World Wide Web
https://doi.org/10.1007/s11280-020-00779-x

A framework for image dark data assessment

Ke Zhou, et al. *[full author details at the end of the article]*

Received: 29 August 2019 / Revised: 27 November 2019 / Accepted: 2 January 2020
Published online: 29 February 2020

Abstract
Image dark data, whose content and value are not clear, consistently occupy the storage space but hardly produce great value. Blindly applying data mining techniques on these data is highly likely to bring disappointed result and waste large resource. Therefore, it is of great significance to assess the dark data before data mining to help the user cognize the data. However, there are several challenges in dark data assessment work. First, the similarity between images must be objectively measured under aunified standard to help the user understand the evaluation values of dark data. Second, it is important to capture semantic features with generalization ability. Third, it is challenging to design an efficient assessment scheme to support large-scale datasets. To overcome these challenges, we propose an assessment framework which includes offline calculation and online assessment. In offline calculation, we first transform unlabeled images into hash codes by our developed Deep Self-taught Hashing (DSTH) algorithm which can extract semantic features with generalization ability, then construct a semantic graph using restricted Hamming distance, and finally use our designed Semantic Hash Ranking (SHR) algorithm to calculate the overall importance score (rank) for each node (image), which takes both the number of connected links and the weight on edges into consideration. During online assessment, we first translate the user's query (semantic images) into hash codes using DSTH model, then match the data contained in the dark data via a predefined Hamming distance query range, and finally return the weighted average value of these matched data to help the user cognize the dark data. The results on real-world dataset show our framework can apply to large-scale datasets, help users evaluate the dark data by different requirements, and assist the user to conduct subsequent data mining work.

Keywords Image dark data · Assessment · DSTH · SHR

This article belongs to the Topical Collection: *Special Issue on Web and Big Data 2019*
Guest Editors: Jie Shao, Man Lung Yiu, and Toyoda Masashi

1 Introduction

Dark data is defined as the information assets that can be easily collected and stored, but generally fail to use for data analytics and mining.[1] Image dark data are ubiquitous and have brought economic costs to enterprises. For example, many social platforms store image data (i.e., albums and chat images) as an independent resource separated from other businesses. These massive image data quickly turn into dark data, which contain lots of historical records and thus are not allowed to be removed. However, they consistently occupy the storage space but can not produce greater value. Therefore, developers are eager to mine image dark data in order to improve the cost performance of storage [38]. However, owing that the image dark data lack labels and associations, owners have no idea how to apply these data. For a given target, blindly conducting data mining techniques on the dark data is highly likely to cause bad results and waste of resources. For example, as shown in Figure 1, we almost waste all the mining resource when searching images about dog head on the dark dataset. Faced with image dark data whose content and value are not clear, the primary issue is to judge whether this dataset are worth mining or not. Therefore, it is of great significance to evaluate the value of image dark data and guide users to know about the potential value of these data.

Given this, which way shall be taken for the assessment and what result shall be fed back to make the user aware of the dark data? Faced with the user's query, there exist many challenges when executing association analysis on dark data.

(1) **Requirement semantic expression.** The user's requirement, as an idea, is usually abstract and difficult to understand by computer, although it can be expressed in words. Especially for image data, directly using image as input is simpler and more intuitive. For example, if the user need pictures depicting dog, he can just input an image about dog. Thus, it puts forward a higher request on extracting content semantic of images.

(2) **Semantic information extraction.** Reasonable semantic extraction method is the key to correctly understand the user's requirement and perceive contents of dark dataset. It remains to be a great challenge to design suitable semantic labels to extract the semantic information of unlabeled dark data, even though deep learning seems to be a feasible scheme. Moreover, the label semantics of any model are limited, so it will inevitably cause a huge semantic bias if directly using existing model trained on other dataset [32, 33]. This so called out-of-sample problem will bring woeful results in deep model.

(3) **Semantic feature with generalization ability.** Deep model suffers from a poor generalization ability when extracting semantic information. Since almost semantic extraction models are based on classification, the similar semantic images identified by labels own a shorter distances between each other. Meanwhile, images with different semantics are given longer distances between them. An excellent classification model will obtain desired classification effects by averaging the distances between different classes as much as possible. However, our goal is not to classify the data, but to get semantic features with generalization ability when acquiring the cognition of a dataset. For example, when training semantic model, we shall make that the semantic feature of a cat similar to that of a dog but different from that of an airplane.

(4) **Similarity and extent of relevance.** If we intend to manifest relevant data, what kind of value (threshold) can be used to defined as similarity between the dark data and the

[1] https://www.gartner.com/it-glossary/dark-data/

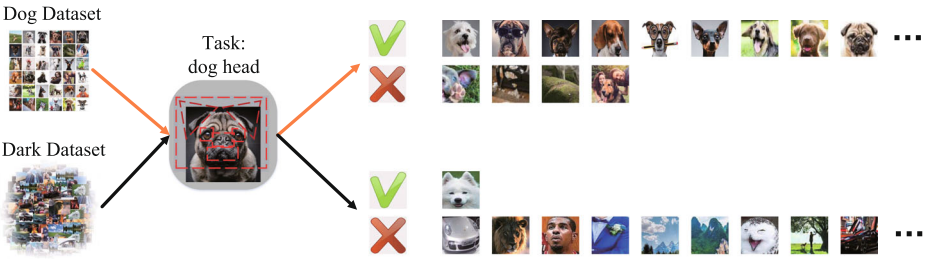


Figure 1 Satisfying result can be achieved on the dog dataset if the task is to search images of dog head. However, we got an awful feedback with only one expected image from the dark dataset for the same task and consumed all the mining resource. This illustrates that directly investing the mining resource on a dark dataset without judgment or assessment whether the dataset are worth mining may cause disappointed result and waste resource

requirement? Measuring the vectors used to express semantic features seems not hard (i.e., Euclidean distance and Cosine distance), but it becomes meaningless to directly return these distances to the user. First, it is impractical to return a large number of distances to the user for judgment on a million scale dataset. Second, even if all the distances are counted, it also seems tricky to objectively give a convincing threshold. Finally, the extent of relevance can hardly be expressed for any given threshold, especially for the heterogeneous goals. For the assessment task, we need to give an objective assessment value under a unified standard from an overall perspective which contains both the number of relevant data and the extent of relevance [39].

(5) **Evaluation standard.** Setting an objective evaluation standard from an overall perspective is also a challenge. One of the most common means is clustering. However, whether the method is based on the number of hypothetical centers [16] or density [20], it needs many iterations and will take a long time to complete the clustering. This may cause an unacceptable cost for potentially changeable dark datasets. Moreover, the clustering results are represented by multiple centers, but one of the original purposes of our assessment work is to get all the semantic relevant data which are similar to the given query on the whole dataset instead of several centers. Also, quantization [31] uses the concept of codebook to specify the evaluation standard in clustering, but codebook is only suitable for encoding data and thus fails to give the overall assessment. In addition to the clustering methods, graph-based computing [5] is another way to achieve global evaluation. The most well-known one is PageRank algorithm [23], which determines the importance of Web pages according to the links. However, PageRank can only express directional attributes on a directed graph, so it fail to measure the mutual extent of relevance between objects. Events detecting [2] can find the hot events though connections on an undirected graph, but the representative data are very limited. Once the query data can not match any hot event, no assessment result will be returned, so it does not apply to our assessment task.

(6) **Online query cost.** Even if the evaluation standard is given, we still need to find the corresponding related data in the whole dataset to measure the feedback of the query. Online computing millions of high-dimensional floating-point vectors means a huge resource consumption. Besides, the assessment task will receive frequent query requests for different requirements. Thus, the assessment work is supposed to be built on more efficient distance measurement for practical feasibility.

In this paper, we propose an assessment framework for image dark data assessment by combining deep learning, hash technique and graph-based computing. Note that there lacks assessment work for dark data, and our designed framework is the first attempt to assess the value of image dark data as well as quantify the assessment process for the given requirement of the user. The framework consists of four parts. First, we use deep self-taught hashing (DSTH) algorithm to transform unlabeled images into deep semantic hash codes. Note that in the model generating stage of DSTH, we combine the clustering method which has the ability to perceive features. This makes those images which look more similar own closer hash codes and thus our model has stronger generalization ability. Second, we built the semantic undirected graph using restricted Hamming distance. Note that Hamming distance can not only speed up the construction of graph but also simplify the measured distance on edge owing to the easy but fast "XOR" operation. Besides, according to what DCH [3] describes, we cut off those unreasonable connections and improve the efficiency of construction and subsequent calculation on graph. Third, on the built graph, we design semantic hash ranking (SHR) algorithm to calculate the importance score for each node by random walk and obtain the rank for each image. It is worth mentioning that we improve the PageRank algorithm and extend it to undirected weighted graph, which takes both the number of connected links and the weight on edges into consideration. For a given query, conventional methods may return all those images within a certain range, while we give an intuitive score (rank) assessment feedback. At last, according to the user's input, we match the corresponding data contained in the dataset which are restricted within a given Hamming distance range, calculate the weighted semantic importance score of these data, and return the rank of this input. The user can decide whether conducting data mining on this dark dataset based on the returned rank of the input. The major contributions of this paper are summarized as follows:

- We design a deep self-taught hashing (DSTH) algorithm, which can extract semantic features without labels and solve the out-of-sample problem.
- Based on the built semantic graph, we propose a semantic hash ranking (SHR) algorithm to calculate the overall importance score for each node (image) according to random walk, which takes both the number of connected links and the weight on edges into consideration.
- We propose a calculation-query-assessment framework consisting of offline calculation and online assessment, which applies to assessing large-scale datasets.
- Our framework can help users to detect the potential value of the dark data and avoid unnecessary mining cost and contributes to data application. To the best of our knowledge, this is the first attempt that assesses image dark data.

2 Design overview

This section first formulates the problem and then presents our framework to address this problem.

2.1 Problem formulation

Given an image dark dataset with a set of images, and a query image (or a query with multiple images), we want to (1) find the matched images corresponding to the query; and (2) return a ranking score which reflects the relevance of the query to the dataset.

For case (1), we need to define whether two images are matched. To this end, we compute a hash code for each data image, and two images are matched (or similar) if their Hamming distance is not larger than a given threshold. Formally, assuming that h_i and h_j respectively represent the hash codes of two images, we use $HD(h_i, h_j)$ to denote the Hamming distance between h_i and h_j, and hd to denote a given matching threshold. If $HD(h_i, h_j) \leq hd$, the two images are matched.

For case (2), we evaluate whether there are enough matched images to the query. To this end, we rank the images based on their semantics, and denote the ranked scores as $\{S_1, S_2, \ldots, S_n\} (\forall k, S_{k-1} \geq S_k)$. Then given a query image, we find all the matched images and calculate the average weighted score of all these matched images, and denote this score as $S(q)$. If $S_{k-1} > S(q) \geq S_k$, we then return the ratio $T(q) = 1 - \frac{k}{N}$ as the score, which shows how much this query is related to the dataset. Obviously, the larger the score is, the higher relevance of the query to the dataset.

2.2 Framework overview

For a large-scale image dark dataset, in order to make our assessment framework effective for real-time analysis, we need to perform offline analysis on the dataset to get the score list (rank of each image). Then given an online application query, we evaluate whether the dataset can be used for the query on-the-fly. As shown in Figure 2, the framework consists of four steps. The first three steps give an offline evaluation on the dark dataset, which calculates the importance scorefor each image. The last step provides a suggestion according to an online matching and weighted computing which returns the sequence number based on the score list with computed score of query.

Offline evaluation We design three steps to effectively calculate the semantic importance score and provide each image with a rank. Formally, we first train a Deep Self-taught Hashing (DSTH) model and transform all dark data into hash codes, then build a semantic undirected graph with restricted Hamming distance, and finally calculate the overall importance score (rank) for each image by our designed Semantic Hash Ranking (SHR) algorithm.

(1) *Step 1: hash function learning and dark data mapping.* As shown in the first frame of Figure 3, we adopt the DSTH algorithm to encode each image of the dark dataset. The DSTH algorithm contains two stages: hash label generating stage and hash function training stage. First, it is important to acquire hash labels, because the premise of feature extraction using deep learning is based on semantic labels. Owing that the entity classes of the dark dataset are not clear, it is better to select the training dataset with as many classes as possible and then transform original classification labels into hash labels. On the one hand, we choose ImageNet and the same amount of sampled image dark data as the training data. On the other hand, we choose GoogLeNet trained on ImageNet to

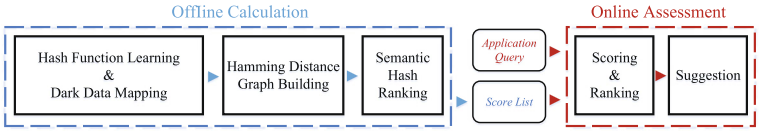


Figure 2 The framework for image dark data assessment

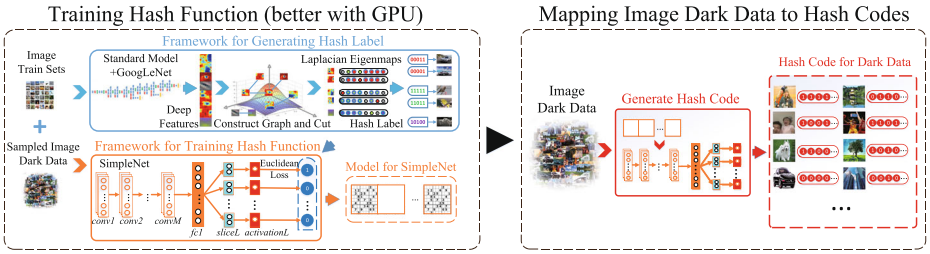


Figure 3 The process of hash function learning and dark data mapping

extract semantic features of these data. Next, we use the features to construct a graph via K-NN ($K = 12$), then map data to predefined l-dimensional space by means of Laplacian eigenvalue decomposition, and finally binarize all data to generate hash label. We conduct clustering on extracted semantic features, which not only preserves original semantic classification information but also makes these semantics automatically closer or estranged according the similarity between themselves. Those labels have the semantics with generalization ability, which directly affects the next hash function learning. Note that the hash function is specially trained on above sampled dark data. Our generalized feature extraction method (DSTH) converts high-dimensional dark data into low-dimensional hash vectors that can be easily but fast measured. The mathematical expression of DSTH and the advantages are described in Section 3. At last, as shown in the last frame of Figure 3, according to the obtained deep hash functions, we map each image of the dark dataset into a hash code which represents the semantic feature of the data.

(2) *xStep 2: graph building with Hamming distance.* As shown in the first frame of Figure 4, we model the images as a graph G where each node is an image and edges are relationships between images. In order to speed up the graph construction, we cut off those edges on which the weight exceeds half of the length of hash code, according to the conclusion of Long [3]. Let N_* denote the $*$-th node of G, $H(N_*)$ denote hash code of N_* and l denote length of hash codes. We define XOR operation as $\oplus$. Therefore, the Hamming distance weight on the undirected link between N_i and N_j can be defined as

$$d_{ij} = \begin{cases} H(N_i) \oplus H(N_j) & i \neq j, H(N_i) \oplus H(N_j) \leq \Omega, \\ NULL & otherwise. \end{cases} \tag{1}$$

where $\Omega = \lceil l/s \rceil$ and $s \in [1, l]$. In practice, the determination of Ω is based on efficiency of building a graph with tolerable loss. Formally, we define the precision of i-th node as C_i/L_i, where L_i represents the number of all nodes connected to i-th node and there exist C_i nodes of the L_i nodes that have the same label as the i-th node. Therefore, the precision of graph $P(G|\Omega)$ is defined as

$$P(G|\Omega) = \frac{1}{N} \sum_{i=1}^{N} \frac{C_i}{L_i} \tag{2}$$

(3) *Step 3: Semantic Hash Ranking.* As shown in the middle frame of Figure 4, after building the graph with restricted Hamming distance, we calculate the importance score for each node by random walk in order to obtain the overall objective evaluation value. We extend the PageRank algorithm and propose the SHR algorithm which takes both the number of connected links and the weight on edges into consideration. Note that we specially design how to reasonably calculate the extent of relevance between nodes, aiming at making full use of the Hamming distance of similarity hash. On the built semantic graph, we use SHR to calculate the importance score for each node. At the same time, according to the physical meaning of Hamming distance, we redesign the iteration matrix elements for obtaining reasonable importance scores. SHR can make full use of the shortened Hamming distance between hash codes with generalization ability, which makes the dominant semantics more prominent, thus reinforcing the user's cognition of the dark dataset and acquiring the score list shown in the last frame of Figure 4. We introduce the detailed calculation process of SHR in Section 4.

Online query assessment As shown in Figure 5, for the image dark data consisting of N images, the query will be mapped to hash codes by hash function calculated in Section 3 and associated with images contained in the dark data. The matching range is defined as hd and we set $hd = 1$ to conduct matching. Mathematically, we let q denote a query with n images, img_i denote the i-th image where $i \in [1, n]$, m_i denote the number of matched images for the i-th image of the query q. Meanwhile, we let $S_j(img_i)$ denote the score of the j-th image where $j \in [1, m_i]$. Therefore, the score of q is defined as follows:

$$S(q) = \sum_{i=1}^{n} \frac{1}{m_i} \sum_{j=1}^{m_i} \beta_i S_j(img_i) \quad s.t. \sum_{i=1}^{n} \beta_i = 1 \tag{3}$$

where $\beta_i \in [0, 1]$ represents the importance weight of the i-th image. Note that the value of β_i is determined by the user. If the user cares more about the i-th image, he can set a relatively larger β_i (as shown in Figure 13).

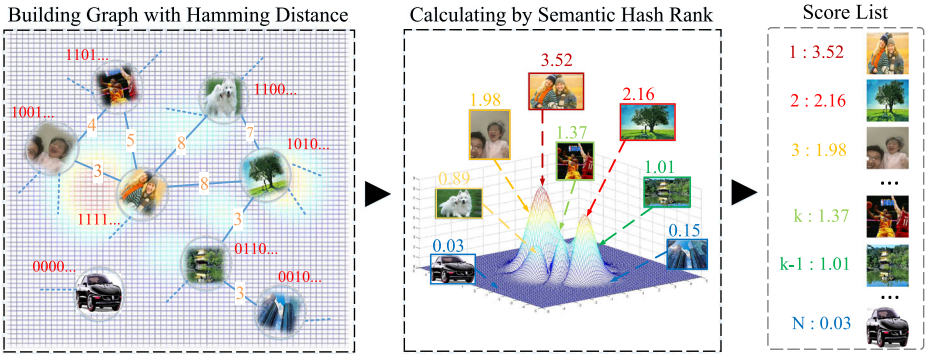


Figure 4 The process of graph building and semantic hash ranking

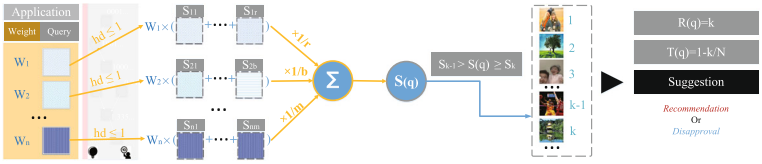


Figure 5 The process of assessment

Compared with the ranked scores denoted as $\{S_1, S_2, \ldots, S_N\}$ of image dark data calculated by SHR, we can acquire the sequence number of $S(q)$ denoted as k in the score list, where $S_{k-1} > S(q) \geq S_k$. Further, $T(q) = 1 - \frac{k}{N}$ represents importance degree of image dark data for the query. As results, we will give a suggestion according to $T(q)$ and the user can decide whether the image dark data are worth mining for the query (application).

3 Deep self-taught hashing (DSTH)

In this section, we detailedly describe DSTH algorithm including how to integrate clustering information into semantic learning under deep learning framework, how to generate hash label, and how to conduct the training process. And then, we elaborate on the advantages of DSTH.

3.1 DSTH algorithm

The algorithm mainly contains hash label generating stage and hash function training stage.

(1) *Hash label generating stage.* The prime task of DSTH is to acquire semantic labels, because labels determine which semantic informations should be extracted from data and directly affect the subsequent function learning. Semantic labels acquiring aims at extracting semantic information and semantic feature with generalization ability (mentioned in Section 1). Supervised deep learning algorithm is able to better extract semantic information owing to the accurate hand-crafted labels denoted as red line in Figure 6. Unsupervised shallow learning algorithm extracts semantics according to the similarity between data themselves, which applies to those scenes without hand-crafted labels denoted as blue line in Figure 6. Our method combines these two advantages, which can not only obtain semantic information without labels but also reach the balance between human semantic cognition and data semantic cognition, so as to acquire our expected semantic features (labels).

We apply the deep and shallow mixed learning method, which integrates clustering information and improves the generalization ability of feature extraction. In the absence of labels, we use the trained deep model to get features. After that, we use Laplacian Eigenvalue and binarization to transform the extracted deep features to hash codes which serve as hash labels for next stage. Mathematically, we use n m-dimensional vectors $\{x_i\}_{i=1}^{n} \in \mathbb{R}^m$ to denote the image features and use $ED(i,j) = \left\|x_i - x_j\right\|_2^2$ to denote the Euclidean distance between i-th and j-th image. For θ_t $(t \in [1, n-1])$, we denote $\{ED(i, \theta_1), ED(i, \theta_2), \ldots, ED(i, \theta_{n-}$

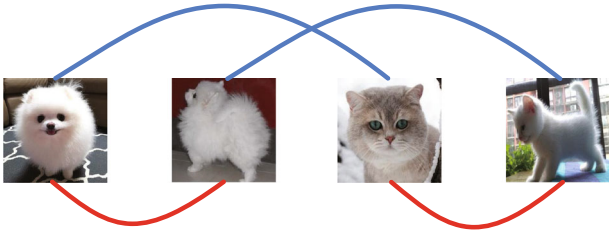


Figure 6 Red line represents the deep semantic classification result based on classification labels, while blue line represents the semantic result by unsupervised shallow method. The deep method can distinguish different classes, but fails to capturethe semantics by structure information of pixels and lose generalization ability, because strictly limited by classification labels. Conversely, the shallow method can just classify images by pixels

$_1)\} = \{ED(i,1), ED(i,2), \ldots, ED(i,i-1), ED(i,i+1), \ldots, ED(i,n)\}$, where $ED(i,\theta_t) \leq ED(i,\theta_{t+1})$. We define that $TK_{ED}(i,j) = t$, if $ED(i,j) = ED(i,\theta_t)$. Further, we use $N_K(i,j)$ to denote neighbor relationship between i-th and j-th data, which is defined as

$$N_K(i,j) = \begin{cases} True & TK_{ED}(i,j) \leq K, \\ False & TK_{ED}(i,j) > K. \end{cases} \tag{4}$$

Next, we use x_i and y_i to represent the i-th sample and its hash codes where $y_i \in \{0,1\}^\gamma$ and γ denotes the length of hash codes. We set $y_i^\rho \in \{0,1\}$ as the ρ-th element of y_i. The hash codeset for n samples can be represented as $[y_1, \ldots, y_n]^T$. Our $n \times n$ local similarity matrix W is

$$W_{ij} = \begin{cases} 0 & \text{if } N_K(i,j) \text{ is False,} \\ \dfrac{x_i^T x_j}{\|x_i\| \cdot \|x_j\|} & \text{otherwise.} \end{cases} \tag{5}$$

Furthermore, we use W_{ij} to obtain the diagonal matrix

$$D_{ii} = \sum_{j=1}^{n} W_{ij} \tag{6}$$

Meanwhile, we use the number of different bits for calculating Hamming distance between y_i and y_j as

$$H_{ij} = \|y_i - y_j\|^2 / 4 \tag{7}$$

We define an object function ζ to minimize the weighted average Hamming distance.

$$\zeta = \sum_{i=1}^{n} \sum_{j=1}^{n} W_{ij} H_{ij} \tag{8}$$

To calculate ζ, we transform it to $\xi = tr(Y^T L Y)/4$, where $L = D - W$ is Laplacian matrix and $tr(\cdot)$ means trace of matrix. At last, we transform ξ to LapEig problem ψ with slacking constraint $y_i \in \{0,1\}^t$, and obtain the optimal t-dimensional real-valued vector $\widetilde{y}$ to represent each sample. ψ is the following:

$$\psi = arg \min_{\widetilde{Y}} \; Tr\left(\widetilde{Y}^T L \widetilde{Y}\right) \; s.t. \begin{cases} \widetilde{Y}^T D \widetilde{Y} = I \\ \widetilde{Y}^T D 1 = 0 \end{cases} \tag{9}$$

where $Tr\left(\widetilde{Y}^T L\widetilde{Y}\right)$ gives the real relaxation of the weighted average Hamming distance $Tr(T^T LY)$. The solution of this optimization problem is given by $\widetilde{Y} = [v_1, \ldots, v_t]$ whose columns are the t eigenvectors corresponding to the smallest eigenvalues of following generalized eigenvalue problem. The solution of ψ can be transformed to

$$Lv = \lambda Dv \tag{10}$$

where vector v is the t eigenvectors which are corresponding to the t smallest eigenvalues (nonzero).

Then, we convert the t-dimensional real-valued vectors $\widetilde{y}_1, \ldots, \widetilde{y}_n$ into binary codes according to the threshold. We set δ^p to present threshold and $\widetilde{y}_i^p$ equivalent to p-th element of $\widetilde{y}_i$. The hash label as final result value of y_i^p is

$$y_i^p = \begin{cases} 1 & \widetilde{y}_i^p \geq \delta^p, \\ 0 & otherwise. \end{cases} \tag{11}$$

where

$$\delta^p = \frac{1}{n} \sum_{i=1}^{n} \widetilde{y}_i^p \tag{12}$$

Note that we refer to STH [40] to calculate δ^p. If we directly use sign function to generate the hash value, most images will obtain indistinguishable hash codes with a high possibility.

(2) *Hash model training stage.* We implement an end-to-end hashing deep learning module. Firstly, we employ CNNs again to receive fine-grained features. After that, we adopt encoding module which is Divide and Encode Module [12] associated with activation function of BatchNorm [9] to approximate hash labels generated in previous stage. The learning framework is the artificial neural network on the multi-output condition. Formally, we set a function $f: \mathbb{R}^I \rightarrow \mathbb{R}^O$, where I is the input set, O is the output set and x is the input vector. The formulation is

$$\begin{aligned} f^{(1)}(x) &= b^{(2)} + W^{(2)} h\left(b^{(1)} + W^{(1)} x\right) \\ f^{(2)}(x) &= b^{(4)} + W^{(4)} h\left(b^{(3)} + W^{(3)} f^{(1)}(x)\right) \\ &\cdots \\ f^{(n)}(x) &= b^{(2\times n)} + W^{(2\times n)} h\left(b^{(2\times n-1)} + W^{(2\times n-1)} f^{(n-1)}(x)\right) \end{aligned} \tag{13}$$

where b is bias vector, W is weight matrix of convolution and $h(*)$ is ReLU and BatchNorm function. When the core of $h(x)$ is BatchNorm, the function is calculated as follows:

$$\widetilde{x}^{(k)} = \frac{x^{(k)} - E\left(x^{(k)}\right)}{\sqrt{Var\left(x^{(k)}\right)}} \tag{14}$$

where

$$E(x) = \frac{1}{m} \sum_{i=1}^{m} x_i \tag{15}$$

$$Var(x) = \frac{1}{m} \sum_{i=1}^{m} (x_i - E(x))^2 \tag{16}$$

In the last layer of CNN, we split a 1024-dimensional vector into 16 groups, and each group is mapped to q elements. The output number $16 \times q$ is the hash code length. Denote the output as one $m \times d$ matrix (m is the number of samples in batch and d is the number of output in last full connection layer), x is the output vector, y is the corresponding label. We define the loss function as follows:

$$F(x) = min \sum_{i=1}^{m} \sum_{j=1}^{d} \left\| x_i^{(j)} - y_i^{(j)} \right\|_2^2 \tag{17}$$

At last, we define the threshold function as the same as Eqs. (11) and (12). Usually, we apply the threshold values of each bit calculated in the hash label generating stage.

3.2 Advantages for dark data

The advantages of DSTH for dark data are summarized as follows. (1) Adaptability. For the image dark dataset without labels, DSTH can complete the feature extraction of the dataset in deep framework. (2) Features with generalization ability. We add the clustering process to the label acquisition in deep learning in order that the extracted features own generalization ability. It must be emphasized that the algorithm does not directly use the results of feature extraction for hash mapping as labels, because traditional deep models will average all the classification distances when implementing classification problems. However, we hope to get semantic results with generalization ability, which is shown in Figure 7. Therefore, through our means mentioned above, the changed semantic distance will be reflected in hash label, which directly affects the next hash function learning and makes the model consider both the artificial semantic classification and the distances between the data themselves. (3) Efficiency. DSTH can fast map images into features and hash codes. Using "XOR" operation to measure Hamming distance between images is easy but fast, which is suitable for large-scale scenes.

4 Semantic hash ranking (SHR)

In this section, we introduce SHR algorithm in detail, which considers both the number of connected links and the weight on edges into consideration, reasonably designs impact factor between different nodes according to similarity hash code, and calculate the importance score for each node by random walk. We also give a concise description of its convergence, its dynamic computational way and its advantages.

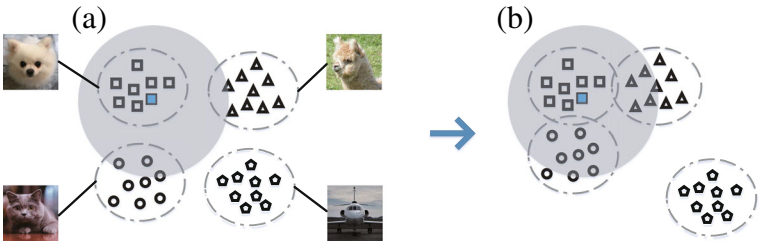


Figure 7 We respectively use the square, circle, triangle and pentagon to denote dog, cat, alpaca and airplane. The classification effects using deep learning without and with generalization ability are respectively reflected in part A and B. The bluedot represents a given center (image) and the shadow part represents the scope associated with the center. A can better solve the classification problem, while B is able to generalize and expose the semantic structure of the whole dataset. For those images which look similar even though they belong to different classes, generalized feature extraction method (DSTH) is able to associate them with each other, but A fails to accomplish this task. In addition, from an overall perspective, SHR can make full use of this kind of generalized semantic information and produce expected importance score for each image in the whole dataset

4.1 SHR algorithm

Let L_* denote number of links to N_*. Draw rank factor $R(N_*)$ for N_* and impact factor $I(N_{ij})$ for N_j to N_i, where $I(N_{ij})$ is defined as

$$I(N_{ij}) = \begin{cases} \frac{l-d_{ij}}{\sum_{t\in T_j} l-d_{tj}} R(N_j) & \exists d_{ij}, \\ 0 & otherwise. \end{cases} \tag{18}$$

where T_j is the set including orders of all nodes associated with N_j. Specially, we design the formulation according to two principals. Firstly, the less d_{ij} is, the greater influence N_j contributes to N_i is. Meanwhile, the longer hash code is, the more compact the similarity presented by d_{ij} is. Secondly, PageRank considers the weights on each edge as the same, but we extend it to be applied to different weights on edges. As a result, when weights on different edges are the same, Eq. (18) should be the same as the impact factor formulation of PageRank. Consequently, $R(N_i)$ should be equal to the sum of the impact factors of all nodes linked to N_i

$$R(N_i) = \sum_{j=1,j\neq i}^{n} I(N_{ij}) \tag{19}$$

Let f_{ij} represent the coefficient of $R(N_j)$ in $I(N_{ij})$. We draw iteration formula as

$$\begin{bmatrix} R^{c+1}(N_1) \\ R^{c+1}(N_2) \\ \cdots \\ R^{c+1}(N_n) \end{bmatrix} = \begin{bmatrix} 0 & f_{12} & \cdots & f_{1n} \\ f_{21} & 0 & \cdots & f_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ f_{n1} & f_{n2} & \cdots & 0 \end{bmatrix} \begin{bmatrix} R^{c}(N_1) \\ R^{c}(N_2) \\ \cdots \\ R^{c}(N_n) \end{bmatrix} \tag{20}$$

where c is the number of iteration rounds. We define the termination condition as

$$R^{c+1}(N_m) - R^{c}(N_m) \leq \varepsilon \tag{21}$$

where $m \in [1, n]$ and $\forall N_m$ should satisfy Eq. (21). Meanwhile, ε (1.0E-15, 1.0E-11, 1.0E-7 in our experiment) is constant. Let $SHR(N_*)$ denote semantic rank of N_*. The last results are

$$SHR(N_m) = R^{\eta}(N_m) \tag{22}$$

where η (65, 141 in our experiment) is the round on termination.

4.2 Convergence of SHR

In order to ensure our algorithm can converge to a stable result, we prove the convergence of Eq. (20), let A_n represent iteration coefficient matrix. The A_n is

$$A_n = \begin{bmatrix} 0 & f_{12} & \cdots & f_{1n} \\ f_{21} & 0 & \cdots & f_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ f_{n1} & f_{n2} & \cdots & 0 \end{bmatrix} \tag{23}$$

Computing the sum of each column of A_n according to Eq. (18), we take the jth column as

$$\begin{aligned} & f_{1j} + f_{2j} + \cdots + f_{nj} \\ &= \frac{l\text{-}d_{1j}}{\sum_{t\in T_j} l\text{-}d_{tj}} + \frac{l\text{-}d_{2j}}{\sum_{t\in T_j} l\text{-}d_{tj}} + \cdots + \frac{l\text{-}d_{nj}}{\sum_{t\in T_j} l\text{-}d_{tj}} = 1 \\ &= \sum_{\in T_j} \frac{l\text{-}d_{tj}}{\sum_{t\in T_j} l\text{-}d_{tj}} \end{aligned} \tag{24}$$

Therefore, Eq. (20) is convergent and satisfies

$$\sum_{m=1}^{n} SHR(N_m) = \sum_{m=1}^{n} R^{\alpha}(N_m) \tag{25}$$

where $\alpha \in [0, \eta]$.

4.3 Dynamic calculation

To cope with the variability of the dark data set whose images (nodes) may be added or deleted, we design a dynamic method to fast calculate scores for the changed data set.

Addition. Generally, when a new node is added to graph G, this node will be marked as N_{n+1} by default if it is connected with one of the n nodes. (As shown in Figure 8a, N_* is added to graph $G_{addition}$ which contains 4 nodes. We directly mark N_* as N_5 because N_* is connected with N_1 and N_4.) And T_{n+1} denotes the set including orders of all nodes connected with N_{n+1} where $T_{n+1} \subseteq [1, n]$. Then we analyze how matrix A_n will change and calculate the scores and ranks for $n+1$ nodes.

(1) For a given new node which is marked as N_{n+1}, we traverse all the n nodes and calculate $d_{i(n+1)}$ according to Eq. (1) for $i \in [1, n]$. Thus, we obtain the set T_{n+1} that contains orders of all nodes connected with N_{n+1}.
(2) If $T_{n+1} = \varnothing$, N_{n+1} is a isolated node and we terminate the following process. Otherwise, we calculate matrix A_{n+1}.
(3) For $\forall\ i \in \{1, 2, \cdots, n\} \cap T_{n+1}$, we calculate the ith column elements of A_{n+1}. For $\forall\ j \in \{1, 2, \cdots, n\}$, the jth element of the ith column of matrix A_n is $f_{ji} = \begin{cases} \frac{l\text{-}d_{ji}}{\sum_{t\in T_i} l\text{-}d_{jt}} & \exists d_{ji}, \\ 0 & otherwise. \end{cases}$

while the jth element of the ith column of matrix A_{n+1} should be $\begin{cases} \frac{l-d_{ji}}{\left(\sum_{t\in T_i} l-d_{ti}\right)+l-d_{(n+1)i}} & \exists d_{ji}, \\ 0 & otherwise. \end{cases}$ In other words, we get $\frac{\frac{l-d_{ji}}{\left(\sum_{t\in T_i} l-d_{ti}\right)+l-d_{(n+1)i}}}{\frac{l-d_{ji}}{\sum_{t\in T_i} l-d_{jt}}} = \frac{\sum_{t\in T_i} l-d_{ti}}{\left(\sum_{t\in T_i} l-d_{ti}\right)+l-d_{(n+1)i}}$. Therefore, the jth element of the ith column of A_n will change into the jth element of the ith column of A_{n+1} if multiplied by $\frac{\sum_{t\in T_i} l-d_{ti}}{\left(\sum_{t\in T_i} l-d_{ti}\right)+l-d_{(n+1)i}}$. Next, the $(n+1)$th element of the ith column is $\frac{l-d_{(n+1)i}}{\left(\sum_{t\in T_i} l-d_{ti}\right)+l-d_{(n+1)i}}$. At last, we update T_i by setting $T_i = T_i \cup \{n+1\}$. Consequently, we can directly obtain the ith column elements of A_{n+1} based on A_n for $i \in \{1, 2, \cdots, n\} \cap T_{n+1}$.

(4) For $\forall\ i \in \{1, 2, \cdots, n\} \backslash T_{n+1}$, we calculate the ith column elements of A_{n+1}. N_i has no connection with N_{n+1}, so the jth element of the ith column of A_n directly becomes the jth element of the ith column of A_{n+1} for $\forall\ j \in \{1, 2, \cdots, n\}$. And the $(n+1)$th element of A_{n+1} is 0. Consequently, we can directly obtain the ith column elements of A_{n+1} based on A_n for $i \in \{1, 2, \cdots, n\} \backslash T_{n+1}$.

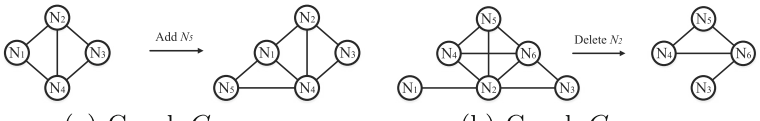
(5) We calculate the $(n+1)$th column elements of A_{n+1}. For $\forall\ j \in \{1, 2, \cdots, n+1\}$, the jth element of the $(n+1)$th column of A_{n+1} is

$$\begin{cases} \frac{l-d_{j(n+1)}}{\sum_{t\in T_{n+1}} l-d_{t(n+1)}} & \exists d_{j(n+1)}, \\ 0 & otherwise. \end{cases}$$

(6) For above D_{n+1}, we use Eq. (20) to calculate scores for $n+1$ nodes.

Deletion. Similarly, if we delete a node from graph G, how can fast adjust the matrix A_n and calculate score for each node in the new graph? For example, as shown in Figure 8b, N_2 is deleted from $G_{deletion}$ which contains 6 nodes. However, N_1 will be removed from $G_{deletion}$ because N_1 is only connected with N_2. Also, those edges which are connected with N_2 will also disappear. Generally, for $i \in \{1, 2, \cdots, n\}$, once we delete N_i from graph G, those edges connected with N_i will be removed from G. Of course, if N_i is deleted, those isolated nodes will be also removed.

We describe the steps that analyze how matrix A_n will change and calculate scores for the remaining nodes after deleting N_i.



(a) Graph $G_{addition}$ (b) Graph $G_{deletion}$

Figure 8 Addition and deletion on graph

(1) We count those nodes which are only connected with N_i. For $\forall\ k \in T_i$, if f_{ik} equals 1, N_k will be a isolated node if deleting N_i. We mark the order of all N_k as the set I_i, where $I_i \subseteq T_i$ and $|I_i|$ denotes the number of elements in I_i.

(2) If (n-$|I_i|$) equals 1, the remaining nodes of graph G will all be isolated nodes after removing N_i and we terminate the following process. Otherwise, we continue to adjust A_n and obtain our expected $A_{n-|I_i|-1}$.

(3) For $\forall\ t \in \{1, 2, \cdots, n\} \cap T_i \backslash I_i$, we adjust the tth column elements of A_n. For $\forall\ j \in \{1, 2, \cdots, n\}$, the jth element of the tth column of matrix A_n is $f_{jt} = \begin{cases} \frac{l-d_{jt}}{\sum_{x \in T_t} l-d_{xt}} & \exists d_{ij}, \\ 0 & otherwise. \end{cases}$ However, after we delete N_i, the jth element of the tth column of matrix A_n should be $\begin{cases} \frac{l-d_{jt}}{\left(\sum_{x \in T_t} l-d_{xt}\right)-(l-d_{it})} & \exists d_{jt}, \\ 0 & otherwise. \end{cases}$ In other words, we get $\frac{\frac{l-d_{jt}}{\left(\sum_{x \in T_t} l-d_{xt}\right)-(l-d_{it})}}{\frac{l-d_{jt}}{\sum_{x \in T_t} l-d_{xt}}} = \frac{\sum_{x \in T_t} l-d_{xt}}{\left(\sum_{x \in T_t} l-d_{xt}\right)-(l-d_{it})}$. Therefore, we adjust the jth element of the ith column of A_n via multiplying it by $\frac{\sum_{x \in T_t} l-d_{xt}}{\left(\sum_{x \in T_t} l-d_{xt}\right)-(l-d_{it})}$. Then we update T_i by setting $T_i = T_i \backslash \{i\}$. Consequently, we directly change the tth column elements of A_n for $t \in \{1, 2, \cdots, n\} \cap T_i \backslash I_i$.

(4) For $\forall\ t \in \{1, 2, \cdots, n\} \backslash T_i$, we adjust the tth column elements of A_n. N_i has no connection with N_t, so we do not change the tth column elements of A_n. Consequently, we reserve the tth column elements of A_n.

(5) Now, we start to calculate the expected $A_{n-|I_i|-1}$. We have adjusted the tth column elements of A_n for $t \in 1,\ 2,\ \cdots,\ n \cap T_i \backslash I_i$. Besides, we have reserved the tth column elements for $t \in \{1, 2, \cdots, n\} \backslash T_i$. Now, we directly delete the tth row as well as the tth column elements of A_n. After removing above ($|I_i|$ +1) rows and columns, we obtain our expected $A_{n-|I_i|-1}$.

(6) For above $A_{n-|I_i|-1}$, we use Eq. (20) to calculate scores for the remaining $n - |I_i| - 1$ nodes.

In this way, we avoid huge computational overhead by updating only few elements of A_n instead of recalculating the whole updated matrix.

4.4 Advantage for assessment

Our SHR is designed with the following advantages. (1) Overall importance. For each node, SHR takes not only its number of connected links but also the weight on edges into consideration from an overall perspective. (2) Rationality. After hash codes with generalization ability are generated by DSTH, SHR specially designs the association relationship between nodes and exposes how a node is affected by another, which can effectively make full use of these Hamming distances and reasonably calculate importance score for each node shown in the shadow part of Figure 7. (3) Convergence. SHR can converge to a stable result owing to our well-designed iteration matrix, which ensures our algorithm can work effectively. (4) SHR can deal with dynamic image dataset by reducing huge computational overhead.

5 Evaluation

In this section, we evaluate our framework and conduct extensive experiments as follows:

(1) Using the feature extraction method with generalization ability, DSTH can solve the out-of-sample problem (see Section 5.1).
(2) The efficiency of graph building using hash codes generated by DSTH can be greatly improved with allowed accuracy loss (see Section 5.2).
(3) SHR can calculate the importance score for each node effectively and efficiently on large-scale datasets (see Section 5.3).
(4) SHR can help highlight and prepose those data whose semantic information account for higher proportion in original dataset (see Section 5.4).
(5) Our framework can deal with large-scale datasets and return a concise score (assessment result) based on the user's query, which assists the user to make a correct decision on subsequent operations with this dataset (see Section 5.5).

We implement the first four experiments on the public CIFAR-10 dataset, and respectively adopt self-defined and large-scale Tencent datasets to conduct the last two experiments. Our evaluation is executed using Python tools including TensorFlow and Scikit-Learn library. Our experiments are run on two 10-core Intel Xeon E5-2640 machines with 128GB of DDR4 memory. At last, we conduct the experiment on Tencent dataset using 12 machines.

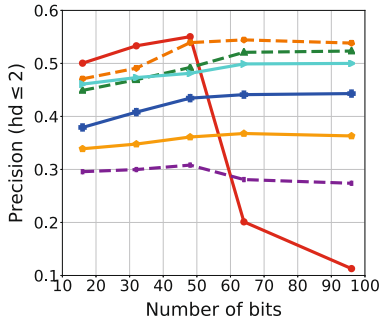
5.1 Generalization ability

In this section, we verify the effectiveness of DSTH mapping hash by executing code length analysis (CLA) and precision-recall (PR) on CIFAR-10. On the one hand, we compare with the state-of-the-art methods on original datasets to show the superiority of the algorithm. On the other hand, compared with those best methods on reorganized CIFAR-10 dataset, our DSTH also shows a stronger generalization ability, which solves the out-of-sample problem.

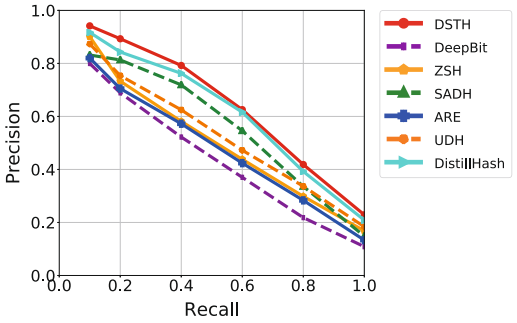
In practice, we execute code length analysis (CLA) and precision-recall (PR) on CIFAR-10, compared with the state-of-the-art of single target unsupervised deep hashing algorithms and zero-shot hashing algorithms including DeepBit [15], ZSH [35], SADH [28], ARE [8], UDH [43] and DistillHash [36]. CIFAR-10 is a labeled data set, which consists of 60,000 32×32 color images in 10 classes, with 6000 images per class. There are 5000 training images and 1000 test images in each class. Particularly, we select the average value of the top 15% nodes in terms of precision in each class as the precision of CLA. In the experiment, we select GoogLeNet and classification model trained on ImageNet to extract deep features. Meanwhile, the CNN structure for generating hash model is similar to [18, 44].

Figure 9a shows the mAP@15% [30] CLA with $hd \leq 2$ and 48-bit codes PR performance on CIFAR-10 compared with others. As Figure 9a shows, DSTH yields a prominent dominance and 48-bit codes is the best at precision of 55.04%. The performance is higher than that of UDH by 1.12%. As Figure 9b shows, DSTH yields a significant dominance in term of precision with 48-bit codes. The results show our superiority.

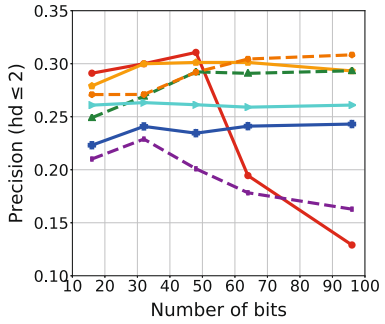
Furthermore, to validate the advantage in solving the out-of-samples problem mentioned in Section 6, we adjust the distribution of CIFAR-10 by taking the image of cat or automobile off from the training set. Besides, we stipulatethat it is correct to classify a cat as a dog and an automobile as a truck. Figure 9c, d show the mAP@15% CLA and 48-bit codes PR



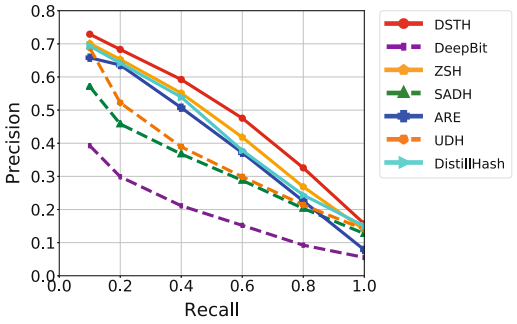
(a) mAP@15% CLA



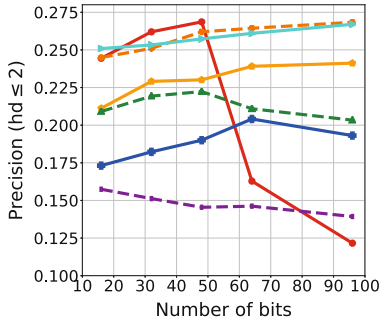
(b) 48-bit codes PR



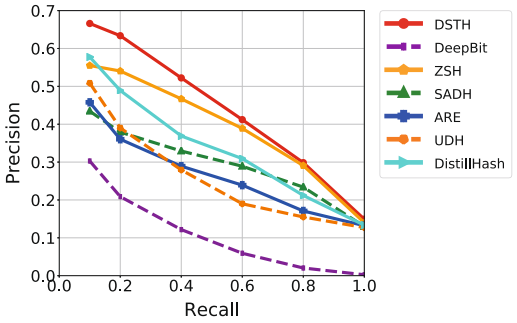
(c) mAP@15% CLA without Training Set of Cat



(d) 48-bit codes PR without Training Set of Cat



(e) mAP@15% CLA without Training Set of Cat and Automobile



(f) 48-bit codes PR without Training Set of Cat and Automobile

Figure 9 mAP@15% Code Length Analysis (CLA) and Precision-Recall (PR) curve on CIFAR-10

performance without training set of cat. Figure 9e, f shows the mAP@15% CLA and 48-bit codes PR performance without training set of cat and automobile. As shown in Figure 9c, d, e, f, our results of mAP@15% CLA and PR also yield a significant dominance. As the same as the code length of previous, 48-bit is best for redefined CIFAR-10 data sets at precision of 31.07% and 26.86% respectively. Specially, as shown in Figure 9c, e, although the gap is reduced, performances of DSTH with 48-bit codes are higher than that of ZSH and UDH by 0.95% and 0.66% respectively in precision, illustrating the superiority of ours for solving the problem of out-of-samples.

5.2 Graph building efficiency

In this section, for verifying that building a graph by Hamming distance is more efficient than Cosine and Euclidean distance, we exhibit the time of graph building using 3 metrics with 48-bit vectors (48-bit hash codes and 48 float numbers). In order to ensure the fairness, we set $\Omega = 48$, making all nodes fully connected. As shown in Figure 10a, the horizontal coordinate represents the number of nodes while the ordinate represents the graph building time. With the same scale of nodes, the graph building time of Hamming distance is nearly 100 times less than that of Cosine and Euclidean, which shows that Hamming distance has overwhelming predominance over other 2 metrics in building a graph. Especially, with the scale of nodes increasing, the graph building time of Cosine and Euclidean grows exponentially which is unacceptable, making that Hamming distance becomes the better choice.

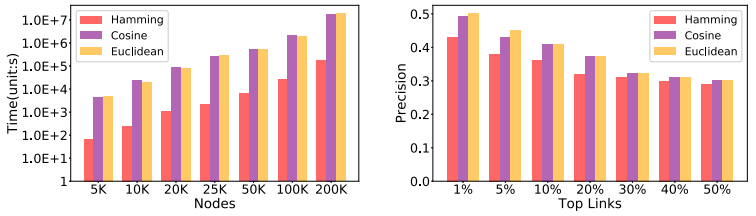
In order to compare precision of graph in 3 metrics, we choose more accurate links from top 1% to top 50% according to the weight of edges with 200,000 nodes. For example, we choose those edges on which the Hamming distance is smaller, while selecting the edges whose Cosine and Euclidean distance is larger. As shown in Figure 10b, Hamming distance is 0.070 lower than Euclidean at top 1% links in the worst case and 0.009 lower than Cosine at top 30% links in the best case in term of precision of graph. Averagely, Hamming distance is 0.040 lower than other 2 metrics in 7 cases.

On the whole, there is not a marked difference of precision between 3 metrics, although hashing will bring certain loss to precision. However, Hamming distance has overwhelming predominance in building a graph in term of time cost. We use hashing and Hamming distance in the follow-up work with comprehensive consideration of tradeoff between efficiency and precision, since an acceptable margin of error is allowed.

5.3 SHR calculation

In this section, we verify that SHR can obtain reasonable importance score for each node on single and double connected domains respectively. Besides, we present the acceptable actual calculation cost of SHR under different number of nodes and iterations, indicating that SHR is able to adapt to large-scale scenes. In practice, we conduct experiments with 48-bit hash codes.

We prove feasibility using graph G_1 shown in Figure 11a. The results calculated by SHR are shown in Table 1 with $\eta = 65$, when we set ε= 1.0E-10 and $R^0(N_m) = 1$ where $m \in [1, 9]$. As shown in Figure 11a, N_1 has the most connections, while N_3 owns more edges where the Hamming distance is smaller relatively. Therefore, the results are deemed reasonable.



(a) Graph Building Time with Different Scale of Nodes

(b) Precision of Graph using Different Top Links with 200,000 Nodes

Figure 10 Graph building time with different scale of nodes and precision of graph with 200,000 nodes using Hamming, Cosine and Euclidean distance

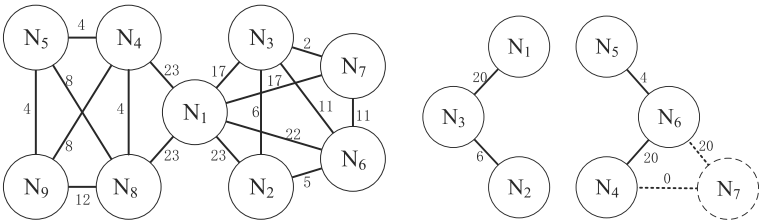


Figure 11 Example graphs

We prove reliability using graph G_2 shown in Figure 11b. Different from G_1, G_2 consists of 2 connected domains. The results calculated by SHR are shown in Table 2 with $\eta = 141$. Similarly, we set ε= 1.0E-10and $R^0(N_m) = 1$ where $m \in [1, 6]$. Obviously, N_3 and N_6 get the same rank and play the most important role in their own connected domain. Furthermore, when we add N_7 which has the same hash code as N_4, theresults will change shown in Table 2. It is easy to find that the ranks in the left domain do not change but the sum of ranks in the right domain has added one unit. Consequently, SHR is able to calculate the rank of each node in its own connected domain, without being influenced by other connected domains. Also, N_4 and N_7 own the same score, illustrating that those nodes which own the same hash code will get the same score.

For illustrating the performance of SHR, the number of nodes, the number of links, the time cost including calculating A in Eq. (23) and iterating, and the number of iterations are displayed with ε= 1.0E-15, 1.0E-11, 1.0E-7 and $\Omega = 24$ after graph building. As shown in Table 3, as the number of nodes increases, the number of iterations is relatively stable, since it is not determined by the scale of nodes and the main factor that causes the time cost of computing is the acquisition of A. In addition, the growth of time cost and number of links are acceptable with scale of nodes increasing. Even though the number of links exceeds 1 billion, the number of iterations is very close to that of PageRank [23] proposed by Google, which shows that SHR algorithm is sufficient to deal with large-scale computing.

5.4 Predominant semantics

Based on the validity shown in Section 5.3, we verify SHR can highlight and prepose those data whose semantic information account for higher proportion in this section, which shows our algorithm has practical significance for assessment tasks. In the next experiment, if the

Table 1 Score and rank in graph G_1

Node	Hash code	Score	Rank
N_1	FFFFFFFFFFFF	1.118	1
N_2	FFFFFF800000	1.028	5
N_3	FFFFFFFE0000	1.069	2
N_4	C000007FFFFF	1.051	4
N_5	0000001FFFFF	0.879	8
N_6	FBFF7F8000E0	0.980	7
N_7	FFFFFF7E0080	1.055	3
N_8	C0003079FFFF	0.996	6
N_9	0300001FFE7F	0.824	9

Table 2 Score and rank in graph G_2

Node	Hash code	$N_1 \sim N_6$		$N_1 \sim N_7$	
		Score	Rank	Score	Rank
N_1	1FFFFFFFFFF	0.646	5	0.646	7
N_2	FFFFFF800000	0.894	4	0.894	5
N_3	FFFFFFFE0000	1.460	1	1.460	1
N_4	000000000001	0.908	3	1.009	3
N_5	C000007FFFFF	0.632	6	0.649	6
N_6	0000001FFFFF	1.460	1	1.333	2
N_7	C000007FFFFF	–	–	1.009	3

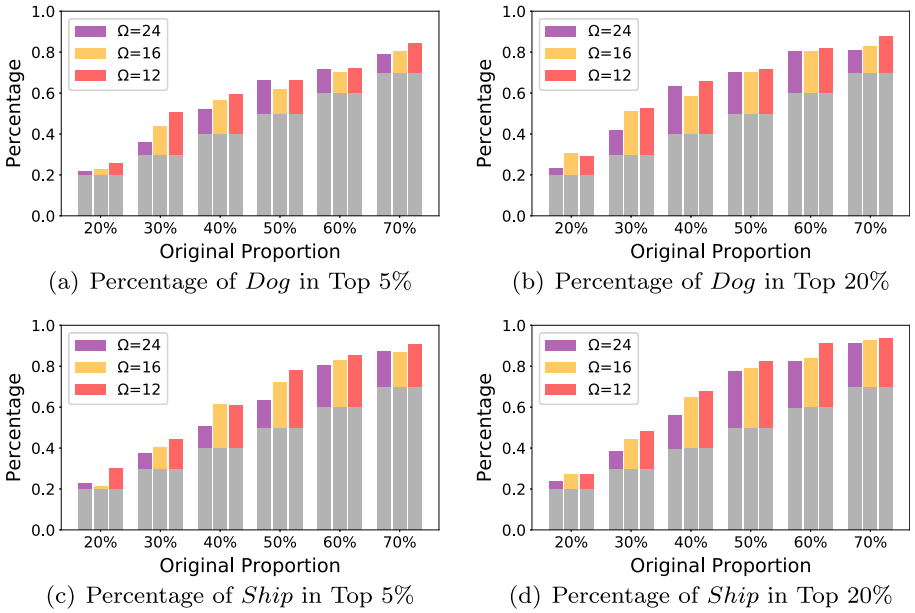
ranked results are correct, those images whose semantic distribution account for higher proportion in original data set will obtain larger scores and higher ranks. Thus, based on CIFAR-10 test set, under the premise that the amount of images of other classes remains unchanged, we choose one class as a study object to be added to the sample, making the amount of this class reach 20%, 30%, 40%, 50%, 60% and 70% respectively on the whole data set. We collect proportion of this class in the top 5% and top 20% of ranked results in 6 cases mentioned above. We set ε= 1.0E-7 and choose Ω= 24, 16 and 12 to conduct the experiments.

Figure 12a, b show the percentage with different Ω in the top 5% and top 20% of ranked results respectively when choosing cat as the study object. As shown in Figure 12a, b, in all cases, SHR magnifies original proportion of cat (the part that goes beyond the blue column), indicating the efficiency of this algorithm. Detailedly, as shown in Table 4, compared with Ω= 16 or 24, setting Ω= 12 yields better performance on the magnification of the *dog* percentage in the top 5% of ranked results, where the *dog* percentage is averagely 2.8% higher than original proportion in 6 cases. Among them, the best result exceeds the original proportion by 20.6% in the case of 30%. In the top 20% of ranked results, choosing Ω= 12 yields better performance in most of the cases, where the titdog percentage is averagely 2.6% higher than original proportion in 6 cases. Among them, the best result exceeds the original proportion by 25.4% in the case of 40%.

Similarly, Figure 12c, d show the results choosing *ship* as the study object. As is shown in Figure 12c, d, SHR achieves the same effect. Detailedly, as shown in Table 4, in the top 5% of ranked results, compared with other setting of Ω, the percentage of *ship* shows the superiority in most of the cases while choosing Ω= 12, which is averagely 3.3% higher than original

Table 3 Statistic of the number of nodes, the number of links, the time cost and the number of iterations with different ε when SHR is running

Node	Link	Time cost	Number of iterations(ε=)		
	Ω=24	unit:s	1.0E-15	1.0E-11	1.0E-7
5 K	7.5 M	88±0.2311	57	40	24
10 K	29 M	380±0.9912	57	40	23
20 K	111 M	1430±2.016	58	40	24
25 K	223 M	2991±3.001	58	40	23
50 K	737 M	8189±5.465	55	38	22
100 K	2.87G	29,372±12.188	52	36	21
200 K	15.33G	172,839±42.077	49	33	20



(a) Percentage of *Dog* in Top 5%

(b) Percentage of *Dog* in Top 20%

(c) Percentage of *Ship* in Top 5%

(d) Percentage of *Ship* in Top 20%

Figure 12 Trend for Percentage of *Dog* and *Ship* in ranked result using different Ω with 48-bit codes

proportion in 6 cases. Among them, the best result exceeds the original proportion by 27.9% in the case of 50%. Besides, in the top 20% of ranked results, the percentage of *ship* also shows great superiority with Ω= 12, which is averagely 3.2% higher than original proportion in 6 cases. Among them, the best result exceeds the original proportion by 31.1% in the case of 60%.

It should be explained that better precision and shorter time cost can be captured theoretically when Ω<12. However, the reduction of links causes too many isolated nodes all of whom get the same score, which may bring certain loss to the ranked results. Usually, with a larger scale of nodes, hash codes are more widely distributed, thus setting a smaller Ω will not

Table 4 Detail for Percentage of *Dog* and *Ship* in ranked result using different Ω with 48-bit codes

C	Top	Ω	Original proportion					
			20%	30%	40%	50%	60%	70%
Dog	5%	24	0.217	0.362	0.519	0.662	0.713	0.792
		16	0.226	0.438	0.562	0.621	0.699	0.807
		12	**0.254**	**0.506**	**0.592**	**0.663**	**0.721**	**0.843**
	20%	24	0.230	0.418	0.631	0.704	0.801	0.810
		16	**0.305**	0.513	0.582	0.704	0.802	0.829
		12	0.291	**0.528**	**0.654**	**0.716**	**0.816**	**0.878**
Ship	5%	24	0.225	0.376	0.503	0.637	0.802	0.873
		16	0.211	0.407	**0.613**	0.723	0.829	0.867
		12	**0.299**	**0.444**	0.609	**0.779**	**0.854**	**0.907**
	20%	24	0.239	0.387	0.562	0.775	0.826	0.914
		16	0.270	0.446	0.645	0.789	0.839	0.926
		12	**0.271**	**0.481**	**0.678**	**0.825**	**0.911**	**0.937**

The entries in boldface represent the maximum values in ranked results.

result in too many isolated nodes. In our follow-up research, we intend to study this issue in depth.

As above experimental results show, both in the top 5% and 20% of ranked results, SHR effectively highlights and preposes the data whose semantic information account for higher proportion in original data set after ranking. Thus, our SHR is correct and effective in practical applications.

5.5 Assessment of query

Finally, based on the validity shown in Section 5.3, we verify that our framework can efficiently complete online assessment work according to the user's query task on large-scale real dark dataset. Also, it can guide and assist the user to conduct subsequent data mining work in order to show our framework is effective to complete the dark dataset assessment. We apply our framework to real-world data set of Tencent which is collected from QQ albums, QQ chat and WeChat in a certain period. The size of data is around 5 TB consisting of 1,000,000 images. Specially, according to the results shown in Section 5.4 that a smaller Ω is proved to be feasible at a million scale, we select $\Omega = 2$ to conduct this experiment.

First, we complete offline calculation to get ranks of images with building a graph by **2.91 h**, constructing matrix 23 by **9.74 h** and iterating by **5 min**. We account the top 500 images of rankedresults and find that the images including females account for 77.6%, of which the individual images and group images account for 44.2% and 33.4% respectively. Besides, the images including males account for 23.6%, of which the individual images and group images account for 3.4% and 20.2% respectively. The others consist of some images including children which account for 37.4%, some images including buildings or landscape which account for 10.4%, a few images including animals which account for 7.4%, and several images including commodities, food or screenshots. From the result of proportion, the main semantic components of this data set are daily life images, most of which are the images of women and children. Therefore, the data set are apt for those applications that are interested to mine data about women and children.

Of course, some images containing important semantic information may not be ranked in top 500 for the data set consisting of 1,000,000 images. Therefore, we carry out assessment for specific applications according to ranked results. We use the general method mentioned in Section 2.2 for assessment by analyzing the value of Tencent data set for three tasks which include human intimacy as task-*A*, children playing in the outskirts as task-*B*, and motorcycle on the bridge as task-*C* respectively. The agent images are collected from BAIDU search and their respective weights are given below. Figure 13 displays the assessing process and results for above tasks. As shown in Figure 13, intimacy images representing the first task are associated with three images whose ranks are high, so it is worth carrying out data mining on this data set for the task-*A*. For task-*B* which are associated with two images contained in Tencent data set, the images of children have high scores and images about landscape own medium ranks. However, the weighted score of this task is relatively high, which shows this data set can help analyze images about children playing in the outskirts. Although there are three images that match the task-*C*, neither motorcycle nor bridge obtain high scores, so this data set are not suitable for the task-*C*.

To further verify the correctness of our assessment, we show the efficiency of the mining algorithm according to our framework. Two sets of results returned by our framework with diverse *hd* and the deep model for above tasks are shown in Table 5. The statistical results

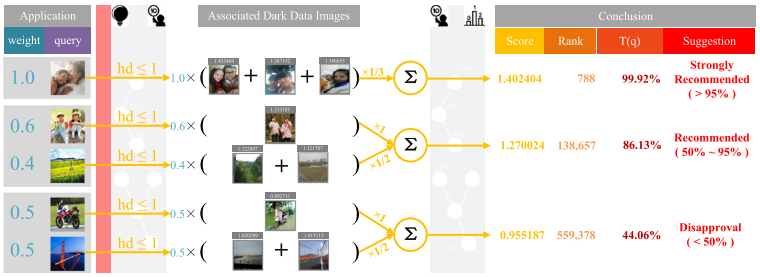


Figure 13 The process of assessment on Tencent data set for three real requirements

include the number of recommended images that above two methods return, the number of suitable images that SSD [17] algorithm picks out from the recommended images, the proportion of suitable images in recommended images, and the time cost that SSD spends on the recommended images.

It is easy to find that, as *hd* becomes larger, the number of recommend images increases while the number of considered valid images is decreasing, because a larger *hd* will lead to a greatly ascending number of those weakly correlatedimages. Even so, our framework has an overwhelming predominance over deep model in terms of precision of the recommendation (ratio). This is because the semantic information which contains the features with generalization ability extracted by DSTH and the association analysis produced by SHR can not directly captured by deep model.

In addition, we also find that although task-*B* has a similar rank with task-*A*, the number of its recommendation is greatly larger, because it is associated with two different semantics. At the same time, task-*C* does not get a large number of recommendation though also associated with two semantics, because the associated semantics account for a low proportion. Even so, for dominant data, our algorithm does not give more recommendation data than deep model, which reduces the analysis time cost for subsequent data mining. Our result benefits from the combined effects of features with generalization ability extracted by DSTH and association analysis produced by SHR, which reduces the number of rough recommended images about single object that deep model tends to return. Finally, according to the ratio, even though

Table 5 Effect comparison between our framework and deep model

	Task	hd					Deep model
		4	8	12	16	24	
Recommendation	A	1522	6952	29,368	41,996	16,668	661,254
	B	4012	12,353	39,399	70,114	311,648	683,722
	C	2512	7076	10,431	21,637	98,795	19,315
SSD adoption	A	1391	5332	17,747	28,139	90,121	21,116
	B	2119	4406	19,863	33,113	79,334	15,197
	C	105	177	218	299	594	222
Ratio	A	0.914	0.767	0.604	0.67	0.541	0.032
	B	0.528	0.357	0.504	0.472	0.255	0.022
	C	0.042	0.025	0.021	0.014	0.006	0.011
Time (s)	A	297.13	1458.021	6090.028	8784.37	34,781.16	13,711.032
	B	1121.07	2846.15	13,973.51	27,729.33	94,101.416	177,777.59
	C	368.03	1137.71	1799.17	3541.32	16,618.12	3280.11

setting $hd \leq 24$, the valid recommendation of task-*A*, task-*B* and task-*C* account for more than 50%, 25% and 0.6% respectively. This is completely consistent with our importance rank and suggestion, indicating that our framework can truly and effectively expose the amount and entity of relevant data in the dataset. Compared with classification results generated by deep learning, our results is the straightforward expression for user's requirements and can provide value judgment for more extensive applications. Consequently, our algorithm which executes association analysis by semantic hash is more effective for value assessment.

6 Related works

Dark data Heidorn has demonstrated the value of dark data by the long tail theory in economics and given the concept of dark data lightening which means constructing relationship according to a new task [6]. Furthermore, he presents the implementation of astronomical dark data management using unified databases [7]. *File WinOver* System [29] is proposed to complete the dark data judgment and risk assessment through fingerprint. Cafarella [1] mentions that the value of dark data depends on both the requirements of the task and the ability of value extraction. GeoDeepDive [41] and DeepDive [42] propose a pragmatic scheme of dark data mining system by correcting annotations and associations of data according to feedback from users. Unfortunately, his work uses the method of human feedback, which requires a long period of time and is affected by human factors. It is not suitable for real-time judgment scenarios. However, the way that relates data inspires and prompts us to further complete the work of assessment.

Content-based hashing for image Content-based Hashing is a technique that generates compact hash codes from the original data to represent the main content which preserves the data semantic relationship [13, 14, 19, 25–28, 34, 45]. It is more efficient to construct relationship between images in large-scale scene because of fast quantization by XOR operation. With the success of Convolution Neutral Network (CNN) [11] in feature extraction, deep hashing becomes the mainstream for image hashing. For unlabeled images, DSTH has better ability to solve out-of-samples problem, because it is able to regard the instances beyond scope of cognition as the samples which have been learnt in the model as close as possible. Therefore, DSTH is a better hashing method to reduce the sensitivity of non-cognitive objects which are widely distributed in large-scale data set.

Graph-based mining for image Most of unsupervised image mining solutions are based on image content and the similarity graph connecting images with each other. Commonly, there are Euclidean distance [4], Cosine [10] and Hamming distance [21] for quantization when different types of features are connected. Specially, Stefan et al. [21] adopts hash code and Hamming distance to construct similarity graph with illustration of validation in large-scale scene, although it brings certain loss to precision. However, it does not consider construction with restricted Hamming distance to improve efficiency and clustering processing is costly.

Graph-based ranking Calculating the importance score of each node is a special quantization method without clustering. It is more effective to get evaluation standards by ranking for each node globally. PageRank [23] considers out-degree of related nodes as impact factor for data ranking. Fabian et al. [24] applies random walking to ranking community images for

searching, which has achieved good results. Personalized Rank [37] introduced the concept of probability to improve the RegEx in PageRank. However, above graph-based ranking algorithms focus on in-degree and out-degree, neglecting the weight on edges, resulting that they are not competent for quantization with Hamming distance. TextRank [22] and SentenceRank [5] take the weights on edges into consideration, both of which mentioned applying PageRank to improve their algorithms, but none of them give proof of convergence.

7 Conclusions

In this paper, we proposed a framework for image dark data assessment. We first transformed unlabeled images into hash codes by our developed DSTH algorithm, then constructed a semantic graph using restricted Hamming distance, and finally used our designed SHR algorithm to calculate the overall importance score for each image. During online assessment, we first translated the user's query into hash codes using DSTH model, then matched the suitable data contained in the dark data, and finally returned theweighted average value of these matched data to help the user cognize the dark data. Experimental results showed DSTH can extract semantic features with generalization ability, and SHR can correctly calculate the importance scores according to the similarity between data, and our framework can apply to large-scale datasets.

Acknowledgments This work is supported by the Innovation Group Project of the National Natural Science Foundation of China No.61821003 and the National Key Research and Development Program of China under grant No.2016YFB0800402 and the National Natural Science Foundation of China No.61672254 and No.61902135.

References

1. Cafarella, M.J., Ilyas, I.F., Kornacker, M., Kraska, T., Ré, C.: Dark data: are we solving the right problems? In: ICDE, pp. 1444–1445 (2016)
2. Cai, H.Y., Huang, Z., Srivastava, D., Zhang, Q.: Indexing evolving events from tweet streams. In: ICDE, pp. 1538–1539 (2016)
3. Cao, Y., Long, M., Liu, B., Wang, J.: Deep cauchy hashing for hamming space retrieval. In: CVPR, pp. 1229–1237 (2018)
4. Gao, S., Cheng, X., Wang, H., Chia, L.-T.: Concept model-based unsupervised Web image re-ranking. In: ICIP, pp. 793–796 (2009)
5. Ge, S.S., Zhang, Z., He, H.: Weighted graph model based sentence clustering and ranking for document summarization. In: ICIS, pp. 90–95 (2011)
6. Heidorn, P.B.: Shedding light on the dark data in the long tail of science. Libr. Trends. **57**(2), 280–299 (2018)
7. Heidorn, P.B., Stahlman, G.R., Steffen, J.: Astrolabe: curating, linking and computing Astronomy's dark data. CoRR. abs/1802.03629 (2018)
8. Hu, M., Yang, Y., Shen, F., Xie, N., Shen, H.T.: Hashing with angular reconstructive Embeddings. IEEE Trans. Image Processing. **27**(2), 545–555 (2018)
9. Ioffe, S., Szegedy, C.: Batch normalization: accelerating deep network training by reducing internal covariate shift. In: ICML, pp. 448–456 (2015)
10. Keane, N., Yee, C., Liang, Z.: Using topic modeling and similarity thresholds to detect events. In: EVENTS@HLP-NAACL, pp. 34–42 (2015)
11. Krizhevsky, A., Sutskever, I., Hinton, G.E.: ImageNet classification with deep convolutional neural networks. In: NIPS, pp. 1106–1114 (2012)

12. Lai, H., Pan, Y., Liu, Y., Yan, S.: Simultaneous feature learning and hash coding with deep neural networks. In: CVPR, pp. 3270–3278 (2015)
13. Li, J., Wu, Y., Zhao, J., Lu, K.: Low-rank discriminant embedding for multiview learning. IEEE Trans. Cybernetics. **47**(11), 3516–3529 (2017)
14. Li, J., Lu, K., Huang, Z., Zhu, L., Shen, H.T.: Transfer independently together: a generalized framework for domain adaptation. IEEE Trans. Cybernetics. **49**(6), 2144–2155 (2019)
15. Lin, K., Lu, J., Chen, C.-S., Zhou, J.: Learning compact binary descriptors with unsupervised deep neural networks. In: CVPR, pp. 1183–1192 (2016)
16. Liu, H., Shao, M., Li, S., Yun, F.: Infinite ensemble for image clustering. In: SIGKDD, pp. 1745–1754 (2016)
17. Liu, W., Anguelov, D., Erhan, D., Szegedy, C., Reed, S., Fu, C.-Y., Berg, A.C.: SSD: single shot MultiBox detector. In: ECCV, pp. 21–37 (2016)
18. Liu, Y., Song, J., Zhou, K., Yan, L., Liu, L., Zou, F., Shao, L.: Deep self-taught hashing for image retrieval. IEEE Trans. Cybernetics. **49**(6), 2229–2241 (2019)
19. Luo, Y., Yang, Y., Shen, F., Huang, Z., Zhou, P., Shen, H.T.: Robust discrete code modeling for supervised hashing. Pattern Recogn. **75**, 128–135 (2018)
20. Mehmood, R., Zhang, G., Bie, R., Dawood, H., Ahmad, H.: Clustering by fast search and find of density peaks via heat diffusion. Neurocomputing. **208**, 210–217 (2016)
21. Michaelis, S., Piatkowski, N., Stolpe, M.: Solving Large Scale Learning Tasks. Challenges and Algorithms - Essays Dedicated to Katharina Morik on the Occasion of her 60th Birthday. Lecture Notes in Computer Science, vol. 9580, (2016)
22. Mihalcea, R. Graph-based ranking algorithms for sentence extraction, applied to text summarization. In ACL, (2004).
23. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web. Technical Report. Stanford InfoLab (1999)
24. Richter, F., Romberg, S., Hörster, E., Lienhart, R.: Multimodal ranking for image search on community databases. In: MIR, pp. 63–72 (2010)
25. Shen, F., Liu, W., Zhang, S., Yang, Y., Shen, H.T.: Learning binary codes for maximum inner product search. In: ICCV, pp. 4148–4156 (2015)
26. Shen, F., Shen, C., Liu, W., Shen, H.T.: Supervised discrete hashing. In: CVPR, pp. 37–45 (2015)
27. Shen, F., Shen, C., Shi, Q., van den Hengel, A., Tang, Z., Shen, H.T.: Hashing on nonlinear manifolds. IEEE Trans. Image Processing. **24**(6), 1839–1851 (2015)
28. Shen, F., Xu, Y., Liu, L., Yang, Y., Huang, Z., Shen, H.T.: Unsupervised deep hashing with similarity-adaptive and discrete optimization. IEEE Trans. Pattern Anal. Mach. Intell. **40**(12), 3034–3044 (2018)
29. Shukla, M., Manjunath, S., Saxena, R., Mondal, S., Lodha, S.: POSTER: WinOver enterprise dark data. In: SIGSAC, pp. 1674–1676 (2015)
30. Song, J., He, T., Gao, L., Xu, X., Shen, H.T.: Deep region hashing for efficient large-scale instance search from images. arXiv preprint arXiv:1701.07901 (2017)
31. Song, J., Gao, L., Liu, L., Zhu, X., Sebe, N.: Quantization-based hashing: a general framework for scalable image and video retrieval. PR. **75**, 175–187 (2018)
32. Wang, B., Yang, Y., Xu, X., Hanjalic, A., Shen, H.T.: Adversarial cross-modal retrieval. In: MM, pp. 154–162 (2017)
33. Xu, X., Shen, F., Yang, Y., Shen, H.T., Li, X.: Learning discriminative binary codes for large-scale cross-modal retrieval. IEEE Trans. Image Processing. **26**(5), 2494–2507 (2017)
34. Yang, Y., Ma, Z., Yang, Y., Nie, F., Shen, H.T.: Multitask spectral clustering by exploring Intertask correlation. IEEE Trans. Cybernetics. **45**(5), 1069–1080 (2015)
35. Yang, Y., Luo, Y., Chen, W., Shen, F., Shao, J., Shen, H.T.: Zero-shot hashing via transferring supervised knowledge. In: MM, pp. 1286–1295 (2016)
36. Yang, E., Liu, T., Cheng, D., Liu, W., Tao, D.: DistillHash: unsupervised deep hashing by distilling data pairs. In: CVPR, pp. 2946–2955 (2019)
37. Yu, L., Li, W., Lu, Z., Zhao, M.: Alternating pointwise-pairwise learning for personalized item ranking. In: CIKM, pp. 2155–2158 (2017)
38. Yu, L., Wang, Y., Zhou, K., Yang, Y., Liu, Y., Song, J., Xiao, Z.: A framework for image dark data assessment. In: APWeb-WAIM, pp. 3–18 (2019)
39. Yu, L., Wang, Y., Zhou, K., Yang, Y., Liu, Y.: Semantic-aware data quality assessment for image big data. Futur. Gener. Comput. Syst. **102**, 53–65 (2020)
40. Zhang, D., Wang, J., Deng, C., Jinsong, L.: Self-taught hashing for fast similarity search. In: SIGIR, pp. 18–25 (2010)
41. Zhang, C., Govindaraju, V., Borchardt, J., Foltz, T., Ré, C., Peters, S.: GeoDeepDive: statistical inference using familiar data-processing languages. In: SIGMOD, pp. 993–996 (2013)

42. Zhang, C., Shin, J., Ré, C., Cafarella, M.J., Niu, F.: Extracting databases from dark data with DeepDive. In: SIGMOD, pp. 847–859 (2016)
43. Zhang, H., Liu, L., Yang, L., Shao, L.: Unsupervised deep hashing with Pseudo labels for scalable image retrieval. IEEE Trans. Image Processing. **27**(4), 1626–1638 (2018)
44. Zhou, K., Yu, L., Song, J., Yan, L., Zou, F., Shen, F.: Deep self-taught hashing for image retrieval. In: MM, pp. 1215–1218 (2015)
45. Zhu, L., Shen, J., Liang, X., Cheng, Z.: Unsupervised visual hashing with semantic assistant for content-based image retrieval. IEEE Trans. Knowl. Data Eng. **29**(2), 472–486 (2017)

Affiliations

Ke Zhou[1] · Yangtao Wang[1] · Yu Liu[1] · Yujuan Yang[1] · Yifei Liu[1] · Guoliang Li[2] · Lianli Gao[3] · Zhili Xiao[4]

✉ Yu Liu
liu_yu@hust.edu.cn

Ke Zhou
k.zhou@hust.edu.cn

Yangtao Wang
ytwbruce@hust.edu.cn

Yujuan Yang
gracee@hust.edu.cn

Yifei Liu
yifeiliu@hust.edu.cn

Guoliang Li
liguoliang@tsinghua.edu.cn

Lianli Gao
lianli.gao@uestc.edu.cn

Zhili Xiao
tomxiao@tencent.com

[1] Wuhan National Laboratory for Optoelectronics, Huazhong University of Science and Technology, Wuhan, China

[2] Tsinghua University, Beijing, China

[3] University of Electronic Science and Technology of China, Chengdu, China

[4] Tencent Inc., Shenzhen, China