Automatically weighted binary multi-view clustering via deep initialization (AW-BMVC)

Khamis Houfar, Djamel Samai, Fadi Dornaika, Azeddine Benlamoudi, Khaled Bensid, Abdelmalik Taleb-Ahmed

Pattern Recognition 137 (2023) 109281

Contents lists available at ScienceDirect

Pattern Recognition

journal homepage: www.elsevier.com/locate/patcog

Automatically weighted binary multi-view clustering via deep initialization (AW-BMVC)

Khamis Houfar[b], Djamel Samai[b], Fadi Dornaika[a,c,d,*], Azeddine Benlamoudi[b], Khaled Bensid[b], Abdelmalik Taleb-Ahmed[e,**]

[a] Henan Key Lab. of Big Data Analysis and Processing, Henan University, Kaifeng, China
[b] Univ Ouargla, Fac. des Nouvelles Technologies de l'Information et de la Communication, Lab. de Génie Électrique (LAGE), Ouargla 30 000, Algeria
[c] University of the Basque Country UPV/EHU, San Sebastian, Spain
[d] IKERBASQUE, Basque Foundation for Science, Bilbao, Spain
[e] Institut d'Electronique de Microélectronique et de Nanotechnologie (IEMN), UMR 8520, Université Polytechnique Hauts de France, Université de Lille, CNRS, Valenciennes, 59313, France

ARTICLE INFO

Article history:
Received 10 August 2022
Revised 5 December 2022
Accepted 27 December 2022
Available online 31 December 2022

Keywords:
Multi-view clustering
Large scale
Anchors
Discrete representation and BD-FFT

ABSTRACT

Clustering is inherently a process of exploratory data analysis. It has attracted more attention recently because much real-world data consists of multiple representations or views. However, it becomes increasingly problematic when dealing with large and heterogeneous data. It is worth noting that several approaches have been developed to increase computational efficiency, although most of them have some drawbacks: (1) Most existing techniques consider equal or static weights to quantify importance across different views and samples, so common and complementary features cannot be used. (2) The clustering task is performed by arbitrary initialization without caring about the rich structure of the joint discrete representation, and thus poorly executed. In this paper, we propose a novel approach called "Auto-Weighted Binary Multi-View Clustering Via Deep Initialization" for large-scale multi-view clustering based on two main scenarios. First, we consider the distinction between different views based on the importance of samples, and therefore apply a dynamic learning strategy for the automatic weighting of views and samples. Second, in the context of initializing binary clustering, we develop a new CNN feature and use a low-dimensional binary embedding by exploiting the efficient capabilities of Fourier mapping. Moreover, our approach simultaneously learns a joint discrete representation and performs direct clustering using a constrained binary matrix factorization; the optimization problem is perfectly solved in a unified learning model. Experimental results conducted on several challenging datasets demonstrate the effectiveness and superiority of the proposed approach over state-of-the-art methods in terms of accuracy, normalized mutual information, and purity.

1. Introduction

In data mining, machine learning, and image processing applications, data is usually represented by multiple feature sets. In particular, in image analysis, each image can be described by different visual descriptors, such as HOG, SIFT, GIST, and LBP, etc. This type of data is called multiview data, and each feature representation corresponds to a view. These views have certain common and complementary information. These two pieces of information are important for the success of multi-view learning [1]. Existing technologies for multi-view learning can be broadly divided into supervised and unsupervised learning. This work focuses on one of the unsupervised learning techniques, namely clustering.

Most partitioning algorithms are only suitable for data with a single view. Even concatenating all views into a single view and then applying the most advanced clustering algorithms to that single view may not improve clustering performance due to redundancy of information, resulting in overfitting.

* Corresponding author at: University of the Basque Country UPV/EHU. Manuel Lardizabal, 1, 20018, San Sebastian, Spain.
** Co-corresponding author.
E-mail addresses: samai.djamel@univ-ouargla.dz (D. Samai), fadi.dornaika@ehu.eus (F. Dornaika), benlamoudi.azeddine@univ-ouargla.dz (A. Benlamoudi), bensid.khaled@univ-ouargla.dz (K. Bensid), Abdelmalik.Taleb-Ahmed@uphf.fr (A. Taleb-Ahmed).

https://doi.org/10.1016/j.patcog.2022.109281

Unlike traditional single-view clustering algorithms, multi-view clustering (MVC) methods [2,3] can be broadly divided into three main classes: (1) common feature subspace (combination by projection), such as using CCA [4] to minimize the cross-correlation error and then grouping the data using one of the clustering algorithms (e.g., *k*-means); (2) multi-view spectral clustering (common eigenvector matrix [5] and/or common graph similarity matrix [6]), which constructs multiple graphs to characterize the geometric structure followed by data partitioning using one of the extant clustering methods; (3) multi-view NMF clustering (common indicator matrix) [7] based on matrix factorization by splitting the feature matrix into a centroid matrix and cluster assignment matrix. With the recent flourish, hashing techniques, also known as binary code learning [8–10] have become increasingly important in big data analysis, resulting in fast Hamming distance computation and much lower memory requirements. Numerous multi-view hash approaches have emerged, especially for visual search and fast object detection, Wang et al. [8], Zhang et al. [11]. The hashing approach embeds the high-dimensional real-valued feature vectors into low-dimensional binary codes through a series of projections that can exchange information between multiple views without affecting the intrinsic aspects of the original space.

Despite significant progress in terms of fast computation and partially satisfactory results, most of the existing multi-view learning algorithms have three drawbacks:

1. Most of the existing models consider equal or static weights with additional parameters to estimate the contribution of each view, resulting in unsatisfactory representation learning.
2. Most of the existing models consider all samples equally in the clustering process.
3. The proposed methods lack a feasible and informative initialization of the binary codes of the clustering task, resulting in a poor local optimum.

To solve the above problems, a novel multi-view clustering method is developed in this paper: Auto-Weighted Binary Multi-View Clustering Via Deep Initialization (AW-BMVC). As a roadmap, we can divide the work motivation into two areas: Data Discovery and Analysis Model. The first area aims to exploit the rich attributes associated with real-world visual applications, different views and/or modalities, and how to integrate them into a unified binary representation. This integration drives us to make features more understandable and linearly separable by exploring nonlinear structures thanks to kernel advantages. The second track involves three mutual analysis steps: view diversity, sample variance, and clustering initialization. Here, the algorithm is automatically stimulated to implicitly measure the degree of importance of each view and explicitly determine the weight of each sample based on the learning loss. The intuition behind the automatically weighted strategy is first to reduce the impact of noisy and outlier views/samples by causing the model to make a good estimate of the importance of each view/sample based on the smallest learned loss error for the views and an explicit weight estimate for the samples; as a result of the automatically weighted scenario, additional manually adjusted parameters are avoided. The trick of automatic and adaptive weighting is also used in several recent machine learning algorithms. Binary embedding of samples interprets the mapping (embedding) from the kernelized higher-dimensional real space of features to the lower-dimensional Hamming space (Common Binary Codes). The advantage is twofold. First, the Common Binary Codes avoid the noise that normally affects the real-valued features in the different views. Second, by using the common binary data representation, the optimization steps can be made more efficient since some steps can be greatly simplified. The last and most important step in this area is to develop a new efficient strategy to bring the clustering model to the best optimal point. We should emphasize that the bottleneck for most multiview clustering approaches is the "fusion capability" that better approximates multiview data in a unified representation, taking into account how to fully exploit the different information that multiview data possess.

The following are the main elements of our contribution:

1. To exploit the heterogeneity of data with multiple views, we introduce an automatically weighted strategy to control the pairwise importance of each sample and each view separately. View weights are implicitly derived from the square root of the view objective function, while sample weights are explicitly estimated.
2. We propose an objective function whose optimization allows the joint estimation of the following entities: the common binary code of the data, the two sets of weights, the view-based mapping from the nonlinear representation to the common binary code space, the binary centroids, and the cluster assignment matrix.
3. Deep features are extracted from the Vgg16 network to obtain a good initialization for the proposed optimization. This feature is mapped to a low-dimensional Hamming space using a bidirectional FFT technique "BD-FFT". We use the generated binary vectors to initialize our iterative clustering algorithm.
4. Based on the presented objective function and alternating optimization scheme, the proposed method can outperform many state-of-the-art multiview clustering techniques, including those with real values.

The rest of the paper is as follows: Section 2 focuses on the key concepts and some related work. Section 3 provides a detailed understanding of the proposed work. Performance analysis through extensive experiments is discussed in Section 4. In Section 5, we conclude the paper with a future study.

2. Preliminaries and related work

2.1. Notations

In this paper, matrices are shown in bold uppercase letters and vectors are shown in bold lowercase letters. All notations used are summarized in Table 1.

2.2. Related work

Multi-view clustering (MVC) is an exciting topic in machine learning and has been explored using various strategies.

Before diving into related work, we present an anthology of MVC methods. The fact that a considerable number of multi-view approaches can be classified into the following categories: Spectral Clustering [12], Graph-based Clustering [13], and Subspace Clustering [14].

Here we briefly describe some interesting works:

- **RMSC** [15]: This method started with the construction of a graph for each view. Then, according to the low-rank constraint and sparse decomposition, a joint transition probability matrix was used as the crucial input to the standard Markov chain method for clustering. This method is not suitable for large datasets. It overlooks the flexible structure of the local manifold, which cannot satisfy the agreement between views.
- **DiMSC** [16]: It is a self-representation based subspace clustering; it describes each data point with the data collection itself on the original view directly and learnes diversity between multiple views by means of the Hilbert-Schmidt independence criterion (HSIC) to estimate the diversity across different representations. Later, a spectral clustering strategy is used to obtain

Table 1
Summary of the main notations.

Notation	Description	Notation	Description
n	Number of samples	$(\cdot)^T$	Transpose operator
c	Number of clusters	$\mathbf{I}$	Identity matrix
V	Number of views	$h(\cdot)$	Discrete hash function
m	Number of anchors	l	Binary code length
d_v	Data dimensionality for view v	$\mathbf{B} = [\mathbf{b}_1, \ldots, \mathbf{b}_n] \in \{-1, +1\}^{l \times n}$	The common binary codes of the n samples
$\mathbf{X}^1, \ldots., \mathbf{X}^M$ where $\mathbf{X}^v \in \mathbb{R}^{d_v \times n}$	A set of V data matrices	$\mathbf{U}^v \in \mathbb{R}^{l \times m}$	The mapping matrix for the vth view
$\mathbf{x}_s^v$	sth sample from the vth view	α	View-weighting vector
$\mathbf{a}_1^v, \mathbf{a}_2^v, \ldots, \mathbf{a}_m^v$	A set of selected anchors from the vth view	$\mathbf{W} \in \mathbb{R}^{n \times n}$	Sample-weighting matrix (a diagonal matrix)
$\mathbf{\Phi}^v \in \mathbb{R}^{m \times n}$	Nonlinear Radial Basis Function mapping for view v	$\beta, \gamma, \lambda, \rho$	Regularization parameters
σ	Kernel width	$\mathbf{C} \in \{-1, +1\}^{l \times c}$	Clustering binary centroids
$sgn(\cdot)$	Signum operator	$\mathbf{G} \in \{0, 1\}^{c \times n}$	Clustering assignment
$\|\| \cdot \|\|_F$	Frobenius norm	$\mathbf{1}$	Column vector of ones
$Tr(\cdot)$	Trace of a matrix		

the partitioning result. This method focuses on merging information rather than improving the ability to represent features.

- **AWP** [17]: It presented spectral embedding of kernels from different views in parallel with the Procrustes analysis technique to learn a unified cluster indicator matrix that fits all spectral embeddings. Despite the low computational cost compared to other graph- based methods, this work requires a post-processing step based on spectral rotation to obtain cluster labels.
- **WMSC** [12]: In this method, a normalized Laplacian matrix was created for each view, followed by a combined joint Laplacian matrix approximated based on the learned weights to discriminate the contribution of each corresponding view. Spectral clustering was performed to obtain the predicted labels, taking into account the principle of spectral perturbation, which aims to minimize the clustering ability between each selected view and the joint clustering. This method is based on the following principle: the proximity of the subspaces spanned by the eigenvectors is measured by the canonical angle between these subspaces to capture the difference in cluster ability.
- **OMSC** [18]: It was proposed to learn the affinity matrix for each view and the consensus graph in the intrinsic space simultaneously, thus assigning a projection matrix for each view to map the constructed affinities in a low-dimensional space. In addition, dynamic view weighting was provided to quantify the importance of each view. Without applying a clustering algorithm, an implicit partitioning result was generated by permuting the consensus affinity matrix into a new form in which it ensures the grouping of the data into a number of connected components based on the Laplacian rule and Ky Fan's theorem. This method is very sensitive to hyperparameters.
- **LMVSC** [19]: Multiple anchor graphs were created to reduce time complexity. Then, the double stochastic similarity matrix was computed, and the eigenvalue decomposition was performed on this small graph. K-means was applied to the embedded space to achieve the final clustering. In this work, the nonlinear high-order correlation between the consensus latent subspace and the different view-spaces is not well explored.
- **NESE** [20]: This model constructs graphs from different views and then considers a spectral embedding. These constructed view-based graph matrices and the spectral representation are simultaneously combined, inspired by a learning model of symmetric matrix factorization, to iteratively estimate a consistent non-negative embedding that directly reveals a joint partitioning result. Unlike AWP, this method estimates the clustering labels directly without post-processing. However, it suffers from the effects of noise and outliers since complementary information is merged into a non-negative embedding matrix.
- **GMC** [13]: It combines graph construction, graph fusion, and data clustering into a single framework, in which the graph of each view and the unified graph of all views are learned by mutual reinforcement, and the unified graph with a rank constraint directly partitions the data points into clusters. In this method, a weight is automatically assigned to each graph matrix to obtain a unified graph matrix.
- **SMVSC** [14]: In this method, a graph filtering technique was introduced to obtain a smooth representation. First, a graph was created for each view using the probabilistic neighborhood method. Then, a graph filter was applied to these graphs, followed by a selection of representative anchors. By concatenating different filtered graphs, a joint anchor graph fusion was created. Finally, an eigenvalue decomposition of this matrix was performed, and clustering was invoked using K-means. This is an alternative approach to LMVSC that uses graph filtering to achieve a smooth representation in each view.
- **Co-FW-MVFCM** [21]: It introduced a feature- and view-weighted scheme by integrating two steps: local and collaborative learning. The local step targeted the partitioning of each view and the collaborative step shared the information about the membership of multiple views. Finally, global clustering was achieved by aggregating the weighted partition matrices from different views. The problem with this model is that there are no clear criteria for selecting the optimal exponent parameter to control the view weights.

However, there are few studies that have looked at the clustering of large binary data. Gong et al. [22] have developed a method for binary clustering in a view that consists of two separate steps, namely binary code generation and binary k-means clustering. The main drawback is that the binary code is generated using a data-independent method, iterative quantization (ITQ). The work described in Gong et al. [9] adopted a two-level clustering breaks the link between binary representation and data partitioning. To accelerate large-scale clustering of single views, Shen et al. [23] combined binary structural SVM and conventional k-means in an optimization algorithm. Neither method can be applied to large-scale MVC, and the characteristics of multi-view data have not been thoroughly investigated. In the meantime, the binary codes generated byShen et al. [23] obtained unsatisfactory results due to the lack of a complete joint representation. Zhang et al. [7] have developed an interesting approach called Binary Multi-view Clustering (BMVC) to overcome a major problem related to multi-view clustering, which requires less computation time and storage cost.

BMVC has uncovered two essential elements: collaborative discrete representation learning and binary clustering structure learning in a common model. By considering only the complementary

features, this framework has considered encoding multi-view features into a common compact binary code. This model provides a non-negative normalized vector to weight the views with an additional adjustable parameter to balance the importance of the different views.

This method suffered from the proper distinction between shared and individual information, which can lead to the loss of local structure preservation in binary code learning. As an alternative working solution to the above problem, the HSIC method has jointly learned a common binary representation and robust discrete cluster structures [11]. The former decomposes each projection into a combination of shareable and individual projections across multiple views to capture the underlying correlations; the latter can greatly improve the computational efficiency and robustness of clustering. However, the above work is very sensitive to the initialization of the binary clustering process, and even the performance degrades when trying to get rid of the extra parameter and learn the weighting factor of each view automatically.

To address the above shortcomings, we drew inspiration from the BMVC framework. Our outstanding work, which falls into the category of multiview NMF clustering, characterizes the relationship between views based on samples using the learning strategy of automatic weighting of samples and automatic weighting of views. In the present work, the clustering was performed based on a joint binary matrix factorization over a bit balance constraint [9], which is a typical requirement of binary code learning. In particular, the initialization of the discrete representation plays a crucial role in driving the iterative binary clustering optimization towards the optimal point solution. Together with this concept of initialization, we developed an efficient solution that integrates a new deep feature of Vgg16. Finally, these features are encoded by a set of compact binary codes using the bidirectional FFT technique [24].

3. The proposed approach

In this section, we provide a detailed description of the new multi-view clustering method, which we call Auto-Weighted Binary Multi-View Clustering Via Deep Initialization (AW-BMVC). It consists of two common learning objectives: a common discrete representation driven by the auto-weighted sampling strategy and the auto-weighted view strategy; and at the same time, the global objective function is initialized with a good binary matrix representation. In short, Fig. 1 illustrates the diagram of the proposed framework.

3.1. Anchor-based representation

Considering an RBF (Radial Basis Function) that could evidently arrange different views into a single tensor with fixed dimensionality and well explore the high-order latent structure within multiple views by projecting them into a higher dimensional space.

We consider a multi-view dataset consists of V representations (i.e. V views) for n instances, which are designated by a set of matrices $\{\mathbf{X}^1, \ldots, \mathbf{X}^V\}$; where $\mathbf{X}^v \in \mathbb{R}^{d_v \times n}$, is the data matrix of the vth view, and d_v is the dimensionality of data features from the vth view. It is assumed that data samples in each view are zero-centered, i.e. $\sum_s \mathbf{x}_s^v = 0$, to maintain the balance of the data.

The first step is to encode data using non-linear RBF mapping. This encoding is given by the following mapping:

$$\mathbf{\Phi}(\mathbf{x}_s^v) = \left[exp\left(-\frac{||\mathbf{x}_s^v - \mathbf{a}_1^v||^2}{\sigma^v} \right), \ldots, exp\left(-\frac{||\mathbf{x}_s^v - \mathbf{a}_m^v||^2}{\sigma^v} \right) \right]^T \quad (1)$$

where σ^v is the kernel width for the vth view, $\mathbf{\Phi}(\mathbf{x}_s^v) \in \mathbb{R}^m$ represents m-dimensional non-linear embedding for the sth sample from the vth view, $\{\mathbf{a}_1^v, \mathbf{a}_2^v, \ldots, \mathbf{a}_m^v\}$ is a set of m selected anchors from vth view. One approach is to think of anchors as being statistically representative of that wider dataset. We get these anchors using the K-medoids technique, by leveraging its robustness to noise [25], rather than random sampling or K-means.

Remark: We fix the number of selected anchors for each view to $m = 1000$ based on the outlined experiments in Zheng et al. [7]. It should also be noted that the influence of the kernel width parameter σ^v is critical, as it determines the extent of smoothing [26] and often requires a lot of manual investigation. Empirically, a universal adaptive scaling is established where the global width for each view can be set to the average of all the Euclidean distances between the samples and their corresponding anchors.

3.2. Common discrete representation

The main goal of our unsupervised method is to perform direct clustering in much lower-dimensional Hamming space using the common binary codes. In particular, compression of multiple views is performed. To mitigate this, hashing has been introduced as a popular approach for a computationally efficient similarity preserving technique.

We consider a discriminative hashing function to be learned for each view in which we aim to quantize each $\mathbf{\Phi}(\mathbf{x}_s^v)$ into a discrete representation as follows:

$$\min_{\mathbf{U}^v, \mathbf{b}_s} \sum_{v=1}^{V} \sum_{s=1}^{n} ||\mathbf{b}_s - \mathbf{U}^v \mathbf{\Phi}(\mathbf{x}_s^v)||^2 = \min_{\mathbf{U}^v, \mathbf{B}} \sum_{v=1}^{V} ||\mathbf{B} - \mathbf{U}^v \mathbf{\Phi}(\mathbf{X}^v)||_F^2 \quad (2)$$

$$\mathbf{b}_s = h_s^v(\mathbf{\Phi}(\mathbf{x}_s^v); \mathbf{U}^v) = sgn(\mathbf{U}^v \mathbf{\Phi}(\mathbf{x}_s^v)) \quad (3)$$

where $\mathbf{B} = [\mathbf{b}_1, ..., \mathbf{b}_n]$ is the common binary codes from different views (i.e., $\mathbf{x}_s^v, \forall v = 1, ..., V$), $\mathbf{\Phi}(\mathbf{X}^v)$ is the matrix nonlinear representation of all samples in view v, $\mathbf{\Phi}(\mathbf{X}^v) = [\mathbf{\Phi}(\mathbf{x}_1^v), \ldots, \mathbf{\Phi}(\mathbf{x}_n^v)]$, $\mathbf{U}^v$ is the mapping matrix, $sgn(\cdot)$ is the element-wise sign operator. Note that although the model in Eq. (2) is linear, the overall mapping from data space to the common binary code space is nonlinear due to the use the nonlinear mapping $\mathbf{\Phi}(\mathbf{X}^v)$.

3.3. Sample-view auto-weighting

It is acknowledged that different views depict the same subject from various measurements hence, the projection $\{\mathbf{U}^v\}_{v=1}^V$ ought to capture consensus information that maximizes the similarities between different views, as well as the disparity that discriminates individual characteristics. Therefore, to characterize the relationship between views, implicit automatic view weighting will be adopted. On the other hand, explicit sample weighting coefficients will be estimated in global optimization. This strategy allows interchangeably highlighting the vital samples and promoting the complementary information between different views that yields a full common discrete representation.

To step further, from the information-theoretic point of view, it is required to maximize the information carried by each bit of the binary codes [27]. Based on this concept, an additional regularizer is adopted for the binary codes $\mathbf{B}$ using the maximum entropy principle [9]. Thus, our goal is to maximize the variance of the matrix $\mathbf{B}$ given by :

$$\begin{aligned} var[\mathbf{B}] &= \frac{1}{n} \sum_{v=1}^{V} var[\mathbf{U}^v \mathbf{\Phi}(\mathbf{X}^v)] = \frac{1}{n} \sum_{v=1}^{V} \|\mathbf{U}^v \mathbf{\Phi}(\mathbf{X}^v)\|^2 \\ &= \frac{1}{n} \sum_{v=1}^{V} tr((\mathbf{U}^v \mathbf{\Phi}(\mathbf{X}^v))(\mathbf{U}^v \mathbf{\Phi}(\mathbf{X}^v))^T) \end{aligned} \quad (4)$$

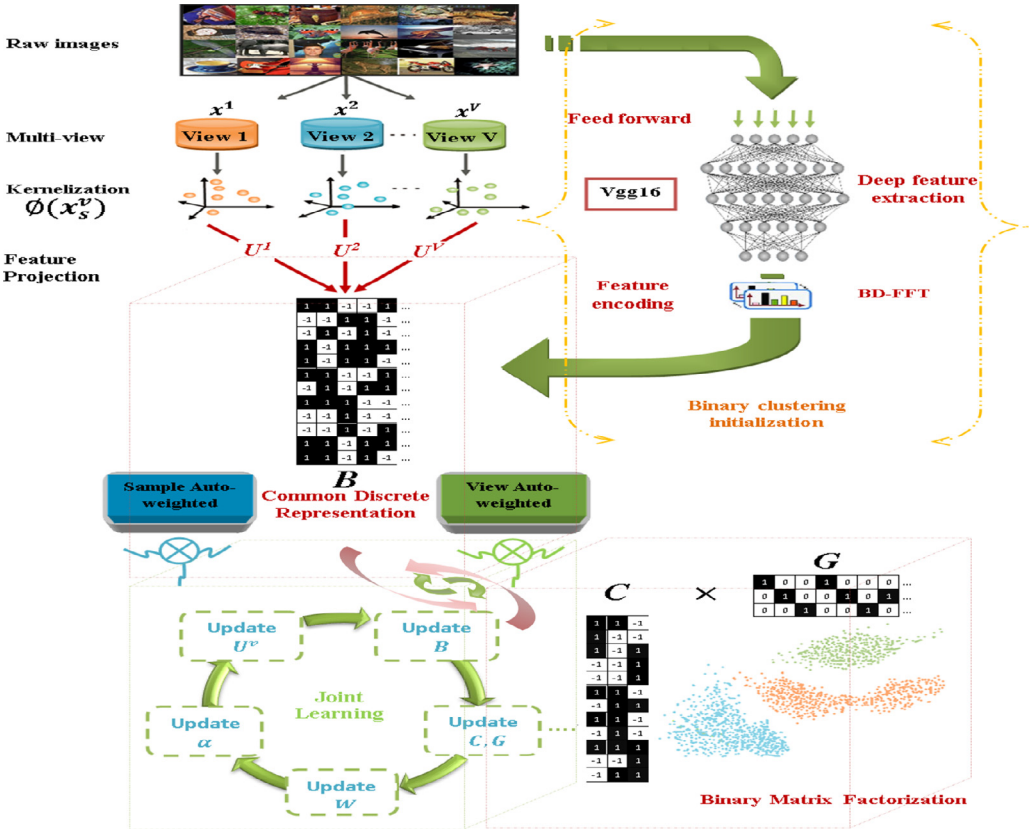

Fig. 1. The flowchart of the proposed method. Common discrete representation, Binary clustering initialization, Sample & view auto-weighting, and binary matrix factorization are integrated into a unified learning framework.

This additional regularization on $\mathbf{B}$ can ensure balanced partition and reduce the redundancy of the binary codes [11]. We formulate the relaxed regularization as a common discrete representation learning problem:

$$\min F\left(\mathbf{U}^v, \mathbf{B}, \mathbf{W}\right) = \sum_{v=1}^{V}\left(\left\|\left(\mathbf{B}-\mathbf{U}^v\mathbf{\Phi}\left(\mathbf{X}_s^v\right)\right)\mathbf{W}\right\|_F^2+\beta\left\|\mathbf{U}^v\right\|_F^2 - \frac{\gamma}{n}tr\left(\left(\mathbf{U}^v\mathbf{\Phi}\left(\mathbf{X}^v\right)\right)\left(\mathbf{U}^v\mathbf{\Phi}\left(\mathbf{X}^v\right)\right)^T\right)\right)$$
$$s.t.\ \mathbf{B}\in\{-1,1\}^{l\times n}, \sum_s w_s = 1, w_s > 0, \quad (5)$$

where β and γ are two regularization parameters.

The second term is a regularizer that controls the parameter scales (contribute to the stable solution). $\mathbf{W} = diag(w_1, w_2, \ldots, w_n)$ is the diagonal sample-weighting matrix. By learning the weights for samples, the important ones will get a large weight.

Motivated by recently proposed auto-weighted techniques [28], we propose the novel formulation where no view weight factors are explicitly defined.

Here we replace the above objective function with a new one that is the square root of the term to be minimized. As such, the problem can be reformulated as follows:

$$\min_{\mathbf{U}^v,\mathbf{B},\mathbf{W}} = \sum_{v=1}^{V}\sqrt{\left\|\mathbf{B}-\mathbf{U}^v\mathbf{\Phi}(\mathbf{X}^v)\mathbf{W}\right\|_F^2+\beta\left\|\mathbf{U}^v\right\|_F^2 - \frac{\gamma}{n}tr\left(\mathbf{U}^v\mathbf{\Phi}(\mathbf{X}^v)(\mathbf{U}^v\mathbf{\Phi}(\mathbf{X}^v))^T\right)}$$
$$s.t.\ \mathbf{B}\in\{-1,1\}^{l\times n}, \sum_s w_s = 1, w_s > 0 \quad (6)$$

As in many multi-view algorithms, this form of criterion will implicitly provide a weight for each view. Therefore, minimizing Eq. (6) is equivalent to minimizing the following:

$$\min_{\mathbf{U}^v,\mathbf{B},\mathbf{W}} = \sum_{v=1}^{V}\alpha^v\left(\left\|\left(\mathbf{B}-\mathbf{U}^v\mathbf{\Phi}\left(\mathbf{X}^v\right)\right)\mathbf{W}\right\|_F^2+\beta\left\|\mathbf{U}^v\right\|_F^2 - \frac{\gamma}{n}tr\left(\mathbf{U}^v\mathbf{\Phi}\left(\mathbf{X}^v\right)\right)\left(\mathbf{U}^v\mathbf{\Phi}\left(\mathbf{X}^v\right)\right)^T\right)$$
$$s.t.\ \mathbf{B}\in\{-1,1\}^{l\times n}, \sum_s w_s = 1, w_s > 0, \quad (7)$$

where the auto-weight α^v is given by the following expression:

$$\alpha^v = \frac{1}{2\sqrt{||(\mathbf{B}-\mathbf{U}^v\mathbf{\Phi}(\mathbf{X}^v))\mathbf{W}||_F^2+\beta||\mathbf{U}^v||_F^2 - \frac{\gamma}{n}tr(\mathbf{U}^v\mathbf{\Phi}(\mathbf{X}^v))(\mathbf{U}^v\mathbf{\Phi}(\mathbf{X}^v))^T}} \quad (8)$$

3.4. Binary matrix factorization and overall objective function

AW-BMVC considers the factorization of the learned discrete representation $\mathbf{B}$ directly into two matrices; the binary clustering centroids $\mathbf{C}$ and the discrete clustering indicators $\mathbf{G}$ with some specific constraints using:

$$\min_{\mathbf{C},\mathbf{g}_s}||\mathbf{b}_s-\mathbf{C}\mathbf{g}_s||_F^2$$
$$s.t.\ \mathbf{C}^T\mathbf{1}=0, \mathbf{C}\in\{-1,1\}^{l\times c}, \mathbf{g}_s\in\{0,1\}^c, \sum_i^c g_{is}=1 \quad (9)$$

where $\mathbf{C}$ and $\mathbf{g}_s$ are the clustering centroids and the assignment vector for the sample s, respectively. The clustering centers constraint ($\mathbf{C}^T\mathbf{1}=0$) grants the balance condition to maximize the information of each bit. Writing Eq. (9) for all samples, we got the following factorization problem:

$$\min_{\mathbf{C},\mathbf{G}}||(\mathbf{B}-\mathbf{C}\mathbf{G})\mathbf{W}||_F^2$$
$$s.t.\ \mathbf{C}^T\mathbf{1}=0, \mathbf{C}\in\{-1,1\}^{l\times c}, \mathbf{G}\in\{0,1\}^{c\times n}, \sum_{i=1}^{c} G_{is}=1 \quad (10)$$

So, the overall joint AW-BMVC is formulated as:

$$\min F\left(\mathbf{U}^{\nu}, \mathbf{B}, \mathbf{C}, \mathbf{G}, \mathbf{W}, \alpha\right) = \sum_{\nu=1}^{V} \alpha^{\nu}\Big[\left\|\left(\mathbf{B} - \mathbf{U}^{\nu}\mathbf{\Phi}\left(\mathbf{X}^{\nu}\right)\right)\mathbf{W}\right\|_F^2 + \beta\left\|\mathbf{U}^{\nu}\right\|_F^2 - \frac{\gamma}{n} tr\left(\left(\mathbf{U}^{\nu}\mathbf{\Phi}\left(\mathbf{X}^{\nu}\right)\right)\left(\mathbf{U}^{\nu}\mathbf{\Phi}\left(\mathbf{X}^{\mathbf{v}}\right)\right)^{T}\right)\Big] + \lambda\left\|(\mathbf{B} - \mathbf{C}\,\mathbf{G})\mathbf{W}\right\|_F^2$$

$$s.t.\ \mathbf{C}^{T}\mathbf{1} = 0, \quad \sum_{s} w_s = 1, w_s > 0,$$

$$\mathbf{B} \in \{-1, 1\}^{l\times n}, \mathbf{C} \in \{-1, 1\}^{l\times c}, \mathbf{G} \in \{0, 1\}^{c\times n}, \quad \sum_{i=1}^{c} G_{is} = 1, \tag{11}$$

where λ is the regularization parameter.

We should draw attention to the presence of sample auto-weighted matrix $\mathbf{W}$ for the binary clustering learning part, this would make sense about conserving information and upholding the inter-relationship equilibrium between the discrete representation and the binary clustering learning.

3.5. Optimization

Basically, the solution of the problem (11) is a challenging combinatorial optimization problem due to the discrete constraints and the nonlinearity of the objective function. Therefore, an alternating optimization scheme is applied to decompose the problem into small subproblems and update it alternately with respect to one variable while fixing the other remaining variables. Therefore, we define each step to iteratively update the mapping matrix $\mathbf{U}^{\nu}$, the discrete representation $\mathbf{B}$, the binary cluster centroids $\mathbf{C}$ and the indicator $\mathbf{G}$, the sample auto-weighting $\mathbf{W}$ and the view auto-weighting α^{ν}, respectively.

• **Step 1: Update $\mathbf{U}^{\nu}, \nu = 1, \ldots, V$.**

By fixing other variables, the optimization formula for $\mathbf{U}^{\nu}$ is

$$\min F\left(\mathbf{U}^{\nu}\right) = \left\|\left(\mathbf{B} - \mathbf{U}^{\nu}\mathbf{\Phi}\left(\mathbf{X}^{\nu}\right)\right)\mathbf{W}\right\|_F^2 + \beta\left\|\mathbf{U}^{\nu}\right\|_F^2 - \frac{\gamma}{n} tr\left(\left(\mathbf{U}^{\nu}\mathbf{\Phi}\left(\mathbf{X}^{\nu}\right)\right)\left(\mathbf{U}^{\nu}\mathbf{\Phi}\left(\mathbf{X}^{\nu}\right)\right)^{T}\right) \tag{12}$$

By computing the derivative of the objective function with respective to $\mathbf{U}^{\nu}$, and setting it to 0, we can obtain the following solution:

$$\mathbf{U}^{\nu} = \mathbf{B}\,\mathbf{W}\,\mathbf{W}\,\mathbf{\Phi}(\mathbf{X}^{\nu})^{T} \cdot \mathbf{Q} \tag{13}$$

where $\mathbf{Q} = \left[\mathbf{\Phi}(\mathbf{X}^{\nu})\mathbf{W}\mathbf{W}\mathbf{\Phi}(\mathbf{X}^{\nu})^{T} - \frac{\gamma}{n}\mathbf{\Phi}(\mathbf{X}^{\nu})\mathbf{\Phi}(\mathbf{X}^{\nu})^{T} + \beta\,\mathbf{I}\right]^{-1}$.

• Step 2: Update $\mathbf{B}$.

The optimization formula for $\mathbf{B}$ is

$$\begin{aligned}\min_{\mathbf{B}} &= \sum_{\nu=1}^{V}\alpha^{\nu}\left(\left\|\left(\mathbf{B} - \mathbf{U}^{\nu}\mathbf{\Phi}\left(\mathbf{X}^{\nu}\right)\right)\mathbf{W}\right\|_F^2\right) + \lambda\left\|(\mathbf{B} - \mathbf{C}\mathbf{G})\mathbf{W}\right\|_F^2 \\ &= \sum_{\nu=1}^{V}\alpha^{\nu}\,tr\left(\left(\mathbf{B}\mathbf{W} - \mathbf{U}^{\nu}\mathbf{\Phi}\left(\mathbf{X}^{\nu}\right)\mathbf{W}\right)^{T}\left(\mathbf{B}\mathbf{W} - \mathbf{U}^{\nu}\mathbf{\Phi}\left(\mathbf{X}^{\nu}\right)\mathbf{W}\right)\right) \\ &\quad + \lambda\,tr\left((\mathbf{B}\mathbf{W} - \mathbf{C}\mathbf{G}\mathbf{W})^{T}(\mathbf{B}\mathbf{W} - \mathbf{C}\mathbf{G}\mathbf{W})\right) \\ &= tr\left[\mathbf{B}^{T}\left(\sum_{\nu=1}^{V}\alpha^{\nu}\mathbf{W}\mathbf{W}^{T} + \lambda\,\mathbf{W}\mathbf{W}^{T}\right)\mathbf{B}\right] - 2\,tr\left[\mathbf{B}^{T}\left(\sum_{\nu=1}^{V}\alpha^{\nu}\mathbf{U}^{\nu}\mathbf{\Phi}\left(\mathbf{X}^{\nu}\right)\mathbf{W}\,\mathbf{W} + \lambda\,\mathbf{C}\mathbf{G}\mathbf{W}\mathbf{W}\right)\right] + \text{cons} \\ &\quad s.t.\ \mathbf{B} \in \{-1, 1\},\end{aligned} \tag{14}$$

where *cons* indicates a constant value w.r.t. $\mathbf{B}$.

The solution for $\mathbf{B}$ is given by:

$$\mathbf{B} = sgn\left(\sum_{\nu=1}^{V}\alpha^{\nu}\mathbf{U}^{\nu}\mathbf{\Phi}(\mathbf{X}^{\nu})\mathbf{W}\mathbf{W} + \lambda\,\mathbf{C}\mathbf{G}\mathbf{W}\mathbf{W}\right) \tag{15}$$

• Step 3: Update $\mathbf{C}$ and $\mathbf{G}$.

The regularized optimization formula for $\mathbf{C}$ and $\mathbf{G}$ taking into account the discrete constraints will be given by:

$$\begin{aligned}&\min F(\mathbf{C}, \mathbf{G}) = ||(\mathbf{B} - \mathbf{C}\mathbf{G})\mathbf{W}||_F^2 + \rho||\mathbf{C}^{T}\mathbf{1}||^2 \\ &s.t.\ \mathbf{C} \in \{-1, 1\}^{l\times c},\ \mathbf{G} \in \{0, 1\}^{c\times n}, \quad \sum_{i} g_{is} = 1,\end{aligned} \tag{16}$$

We iteratively optimize the cluster centroids following the adaptive discrete proximal linearized minimization (ADPLM) technique [10], by maintaining the discrete constraints during the optimization process.

Update $\mathbf{C}$.

With $\mathbf{G}$ fixed we have the following minimization problem:

$$\min F(\mathbf{C}) = -2tr\Big[(\mathbf{B}\mathbf{W})^{T}(\mathbf{C}\mathbf{G}\mathbf{W})\Big] + \rho||\mathbf{C}^{T}\mathbf{1}||^2 + \text{cons} \tag{17}$$

The derivative of the obtained functional with respect to $\mathbf{C}$ is given as follows:

$$\nabla F(\mathbf{C}) = -2\mathbf{B}\,\mathbf{W}\,(\mathbf{G}\,\mathbf{W})^{T} + 2\rho\,\mathbf{E}\,\mathbf{C} s.t.\ \mathbf{C} \in \{-1, 1\}^{l\times c}, \tag{18}$$

where $\nabla F(\mathbf{C})$ is the gradient of $F(\mathbf{C})$ and $\mathbf{E}$ is $l \times l$ square matrix of ones.

Based on the rule of ADPLM, we update $\mathbf{C}$ in the $p+1$th iteration by

$$\mathbf{C}^{p+1} = sgn\left(\mathbf{C}^{p} - \frac{1}{\mu}\nabla F(\mathbf{C}^{p})\right) \tag{19}$$

where $\frac{1}{\mu}$ is a step size. We set $\mu^{p} \in (L, 2L)$, where L is the Lipschitz constant.

Update $\mathbf{G}$.

$$\min F(\mathbf{G}) = ||(\mathbf{B} - \mathbf{C}\mathbf{G})\mathbf{W}||_F^2 \tag{20}$$

Every column in $\mathbf{G} \in \{0, 1\}^{c\times n}$ represents the hard cluster assignment for sample s (i.e., the vector $\mathbf{g}_s$). It is given by:

$$g_{is}^{p+1} = \begin{cases} 1 & i = \arg\min_k H(\mathbf{b}_s, \mathbf{c}_k^{p+1}) \\ 0 & \text{otherwise} \end{cases} \tag{21}$$

where $H(\mathbf{b}_s, \mathbf{c}_k)$ is the Hamming distance between the sth binary code $\mathbf{b}_s$ and the kth cluster centroid $\mathbf{c}_k$.

• Step 4: Update the Sample weighting matrix $\mathbf{W}$.

$\mathbf{W}$ is the diagonal sample weight matrix. It is initialized by $w_1 = \ldots = w_s = \ldots = w_n = \frac{1}{n}$. It is updated using the following:

$$\begin{aligned}\min F(\mathbf{W}) &= \sum_{\nu=1}^{V}\alpha^{\nu}(||(\mathbf{B} - \mathbf{U}^{\nu}\mathbf{\Phi}(\mathbf{X}^{\nu}))\mathbf{W}||_F^2) + \lambda||(\mathbf{B} - \mathbf{C}\mathbf{G})\mathbf{W}||_F^2 \\ &\quad s.t.\ \sum_{s=1}^{n} w_s = 1, w_s > 0,\end{aligned} \tag{22}$$

The loss function (22) is simplified by adopting the following intermediate matrices:

$\mathbf{P}^{\nu} = [\mathbf{p}_1^{\nu}, \ldots, \mathbf{p}_n^{\nu}] = \mathbf{B} - \mathbf{U}^{\nu}\mathbf{\Phi}(\mathbf{X}^{\nu})$

$\mathbf{M} = [\mathbf{m}_1, \ldots, \mathbf{m}_n] = \mathbf{B} - \mathbf{C}\,\mathbf{G}$

$$F(\mathbf{W}) = \sum_{\nu=1}^{V}\alpha^{\nu}\left(\sum_{s=1}^{n} w_s^2||\mathbf{p}_s^{\nu}||^2\right) + \lambda\sum_{s=1}^{n} w_s^2||\mathbf{m}_s||^2 - \varepsilon\left(\sum_{s=1}^{n} w_s - 1\right) \tag{23}$$

$$\frac{\partial F(\mathbf{W})}{\partial w_s} = 0 \Rightarrow \sum_{\nu=1}^{V}\alpha^{\nu} 2w_s\|\mathbf{p}_s^{\nu}\|^2 + 2\lambda w_s\|\mathbf{m}_s\|^2 - \varepsilon = 0 \tag{24}$$

$$\Rightarrow 2\,w_s\left[\sum_{v=1}^{V}\alpha^v\,||\mathbf{p}_s||^2+\lambda\,||\mathbf{m}_s^v||^2\right]=\varepsilon \tag{25}$$

$$\Rightarrow 2\,w_s\,A_s=\varepsilon \tag{26}$$

where $A_s=\sum_{v=1}^{V}\alpha^v\,\|\mathbf{p}_s^v\|^2+\lambda\,\|\mathbf{m}_s\|^2$

$$\Rightarrow w_s=\frac{\varepsilon}{2\,A_s} \tag{27}$$

$$\sum_{s=1}^{n}w_s=1\Rightarrow\varepsilon=\frac{1}{\sum_{s=1}^{n}\frac{1}{2A_s}} \tag{28}$$

$$\Rightarrow w_s=\frac{\frac{1}{\sum_{s=1}^{n}(\frac{1}{2A_s})}}{2\,A_s} \tag{29}$$

• Step 5: Update the View weight $\alpha^v, v=1,\ldots,n$.
These are initialized by $\alpha^v=\frac{1}{v}$, $\forall v=1,...,V$.
With fixed $\mathbf{U}^v$, $\mathbf{B}$, $\mathbf{W}$; α^v can be optimized using Eq. (8).
Algorithm 1 summarizes the proposed framework.

Algorithm 1: Auto-weighted binary ulti-view clustering via deep initialization (AW-BMVC).

Input: Multi-view data $\mathbf{X^v}\in\mathbb{R}^{\mathbf{d_v}\times\mathbf{n}}$, and Selected anchors $\mathbf{A^v}\in\mathbb{R}^{\mathbf{d_v}\times\mathbf{m}}$, $v=1,...,V$, Parameters β,γ,λ, # of clusters c, # of iterations r & t, Length of binary codes l.
Output: Binary representation **B**, Cluster centroid **C**, Cluster indicator **G**.
Initialization: Initialize view weights $\alpha^v=\frac{1}{V}$, Initialize sample weights $w_s=\frac{1}{n}$, Initialize binary representation **B** (see section (3.6)).
Compute anchor-based representation $\mathbf{\Phi}(\mathbf{X^v}),\mathbf{v}=\mathbf{1},...,\mathbf{V}$ using (1).
repeat
  Update $\mathbf{U^v}$ using (13).Update **B** using (15).
  repeat
    Update **C** using (19).
    Update **G** using (21).
  until *convergence or reach r iterations*;
  Update **W** using (29).Update α using (8).
until *convergence or reach t iterations*;

3.6. Binary clustering initialization

The solution to our iterative clustering problem depends heavily on the initial setup of the binary matrix to be factorized. The effect of an improper initialization is interpreted as the clustering algorithm getting stuck in a bad local minimum.

Various Convolutional Neural Networks (CNN) architectures have been found to perform better than innovative hand-crafted feature detectors in detecting object features [29]. Under this concept, we introduce new deep method called Bidirectional-Fast Fourier Transform "BD-FFT", which provides effective representative codes using Fourier decomposition [24].

We use the rich feature representation of a pre-trained Visual Geometry Group model VGG16; the first task is to forward our image dataset and retrieve features from the second FC layer (4096 neurons). Each of these neurons is sensitive to a particular feature [30]. The second task is to create a frequency domain representation as a sequence of sorted frequencies using bidirectional FFT. In this transformation, we treat each deep feature vector as a one-dimensional signal. Based on this idea, the coefficients corresponding to "l" low frequencies were selected and transformed into binary codes [24,31], with the threshold set to the mean of the frequency coefficients.

Note that the deep features are not used as an additional view in the proposed criterion(11), but only to obtain a good initialization of the matrix **B**.

Table 2
Datasets used in our experiments. "dim" refers to the feature dimension.

Dataset	#Samples	#Views	Feature descriptors	#Classes
Caltech101-7/20	1474/2386	6	48-dim Gabor features 40-dim Wavelet moments 254-dim Centrist features 1984-dim HOG 512-dim GIST 928-dim LBP	7/20
NUSWIDE-Obj	30,000	5	65-dim Color Histogram 226-dim Color moments 145-dim Color correlation 74-dim Edge distribution 129-dim Wavelet texture	31
Scene-15	4485	3	20-dim GIST 59-dim PHOG 40-dim LBP	15

4. Performance analysis

4.1. Experimental setup

4.1.1. Datasets

We perform experiments on four public multiview image datasets commonly used to benchmark clustering algorithms, including **Caltech101-7, Caltech101-20**[1][32], **NUSWIDE-Obj**[2] [33], and **Scene-15**[3] [34]. Multi-view features are extracted to describe each image. Table 2 exhibits a detailed description of these datasets. Caltech101 contains 9144 images grouped into 101 objects. By tracking the earlier work in Wang et al. [28], we select the frequently used object recognition dataset with 7 categories. 1474 images from the data are assembled to produce the so-called Caltech101-7. In addition, 2386 images affiliated with 20 classes are selected. This dataset is called Caltech101-20. Six different features were selected for both Caltech101-7 and Caltech101-20.

NUSWIDE-Obj includes 30,000 images distributed amongst 31 classes. Five popular descriptors are used for this dataset.

Scene-15, formed by 4485 images grouped into 15 categories of indoor and outdoor scenes. Features are extracted from each image to form three views.

4.1.2. Evaluation metrics and competitors

We validated the proposed approach using the three most commonly used external evaluation criteria [35]: Accuracy (ACC), Normalized Mutual Information (NMI), and purity. We support our proposal by comparing it with eleven state-of-the-art algorithms which are precisely described in the related work section (2.2): **(RMSC)** [15], **(DiMSC)** [16], **(AWP)** [17], **(WMSC)** [12], **(BMVC)** [7], **(OMSC)** [18], **(LMVSC)** [19], **(NESE)** [20], **(GMC)** [13], **(SMVSC)** [14], **(Co- FW-MVFCM)** [21]. We run the compared algorithms based on the prescribed optimal parameter setting of each work.

4.2. Parameter sensitivity

The proposed model is parameterized so that its behavior can be tuned with three hyperparameters: β,γ, and λ. These regularization parameters are expected to contribute to a stable solution.

[1] https://data.caltech.edu/records/20086.
[2] https://lms.comp.nus.edu.sg/wp-content/uploads/2019/research/nuswide/NUS-WIDE.html.
[3] https://figshare.com/articles/dataset/15-Scene_Image_Dataset/7007177.

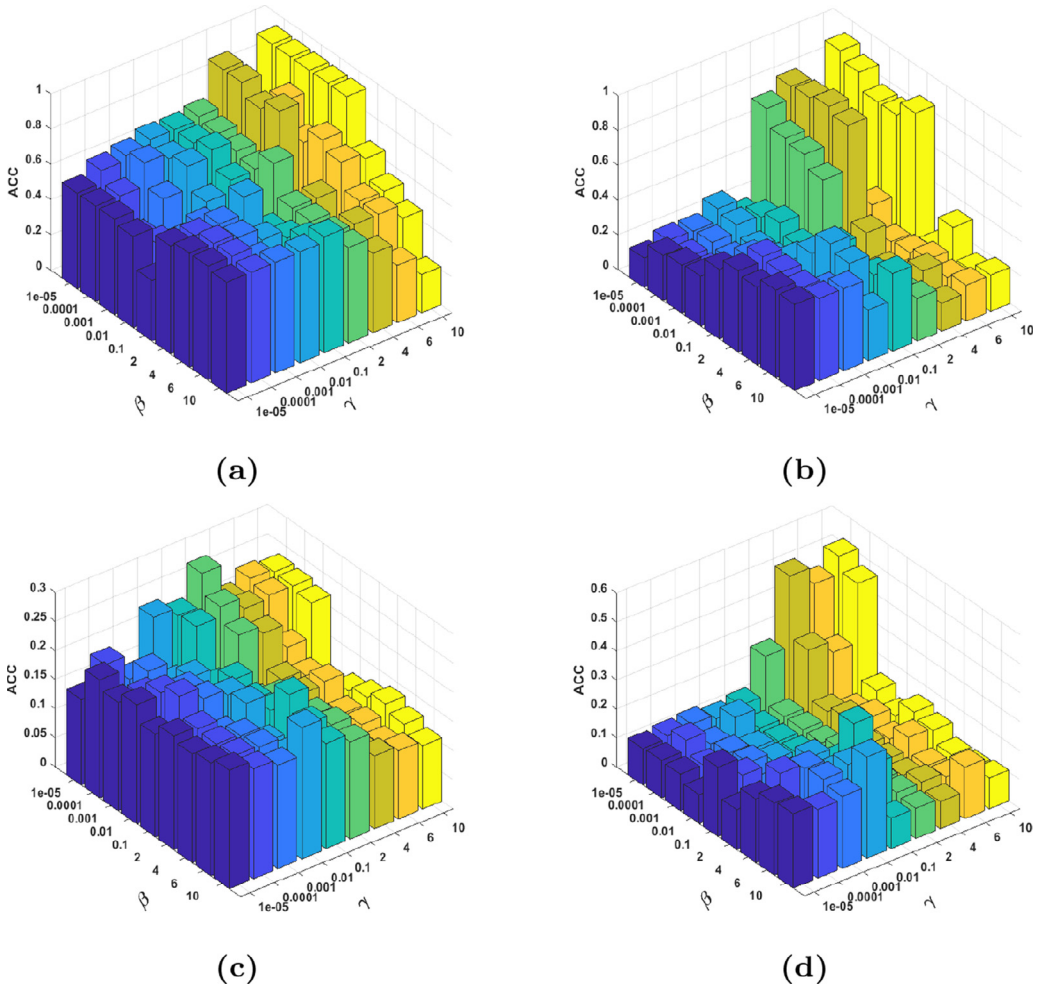

Fig. 2. Variability of accuracy with respect to β and γ parameters on: (a) Caltech101-7, (b) Caltech101-20, (c) NUSWIDE-Obj, (d) Scene-15.

Table 3
Best parameter tuning.

Datasets	β	γ	λ
Caltech101-7(20)	1e−05	10	1e−09
NUSWIDE-Obj	1e−05	2	1e−09
Scene-15	1e−05	10	1e−09

We analyzed the effects of these parameters by setting λ to 1e−9 and empirically varying the values of β and γ from the grid {1e−5, 1e−4, 1e−3, 1e−2 1e−1 2, 4, 6, 10}.

The variability of clustering accuracy for the four datasets and for different configurations of β and γ is shown in Fig. 2.

We should mention here that the sensitivity across the Caltech101-7/20 datasets is also dependent on the number of selected anchors, which is recommended to be less than 1000 anchors; this comes from the numerical perturbation, that will be addressed in the convergence analysis (section (4.6) P.30). We obtain excellent clustering performance when working with low values of β ($\beta = 1e-5$) and relatively high values of γ ($\gamma = 10$). Clustering performance is relatively stable when $1e-5 < \beta < 1e-2$; $2 < \gamma < 10$; otherwise, we run the risk of losing effectiveness outside the optimal range.

The summary of the best parameter setting for the three parameters is given in Table 3, which shows that despite the sensitivity mentioned above, we obtained very good clustering results for all tested datasets with only one tuning of γ in a small search range.

Table 4
The running time (seconds) of different clustering approaches on the Caltech101-7 dataset.

Method	Time (s)	Method	Time (s)
RMSC-2014	92.08	**LMVSC-2020**	135.79
DiMSC-2015	355.77	**NESE-2020**	63.18
AWP-2018	7.76	**GMC-2020**	92.81
WMSC-2018	7.73	**SMVSC-2021**	236.32
BMVC-2018	**6.18**	**Co-FW-MVFCM-2021**	1864.57
OMSC-2019	107.55	**AW-BMVC(Ours)**	15.23

4.3. Computational complexity

In the proposed work, the problem of binary code learning was addressed as an interesting research approach to solve the problem of large-scale clustering of multiple views. The total complexity of AW-BMVC is $O(nlm^2V)t$ when five optimization operations are considered: $\mathbf{U}^v$, $\mathbf{B}$, $\mathbf{C}$, $\mathbf{G}$, $\mathbf{W}$, and α,. This calculation goes beyond the complexity of deep feature extraction, which is a part of the preprocessing step.

It can be observed that $l \ll n$ and $m \ll n$ where n is the number of data samples, and the small number of iterations "t", since the proposed model converges rapidly; consequently, the time complexity can be summarized to $O(n)$, which depends linearly on n.

We fulfill our running time experiments using Matlab R2019b on PC machine with 2.39 GHz, i-5-2430M CPU and 6 GB RAM memory. In Table 4, we give the running time of the different methods on Caltech101-7.

Table 5
Ablation experimental results. SAW: Sample Auto-Weighted. VAW: View Auto-Weighted. BCI: Binary Clustering Initialization.

	Removing or adding a component			Dataset			
	SAW	VAW	BCI	Caltech101-7	Caltech101-20	NUSWIDE-Obj	Scene-15
ACC	✗	✗	✗	0.2856	0.2355	0.1680	0.2312
NMI				0.1079	0.1864	0.1621	0.1466
Purity				0.5916	0.4392	0.2872	0.2580
ACC	✔	✗	✗	0.2904	0.2921	0.1875	0.1739
NMI				0.1645	0.2149	0.1082	0.1062
Purity				0.6832	0.4715	0.2383	0.1835
ACC	✗	✔	✗	0.3209	0.3814	0.1336	0.2446
NMI				0.2065	0.4932	0.1457	0.2062
Purity				0.7123	0.7318	0.2721	0.2999
ACC	✗	✗	✔	0.5122	0.5159	0.1874	0.5032
NMI				0.4935	0.6830	0.1935	0.43.22
Purity				0.8718	0.8688	0.3081	0.55.74
ACC	✔	✔	✗	0.3141	0.2200	0.1695	0.2881
NMI				0.1583	0.1898	0.1676	0.2080
Purity				0.6784	0.4484	0.2714	0.3097
ACC	✔	✗	✔	0.4274	0.6144	0.1598	0.433
NMI				0.2746	0.4900	0.1552	0.3986
Purity				0.7822	0.6174	0.2811	0.4384
ACC	✗	✔	✔	0.5102	0.4736	0.1853	0.4932
NMI				0.4931	0.6659	0.1957	0.3564
Purity				0.8718	0.8395	0.3137	0.54.62
ACC	✔	✔	✔	**0.9022**	**0.8734**	**0.2190**	**0.5634**
NMI				**0.8733**	**0.8180**	**0.2156**	**0.5089**
Purity				**0.9022**	**0.8873**	**0.322**	**0.5884**

It can be seen that DiMSC (355.77 s), OMSC (107.55 s), LMVSC (135.79 s), SMVSC (236.32 s) and Co-FW-MVFCM (1864.57 s) are relatively time-consuming methods (more than 100 s) compared to other approaches. As you can see from this table, our method achieves clustering results within 15.23 s in only $t = 3$ iterations for the Caltech101-7 dataset, which is due to the self-weighted term of the sample that slightly increases the time cost. The AWP, WMSC and BMVC methods show better time efficiency, but our approach achieves the best clustering results.

4.4. Ablation study

Our proposed method consists of three core modules: (Automatic sample weighting; Automatic view weighting; and Initialization of binary clustering). In Table 5, we report the performance on two tested datasets when each module is removed or added.

Note that the combination of these three indispensable components leads to the all-out proposal, whose dominance is evidenced by the bold results, while the removal of these components leads us back to the BMVC framework [7] (the first row in Table 5).

We note that merging the two variants sample and view auto-weighted without considering the binary clustering initialization part can result in a massive performance drop; accordingly, the BCI module plays an important role in improving the clustering performance. Furthermore, attempting to separate either auto-weighted component from the binary clustering initialization variant reduces the capacity of the model.

4.5. Clustering initialization analysis

To further investigate the impact of different initialization scenarios of the binary codes on our proposed optimization algorithm, we conducted a group of experiments on all datasets. Algorithm 1 was used with three initialization scenarios for the matrix of binary codes **B**: A random binary matrix, non-linear PCA, and Deep-FFT. In the first scenario, we generate a random binary matrix; the second scenario shows a non-linear PCA technique adopted by the BMVC method[7]; the third technique refers to Deep-FFT, which is our proposed initialization scenario. As can be seen in Table 6, the obvious worst results are obtained by the binary random matrix, which is completely independent of the data. The non-linear PCA method exposes homogeneous but sub-optimal scoring metrics. We conclude that this is due to performing the eigenvalue decomposition over an embedded view. In particular, compared to the previous two scenarios, we can draw a conclusion about our initialization approach that achieves a significant improvement in clustering results as we consider a new deep feature extraction that explicitly improves the optimization process.

4.6. Convergence analysis

Figure 3 shows the objective function value for each iteration on four datasets. The alternating iterative optimization strategy is used to iteratively update each variable: the mapping matrix $\mathbf{U}^v$, the discrete representation **B**, the binary cluster centroids **C**, the cluster indicator matrix **G**, the auto-weighted sample **W**, and the auto-weighted view α. The subproblems $\mathbf{U}^v$ and **B** arising from Eqs. (12) to (14) guarantee a closed form optimal solutions given by Eqs. (13) and (15), respectively. The subproblem **C** in Eq. (17) has an analytical solution using ADPLM [10], Eq. (20) effectively shows its optimal solution, followed by the obvious solution for **G** in Eq. (21), which is similar to the K-means learning scheme. The solution for the automatic weighting of samples in Eq. (29) as well as the automatic weighting of views in Eq. (8) are the exact minimum points. As a result, the loss values of the globally adopted objective function $F(\mathbf{U}; \mathbf{B}; \mathbf{C}; \mathbf{G}; \mathbf{W}; \alpha)$ in Eq. (11) decrease rapidly and reach the minimum point after about $t = 5$ iterations, along with verifying the monotonic bound, which is a sufficient condition for convergence.

Numerical perturbation on Caltech101-7/20 Another phenomenon that may reveal the stability dilemma occurred when we experimented with the Caltech101-7/20 datasets in particular. A numerical perturbation occurred that resulted in a rapid and transient drop in the objective function to its minimum value. Subsequently, a sudden rise and/or stall was observed as the computation of the $\mathbf{U}^v$ mapping matrices became ill-conditioned. For a small dataset such as Caltech-7 ($n = 1474$), the number of anchors may be upper bounded. Therefore, an experimental extension is achieved and

Table 6
Clustering initialization study. RI: Random Initialization. PCA: One-view PCA Initialization. Deep: Deep-FFT Initialization.

	Variant											
	RI	PCA	Deep	RI	PCA	Deep	RI	PCA	Deep	RI	PCA	Deep
Dataset	Caltech-7			Caltech-20			NUSWIDE-Obj			Scene-15		
ACC	0.2863	0.2924	**0.9022**	0.2393	0.2200	**0.8734**	0.1508	0.1956	**0.2190**	0.1445	0.2453	**0.5634**
NMI	0.0622	0.2103	**0.8733**	0.1471	0.1898	**0.8180**	0.0785	0.1500	**0.2156**	0.0476	0.1536	**0.5089**
Purity	0.5733	0.6934	**0.9022**	0.4510	0.4484	**0.8873**	0.2041	0.2634	**0.3220**	0.1521	0.2660	**0.5884**

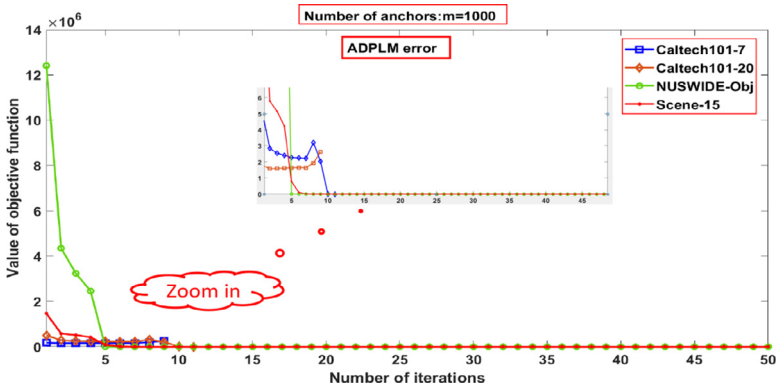

Fig. 3. Objective function as a function of iteration number on all datasets. The number of anchors m is set to 1000.

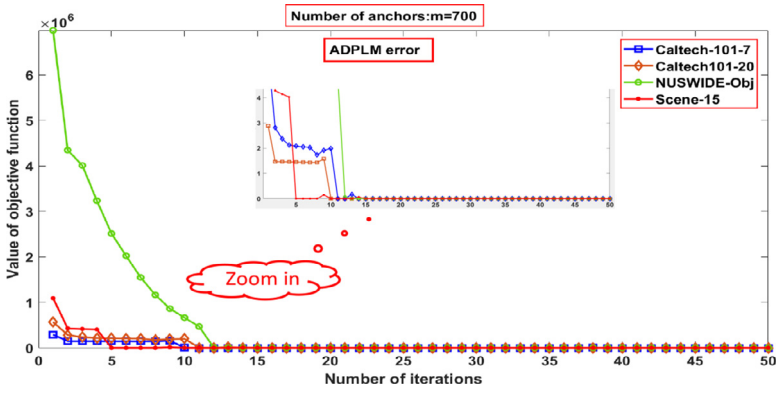

Fig. 4. Objective function as a function of iteration number on all datasets. The number of anchors m is set to 700.

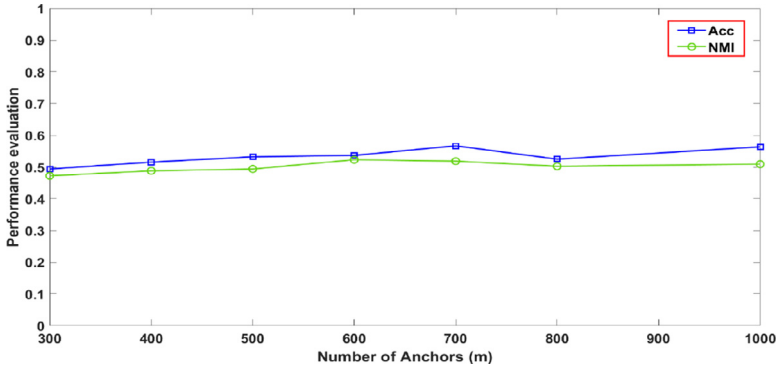

Fig. 5. ACC and NMI variation versus the number of anchors on the Scene-15 dataset.

the numerical perturbation problem is solved by simply reducing the number of selected anchors to less than 1000 anchors, where $m = 700$ anchors are experimented and validated (see Fig. 4).

Figure 5 illustrates the clustering performance as a function of the number of anchors using the Scene-15 dataset. As can be seen, the clustering performance can be affected by this number. The accuracy varies between 0.4939 ($m = 400$) and 0.5666 ($m = 700$). There are no specific criteria for determining the optimal number of anchors, but we can admit that this depends strongly on the number of samples. As an example for Scene-15, if the number of

Table 7
The clustering performance comparisons on challenging datasets."–" indicates unavailable results due to out of memory.

Methods	Caltech101-7			Caltech101-20			NUSWIDE-0bj			Scene-15		
	ACC	NMI	Purity	ACC	NMI	Purity	ACC	NMI	Purity	ACC	NMI	Purity
RMSC-2014 [15]	0.4037	0.3544	0.8026	0.4035	0.5073	0.7360	0.1473	0.1421	0.2624	0.3482	0.3483	0.3797
DiMSC-2015 [16]	0.5611	0.4221	0.8318	0.4728	0.4935	0.7347	0.1330	0.1363	0.2165	0.2555	0.2083	0.2758
AWP-2018 [17]	0.5685	0.4710	0.8554	0.4953	0.559	0.7594	0.1440	0.1123	0.2446	0.3429	0.3366	0.4035
WMSC-2018 [12]	0.5943	0.496	0.8588	0.5310	0.5893	0.7682	0.1382	0.1344	0.2475	0.4370	0.4341	0.4807
BMVC-2018 [7]	0.2856	0.1079	0.5916	0.2355	0.1864	0.4392	0.1680	0.1621	0.2872	0.2312	0.1466	0.258
OMSC-2019 [18]	0.0257	0.1770	0.9545	0.0255	0.3108	0.9241	0.0678	0.2530	0.4465	0.0084	0.3133	0.8403
LMVSC-2020 [19]	0.7266	0.5193	0.7517	0.5306	0.5271	0.5847	0.1181	0.1063	0.1363	0.3134	0.3297	0.3551
NESE-2020 [20]	0.4857	0.4614	0.8548	0.6085	0.6045	0.7556	–	–	–	0.4312	0.4042	0.4822
GMC-2020 [13]	0.6919	0.6056	0.8846	0.4564	0.3845	0.5549	0.1192	0.1128	0.1205	0.1400	0.1105	0.1464
SMVSC-2021 [14]	0.7354	0.5204	0.8487	0.5692	0.5190	0.6442	0.1254	0.1123	0.1587	0.3583	0.3433	0.3861
Co-FW-MVFCM-2021 [21]	0.4016	0.2819	0.7944	0.3051	0.3887	0.5746	0.1673	0.0913	0.2209	0.2856	0.2822	0.3257
AW-BMVC(Ours)	0.9022	0.8733	0.9022	0.8734	0.8180	0.8873	0.2190	0.2156	0.3220	0.5634	0.5089	0.5884

anchors is less than 500 a degradation in clustering performance is observed. However, when m is greater than 500, the performance peaks and becomes stable.

4.7. Comparison with state-of-the-art multi-view methods

To validate the superiority of the proposed algorithm, we performed extensive experiments with 11 state-of-the-art comparison methods. Table 7 shows the performance of all the competing clustering methods for the four datasets. In this table, the best clustering performance is highlighted in bold.

According to Table 7, the OMSC method shows unbalanced performance, noticeable at extremely low ACC, in contrast to the superior NMI and Purity for all datasets; this approach may require special adjustment.

Based on the results depicted in Tables 4 and 7, we can draw the following observations. Analytically, the SMVSC and LMVSC methods require quite a long runtime, but have the second best results. This is due to the smooth representation task achieved by the graph filtering in SMVSC and the anchor graph technique in LMVSC. The three methods WMSC, AWP and DiMSC achieve the third best results respectively. The first approach features lower running time and reveals the trick of minimizing the clustering ability between two groups of eigenvectors, the Laplacian for each view and the Laplacian of the consensus matrix. The second approach uses the Procrustes analysis technique to achieve the clustering assignment and avoids the eigenvalue decomposition in each iteration step, which makes it more efficient. The third method is the second most time consuming because it has the property of self-expression for each view in the original space. Moreover, it may require expanding the fusibility study towards a full diversity estimation. The NESE method is quite time efficient and gives good results even for small data sets. It takes advantage of consistent non-negative embedding, but needs to better address the problem of diversity of multiple views by specifying the degree of contribution of each view. BMVC is computationally very efficient thanks to its simultaneous binary representation and binary clustering. Three major weaknesses of this framework have been effectively tackled by our proposed approach (automatic view weighting, automatic sample weighting, and binary clustering initialization). RMSC performs poorly due to the two separate steps of consensus graph learning and clustering structure learning. The worst performing method is Co-FW-MVFCM, which has a very long runtime due to the a priori partitioning of the individual views and the shared information between the individual members. The technical treatment of feature reduction by thresholding each view is inadequate, as is the empirical exponent parameter to handle the distribution of each view.

In terms of general clustering metrics, most baselines perform better than other competing methods for a given dataset. For example, BMVC is better for large datasets such as NUSWIDE-Obj. WMSC is better for the Scene-15 dataset, NESE for Caltech101-20, and GMC for Caltech101-7. We can deduce that AW-BMVC achieves superior performance on the four benchmark image datasets: Caltech101-7, Caltech101-20, NUS-WIDE-obj, and Scene-15 in three evaluation indices and outperforms the results of the other methods.

5. Conclusion

In this work, we introduced a large-scale method called Auto-Weighted Binary Multi-View Clustering Via Deep Initialization (AW-BMVC) to learn a common discrete representation of multi-view data while optimizing binary clustering based on matrix factorization. Thanks to the advantages of self-weighted samples and views as the first component in this system, which demonstrates its ability to discriminate between views based on important samples and obtain a complete joint discrete representation. We have also placed great emphasis on a new deep representation technique to address the clustering initialization problem. As a result, our binary clustering initialization strategy proved to be positive for final clustering with excellent performance. Accordingly, fast convergence was achieved within a few iterations. Empirical results on several well-known datasets have confirmed the superiority of our approach over considerable state-of-the-art multi-view clustering methods.

However, for the scientific integrity of the proposed approach three unavoidable accompanying flaws must be mentioned: (1) It is pretty obvious as long as the preferred number of anchors is fixed to 1000 samples, the thing that makes the model selective as it doesnt deal with datasets any less. (2) Despite the efforts to make the model autonomously learn view and sample weights, we still have the weary manually tunable regularization parameters(β,γ,λ) for its stability, on which the clustering performance is highly dependent. (3) The choice of pre-trained deep Vgg16 as a part of binary matrix initialization technique restricts the scope of evaluation for handling only image datasets. Extending our work to text datasets through variants of viable binary clustering initialization methods is considered promising and it serves as a solution to one of the most important mentioned weaknesses.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

Acknowledgments

The authors gratefully acknowledge the Directorate General for Scientific Research and Technological Development (DGRSDT) of Algeria for the financial support to this work. This work is part of the grant PID2021-126701OB-I00 funded by MCIN/AEI/10.13039/501100011033 and by ERDF A way of making Europe.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.patcog.2022.109281.

References

[1] T. Hong, H. Chenping, et al., Multiview classification with cohesion and diversity, IEEE Trans. Cybern. 50 (5) (2018) 2124–2137.
[2] C. Guoqing, S. Shiliang, et al., A survey on multiview clustering, IEEE Trans. Artif. Intell. 2 (2) (2021) 146–168.
[3] P. Van, N. Pham, et al., Multi-view clustering and multi-view models, in: Recent Advancements in Multi-View Data Analytics, Springer, 2022, pp. 55–96.
[4] C. Kamalika, K. Sham, et al., Multi-view clustering via canonical correlation analysis, in: Proceedings of the 26th Annual International Conference on Machine Learning, 2009, pp. 129–136.
[5] L. Yeqing, N. Feiping, et al., Large-scale multi-view spectral clustering via bipartite graph, in: Twenty-Ninth AAAI Conference On Artificial Intelligence, 2015.
[6] S. El Hajjar, F. Dornaika, et al., Consensus graph and spectral representation for one-step multi-view kernel based clustering, Knowledge-Based Syst. 241 (2022) 108250.
[7] Z. Zheng, L. Li, et al., Binary multi-view clustering, IEEE Trans. Pattern Anal. Mach. Intell. 41 (7) (2018) 1774–1782.
[8] J. Wang, Z. Ting, et al., A survey on learning to hash, IEEE Trans. Pattern Anal. Mach. Intell. 40 (4) (2017) 769–790.
[9] Y. Gong, L. Svetlana, et al., Iterative quantization: a procrustean approach to learning binary codes for large-scale image retrieval, IEEE Trans. Pattern Anal. Mach. Intell. 35 (12) (2012) 2916–2929.
[10] S. Fumin, Z. Xiang, et al., A fast optimization method for general binary code learning, IEEE Trans. Image Process. 25 (12) (2016) 5610–5621.
[11] Z. Zhang, L. Liu, et al., Highly-economized multi-view binary compression for scalable image clustering (2018) pp. 717–732.
[12] L. Zong, X. Zhang, et al., Weighted multi-view spectral clustering based on spectral perturbation, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, 2018, doi:10.1609/aaai.v32i1.11625.
[13] W. Hao, Y. Yan, et al., GMC: graph-based multi-view clustering, IEEE Trans. Knowl. Data Eng. 32 (6) (2019) 1116–1129.
[14] C. Peng, L. Liang, et al., Smoothed multi-view subspace clustering, in: Neural Computing for Advanced Applications, Springer Singapore, Singapore, 2021, pp. 128–140.
[15] X. Rongkai, P. Yan, et al., Robust multi-view spectral clustering via low-rank and sparse decomposition, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 28, 2014, doi:10.1609/aaai.v28i1.8950.
[16] C. Xiaochun, Z. Changqing, et al., Diversity-induced multi-view subspace clustering, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 586–594.
[17] N. Feiping, T. Lai, et al., Multiview clustering via adaptively weighted procrustes, in: Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, in: KDD '18, Association for Computing Machinery, New York, NY, USA, 2018, pp. 2022–2030.
[18] Z. Xiaofeng, Z. Shichao, et al., One-step multi-view spectral clustering, IEEE Trans. Knowl. Data Eng. 31 (10) (2018) 2022–2034.
[19] K. Zhao, Z. Wangtao, et al., Large-scale multi-view subspace clustering in linear time, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 34, 2020, pp. 4412–4419, doi:10.1609/aaai.v34i04.5867.
[20] H. Zhanxuan, N. Feiping, et al., Multi-view spectral clustering via integrating nonnegative embedding and spectral embedding, Inf. Fusion 55 (2020) 251–259.
[21] Y. Miin-Shen, S. Kristina, Collaborative feature-weighted multi-view fuzzy *c*-means clustering, Pattern Recognit. 119 (2021) 108064.
[22] Y.Gong, P. Marcin, et al., Web scale photo hash clustering on a single machine, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2015, pp. 19–27.
[23] X. Shen, L. Weiwei, et al., Compressed *k*-means for large-scale clustering, in: Thirty-first AAAI Conference on Artificial Intelligence, 2017.
[24] A. Jamil, M. Khan, et al., Efficient conversion of deep features to compact binary codes using fourier decomposition for multimedia big data, IEEE Trans. Ind. Inf. 14 (7) (2018) 3205–3215.
[25] B. Aruna, K-medoids clustering using partitioning around medoids for performing face recognition, Int. J. Soft Comput., Math. Control 3 (3) (2014) 1–12.
[26] S. Weglarczyk, Kernel density estimation and its application, in: ITM Web Conf., vol. 23, 2018, p. 00037, doi:10.1051/itmconf/20182300037.
[27] J. Wang, S. Kumar, et al., Semi-supervised hashing for scalable image retrieval, in: 2010 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2010, pp. 3424–3431.
[28] B. Wang, Y. Xiao, et al., Robust self-weighted multi-view projection clustering, in: Proceedings of the AAAI Conference on Artificial Intelligence, 2020, pp. 6110–6117.
[29] A. Laith, Z. Jinglan, et al., Review of deep learning: concepts, CNN architectures, challenges, applications, future directions, J. Big Data 8 (1) (2021) 1–74.
[30] A. Valdez, P. Megan, et al., Distributed representation of visual objects by single neurons in the human brain, J. Neurosci. 35 (13) (2015) 5180–5186.
[31] A. Jamil, M. Khan, et al., Medical image retrieval with compact binary codes generated in frequency domain using highly reactive convolutional features, J. Med. Syst. 42 (2) (2018) 1–19.
[32] L. Fei-Fei, R. Fergus, et al., Learning generative visual models from few training examples: an incremental Bayesian approach tested on 101 object categories, in: 2004 Conference on Computer Vision and Pattern Recognition Workshop, 2004, p. 178, doi:10.1109/CVPR.2004.383.
[33] C. Tat-Seng, T. Jinhui, et al., NUS-WIDE: a real-world web image database from national university of Singapore, in: Proceedings of the ACM International Conference on Image and Video Retrieval, in: CIVR '09, Association for Computing Machinery, New York, NY, USA, 2009, pp. 1–9.
[34] S.Lazebnik, C. Schmid, et al., Beyond bags of features: Spatial pyramid matching for recognizing natural scene categories, in: 2006 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'06), vol. 2, 2006, pp. 2169–2178, doi:10.1109/CVPR.2006.68.
[35] N. Liang, Z.Yang, et al., Multi-view clustering by non-negative matrix factorization with co-orthogonal constraints, Knowledge-Based Syst 194 (2020) 105582.

Khamis Houfar received his B.E. and M.Sc. degrees from Kasdi Merbah University in Ouargla, Algeria, in 2008 and 2017, respectively, and is now a Ph.D. student in the Department of Electronics and Telecommunications at the same university. His research interests include computer vision, pattern recognition, data mining and machine learning.

Djamel Samai is an assistant professor in the Department of Electronics and Telecommunications at the University of Ouargla, Algeria. He received his Ph.D. in signal processing from the University of Annaba, Algeria. Dr. Samai's research interests include signal and image processing, image and video compression, pattern recognition, and biometrics.

Fadi Dornaika received his engineer degree in Electrical Engineering from the Lebanese University, in 1990, an M.S. degree in signal, image and speech processing from Grenoble Institute of Technology, France, in 1992, and a Ph.D. degree in computer science from Grenoble Institute of Technology, France and INRIA, in 1995. He is currently a Research Professor at IKERBASQUE (Basque Foundation for Science) and the University of the Basque Country. Prior to joining IKERBASQUE, he held numerous research positions in Europe, China, and Canada. He has published more than 350 papers in the field of computer vision and pattern recognition. His research covers a wide range of topics in computer vision. His current research interests include machine learning and pattern recognition.

Azeddine Benlamoudi received his B.S. and M.Sc. degrees from Mohamed Khider University of Biskra, Algeria, in 2010 and 2012, respectively. He earned his Ph.D. in 2018 in Electronics - Communications and Signal Processing from Ouargla University, where he has been an associate professor in the Department of Electronics and Telecommunications since 2018. His research interests include computer vision, pattern recognition, signal and image processing, biometrics, and spoofing detection.

Khaled Bensid Khaled Bensid received his B.S. and M.Sc. degrees in the University Kasdi Merbah of Ouargla, Algeria, in 2010 and 2012, respectively. The Ph.D. was received in Electronic - Communication and signal processing in 2018, from Ouargla University, where he is actually an associate professor since 2019 in the Department of electronics and telecommunication. His research interests include computer vision, pattern recognition, signal and image processing, biometrics.

Abdelmalik Taleb-Ahmed received his Ph.D. in electronics and microelectronics from the Universit des Sciences et Technologies de Lille 1 in 1992 and was an associate professor at Calais until 2004. In 2004, he moved to the Universit Polytechnique des Hauts de France, where he is currently a full professor. He joined the lab IEMN DOAE. His research focused on computer vision, artificial intelligence and machine vision. His research interests include segmentation, classification, data fusion, pattern recognition, computervision and machine learning with applications in biometrics, video surveillance, autonomous driving and medical imaging. He has (co-)authored over 85 peer-reviewed articles and (co-)supervised 20 graduate students in these research areas. His current research mainly revolves around: Enhanced perception and HD Mapping in intelligent transportation, digitalization of road and signaling, E-Health and Artificial Intelligence, pattern recognition, computer vision, and information fusion, with applications in affective computing, biometrics, medical image analysis, and video analytics and surveillance.