liu, F. Zhu, F. Shen, and L. Shao, "Auto-encoding twin-bottleneck hashing," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit. (CVPR)*, Jun. 2020, pp. 2815–2824.

[26] N. Ye, K. M. Chai, W. S. Lee, and H. L. Chieu, "Optimizing f-measures: A tale of two approaches," in *Proc. 29th Int. Conf. Mach. Learn.*, 2012, pp. 289–296.

[27] Z. C. Lipton, C. Elkan, and B. Naryanaswamy, "Optimal thresholding of classifiers to maximize F1 measure," in *Machine Learning and Knowledge Discovery in Databases*, T. Calders, F. Esposito, E. Hüllermeier, and R. Meo, Eds. Berlin, Germany: Springer, 2014, pp. 225–239.

[28] N. Natarajan, O. Koyejo, P. Ravikumar, and I. S. Dhillon, "Consistent binary classification with generalized performance metrics," in *Proc. Neural Inf. Process. Syst. (NIPS)*, Dec. 2014, pp. 2744–2752.

[29] S. A. P. Parambath, N. Usunier, and Y. Grandvalet, "Optimizing F-measures by cost-sensitive classification," in *Proc. 27th Int. Conf. Neural Inf. Process. Syst.*, vol. 2, 2014, pp. 2123–2131.

[30] K. Bascol, R. Emonet, E. Fromont, A. Habrard, G. Metzler, and M. Sebban, "From cost-sensitive classification to tight F-measure bounds," in *Proc. 22nd Int. Conf. Artif. Intell. Statist. (AISTATS)*, vol. 89, no. 1, Naha, Japan, Apr. 2019, pp. 1245–1253.

[31] M. Gygli, M. Norouzi, and A. Angelova, "Deep value networks learn to evaluate and iteratively refine structured outputs," in *Proc. 34th Int. Conf. Mach. Learn.*, vol. 70, 2017, pp. 1341–1351.

[32] E. Eban, M. Schain, A. Mackey, A. Gordon, R. Rifkin, and G. Elidan, "Scalable learning of non-decomposable objectives," in *Proc. Mach. Learn. Res.*, vol. 54, A. Singh and J. Zhu, Eds. Fort Lauderdale, FL, USA: PMLR, Apr. 2017, pp. 832–840.

[33] M. Berman, A. R. Triki, and M. B. Blaschko, "The lovasz-softmax loss: A tractable surrogate for the optimization of the intersection-over-union measure in neural networks," in *Proc. IEEE/CVF Conf. Comput. Vis. Pattern Recognit.*, Jun. 2018, pp. 4413–4421.

[34] D. Tian and D. Tao, "Global hashing system for fast image search," *IEEE Trans. Image Process.*, vol. 26, no. 1, pp. 79–89, Jan. 2017.

YIWEN WEI received the Ph.D. degree in radio science from the School of Physics and Optoelectronic Engineering, Xidian University, Xi'an, China, in 2016. From 2016 to 2018, she worked as a Research Scientist with the Temasek Laboratories, National University of Singapore, Singapore. She is currently an Assistant Professor with the School of Physics and Optoelectronic Engineering Science, Xidian University. Her research interests include electromagnetic wave propagation and scattering in complex systems, computational electromagnetic, remote sensing, parameters retrieval, in particular applying machine learning methods on complex electromagnetic problems.

DAYONG TIAN received the B.S. and M.E. degrees from Xidian University, Xi'an, China, in 2010 and 2014, respectively, and the Ph.D. degree from the University of Technology Sydney, Ultimo, NSW, Australia, in 2017. He is currently an Assistant Professor with the School of Electronics and Information, Northwestern Polytechnical University, Xi'an. His research interests include computer vision and machine learning, in particular on image restoration, image retrieval, and face recognition.

JIAO SHI (Member, IEEE) received the B.Eng. and Ph.D. degrees from the School of Electronic Engineering, Xidian University, Xi'an, China, in 2009 and 2015, respectively. From August 2013 to August 2014, she was a Visiting Scholar with the Leiden Institute of Advanced Computer Science, Leiden University, Leiden, The Netherlands. In 2015, she was a Teacher with the School of Electronics and Information, Northwestern Polytechnical University, Xi'an. From November 2016 to December 2016, she was a Visiting Scholar with the College of Information Technology, Incheon National University, Incheon, South Korea. In 2018, she was promoted to an Associate Professor. From September 2019 to October 2019, she was a Visiting Scholar with the College of Information Technology, Incheon National University. She is currently an Associate Professor with the School of Electronics and Information, Northwestern Polytechnical University. Her research interests include computational intelligence and remote sensing image processing.

YU LEI (Member, IEEE) received the B.Eng. and Ph.D. degrees from the School of Electronic Engineering, Xidian University, Xi'an, China, in 2009 and 2015, respectively. In 2015, he was a Teacher with the School of Electronics and Information, Northwestern Polytechnical University, Xi'an. From November 2016 to December 2016 and from September 2019 to October 2019, he was a Visiting Scholar with the College of Information Technology, Incheon National University, Incheon, South Korea. In 2019, he was promoted to an Associate Professor. He is currently an Associate Professor with the School of Electronics and Information, Northwestern Polytechnical University. His research interests include computational intelligence and remote sensing image processing.