PRIOR-CORRECTED ANCHOR LEARNING WITH DIRECT DISCRETE PARTITIONING FOR MULTI-VIEW CLUSTERING

Anonymous authors
Paper under double-blind review

ABSTRACT

Anchor-based multi-view clustering improves computational efficiency by representing the relationships among samples using a small number of anchors. However, existing methods typically employ shared semantic priors across different views. Since individual views may contain view-specific noise, feature distortions,and outliers, a shared semantic prior may fail to accurately characterize each view.Moreover, learning representations and performing clustering assignments in separate stages may result in representations that are not directly optimized toward the final clustering objective. To address these issues, we propose a Prior-Corrected Anchor Learning method, termed PCAL, which jointly learns view-specific anchor matrices, a shared anchor representation, and discrete clustering assignments within a unified framework. PCAL reconstructs each view through the shared anchor representation and aligns the anchor matrix of each view with the shared semantic prior via orthogonal projection. Building upon this mechanism, we further introduce regularized view-specific correction matrices, enabling the shared prior to adapt to the feature discrepancies across different views. In addition, a discrete factorization term is incorporated to directly connect anchor representation learning with cluster label assignment, thereby avoiding additional spectral clustering post-processing. For the proposed model, we develop an alternating optimization algorithm, in which each variable subproblem can be efficiently solved either in closed form or through singular value decomposition. Experimental results demonstrate that, compared with other multi-view clustering methods, the proposed PCAL method substantially reduces computational time while achieving competitive clustering performance in terms of metrics such as ACC and NMI.

1 INTRODUCTION

Multi-view data are widely used in image analysis, text mining, and multimedia retrieval, where the same object can be represented by different features or data sources. These views often contain complementary information,but they also differ in feature scale, expressive capacity,and noise distribution. Multi-view clustering aims to integrate information across views and discover latent cluster structure without using ground-truth sample labels. Its central challenge is to exploit cross-view consistency while accommodating differences in how the views represent the data.

Graph- and subspace-based multi-view clustering methods reveal latent structure by learning relationships among samples (Kumar et al., 2011; Peng et al., 2019). However,explicitly constructing sample–sample relationship matrices typically requires storage that grows quadratically with the number of samples, and the associated matrix operations can limit scalability. Anchor-based methods connect samples through a small set of representative anchors,moving relationship modeling from the sample–sample space to the anchor–sample space. LMVSC improves computational efficiency by constructing and fusing small graphs across views (Kang et al., 2020). SMVSC jointly optimizes anchor learning and graph construction, while FPMVS-CAG coordinates anchor selection and representation learning through consensus-anchor guidance (Sun et al., 2021; Wang et al., 2022b). Subsequent work has addressed cross-view anchor alignment, flexible anchor counts, and view-correlation-based enhancement (Wang et al., 2022a; Zhang et al., 2023; Liu et al., 2024).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

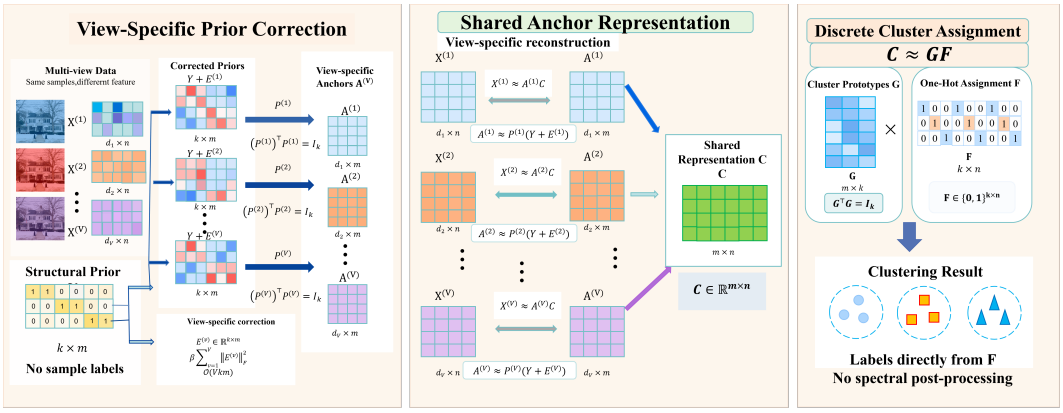

Figure 1: Overview of PCAL. View-specific corrections adapt a shared structural prior to guide anchor learning. View-specific anchors reconstruct the data through a shared representation, which is directly coupled to discrete cluster assignments.

As relationship representations become smaller, providing anchors with effective structural information becomes increasingly important for clustering quality. PALCS guides anchor learning with a predefined category–anchor structural prior and links the shared representation to a discrete cluster partition (Peng et al., 2025). Its structural prior, however, remains fixed across views.When views differ in noise,feature distortion, or local structure, the same prior may produce different alignment errors.Strictly following a fixed prior may limit adaptation to view-specific differences, whereas describing those differences with sample-dependent view-specific variables increases storage and update costs. Efficient multi-view clustering therefore calls for a compact way to adjust structural guidance while retaining shared information across views.

Coordination between representation learning and cluster assignment also affects efficiency and clustering quality. When continuous representations are learned before labels are generated, the final partition cannot directly guide representation learning,and additional post-processing is required. Previous studies have jointly modeled relationship learning and clustering through structural constraints or matrix factorization (Fang et al., 2024; Peng et al., 2025). These studies motivate an efficient anchor model that retains clustering feedback in a compact representation and supports joint optimization through small matrix operations.

Motivated by these considerations, we propose Prior-Corrected Anchor Learning (PCAL), which jointly learns view-specific anchors,a shared anchor representation, and a discrete cluster partition with the aim of balancing computational efficiency and clustering quality. A reconstruction term uses view-specific anchors to accommodate different feature spaces while integrating cross-view information through the shared representation. A prior-alignment term maps a label-independent anchor-grouping prior to the anchors of each view through a column-orthogonal transformation. View-specific correction matrices allow this prior to adapt to feature differences, while regularization controls the correction magnitude to balance shared structure and view adaptation. A discrete factorization term further connects the shared representation to the final partition, allowing cluster assignments to influence representation learning and avoiding additional spectral-clustering post-processing.

Our main contributions are summarized as follows:

1. We introduce a view-specific prior-correction mechanism that learns regularized corrections in the category–anchor space. It allows a structural prior to adapt to heterogeneous views while keeping the total correction storage at $\mathcal{O}(Vkm)$, independent of the sample count when V, k, and m are fixed.

2. We formulate a joint anchor-clustering model that coordinates data reconstruction, prior correction,and discrete cluster assignment. Structural adjustments influence the final par-

tition through the shared representation, and sample labels are obtained directly from the learned indicator matrix.

3. We develop an alternating optimization algorithm based on linear-system solves, singular value decompositions, and closed-form updates. Experiments on multiple datasets assess computational efficiency and clustering quality through running time, ACC, NMI, and related measures.

2 Related Work

2.1 Matrix-factorization-based multi-view representation learning

Matrix factorization uses view-specific basis matrices to accommodate heterogeneous feature spaces and shared or consensus constraints to combine latent information. Joint nonnegative matrix factorization coordinates the latent representations of different views (Liu et al., 2013), while deep semi-nonnegative factorization models hierarchical representations (Zhao et al., 2017) Related formulations use orthogonal factorization to obtain cluster indicators directly (Yang et al., 2021), or couple graph learning with low-rank tensor modeling (Wu et al., 2020). A low reconstruction error, however,measures how well observations are explained and may preserve variation unrelated to the desired categories.Effective cluster-structure constraints are therefore still needed.

2.2 Consistency and view specificity

Multi-view self-representation can distinguish common relationships from view-specific differences through consistent and specific components (Luo et al., 2018). Adaptive and sample-level graph models provide other ways to capture cross-view structure (Tao et al., 2017; Tan et al., 2023). Regularized multi-task learning similarly couples shared parameters with constrained task-specific variation (Evgeniou & Pontil, 2004). Recent anchor learning also organizes anchors with cluster-aware structure (Chen et al., 2026). These studies motivate attention to both the space in which view differences are modeled and the size of the resulting auxiliary variables.

2.3 Structured factorization and clustering

The connection between nonnegative matrix factorization and spectral clustering shows that factor constraints can encode category structure (Ding et al., 2005). The orthogonal Procrustes problem provides an SVD-based solution for constrained matrix alignment (Sch"onemann, 1966). Including discrete indicator variables in representation learning provides direct clustering feedback,although the resulting continuous and discrete variables require a coupled optimization procedure. Recent models couple robust consensus-anchor learning with partitioning or exploit anchor–anchor proximity (Qin et al., 2025; Yu et al., 2025).

3 Prior-Corrected Anchor Learning

PCAL jointly learns view-specific anchors, a shared anchor representation, prior corrections, and a discrete cluster partition within a unified objective. As illustrated in Figure 1, each view is reconstructed through its own anchor matrix while sharing a common sample representation.A predefined anchor grouping provides a structural reference that does not depend on ground-truth labels, and a view-specific correction matrix adjusts the alignment between this prior and the anchors of each view. The shared representation is further factorized into cluster prototypes and discrete indicators, allowing cluster assignments to participate directly in representation learning. The model obtains cluster labels through alternating updates of its variables.

3.1 Shared Anchor Representation

Given V view-specific data matrices $X^{(v)} \in \mathbb{R}^{d_v \times n}$, let n denote the number of samples and d_v the feature dimension of view v.PCAL learns an anchor matrix $A^{(v)} \in \mathbb{R}^{d_v \times m}$ for each view and uses a shared matrix $C \in \mathbb{R}^{m \times n}$ to represent all samples, where $m \ll n$. Each view is reconstructed

by $A^{(v)}C$, connecting heterogeneous feature spaces through the same shared representation. The reconstruction objective is

$$\min_{\{A^{(v)}\}_{v=1}^{V},\, C} \sum_{v=1}^{V} \left\| X^{(v)} - A^{(v)}C \right\|_F^2 . \tag{1}$$

Feature dimensions and distributions may differ substantially across views. Accordingly, each $A^{(v)}$ describes view-specific feature structure, whereas the jth column of C always represents the same sample across all views.When this column is updated, reconstruction errors from all views jointly constrain its representation, allowing complementary information to be integrated in the anchor space. The shared matrix C must retain sample-level information so that each sample can ultimately receive a cluster assignment. The model's compactness primarily comes from sharing C across views and from using view-specific correction variables whose sizes do not grow with the number of samples.

PCAL confines prior adaptation to the category–anchor space. Let n, m, k, and V denote the numbers of samples, anchors, clusters, and views, respectively. Each view has a correction matrix of size $k \times m$, so all correction matrices require $\mathcal{O}(Vkm)$ storage. When V, k, and m are fixed, this cost does not grow with the number of samples. Sample-level information remains in a shared representation, limiting the size of the auxiliary variables needed for view adaptation. This compact correction,together with direct discrete partitioning, forms the basis for PCAL's efficiency–quality trade-off.

3.2 Compact View-Specific Prior Correction

Data reconstruction alone does not specify how the anchors should be structurally organized. Motivated by prior-guided anchor learning (Peng et al., 2025; Zhang et al., 2024), PCAL uses a predefined category–anchor grouping matrix Y as a common reference. Constructing this matrix does not require ground-truth sample labels. We take $m = kh$, requiring the number of anchors to be divisible by the number of clusters, so that each latent category corresponds to h anchor positions. The structural prior is defined as

$$Y = \left[e_1 \mathbf{1}_h^\top, e_2 \mathbf{1}_h^\top, \ldots, e_k \mathbf{1}_h^\top \right], \qquad m = kh. \tag{2}$$

Here $e_i \in \mathbb{R}^k$ is the ith standard basis vector,and $\mathbf{1}_h \in \mathbb{R}^h$ is the all-ones vector. This matrix assigns the same number of anchor positions to each latent category. It is constructed solely from the numbers of clusters and anchors; it neither uses ground-truth sample labels nor prespecifies the number of samples in each cluster.

Because the anchors of different views lie in different feature spaces, PCAL introduces a column-orthogonal mapping $P^{(v)} \in \mathbb{R}^{d_v \times k}$ to align the common prior with the anchors of view v, subject to $P^{(v)\top} P^{(v)} = I_k$. The parameter $\alpha > 0$ controls the weight of prior alignment. However, a fixed common prior may produce different degrees of alignment error across views. PCAL therefore introduces a view-specific matrix $E^{(v)} \in \mathbb{R}^{k \times m}$ in the category–anchor space and uses $Y + E^{(v)}$ as the corrected structural code. A regularization term weighted by $\beta > 0$ limits the magnitude of this correction. The adjustment concerns the anchor-grouping structure rather than a view-specific representation for every sample. Consequently, all correction matrices require only $\mathcal{O}(Vkm)$ storage. This advantage applies specifically to the correction variables;the model still stores the $m \times n$ shared representation.

3.3 Joint Discrete Clustering

Optimizing only a continuous shared representation does not allow the final cluster assignments to directly influence representation learning. PCAL approximates C by the product of a cluster-prototype matrix $G \in \mathbb{R}^{m \times k}$ and a discrete indicator matrix $F \in \{0,1\}^{k \times n}$, incorporating the cluster partition into the unified objective. Joint learning of discrete indicators has been explored in prior work (Peng et al., 2025); PCAL additionally optimizes compact view-specific prior corrections

within this objective. The complete model is

$$\min_{\substack{\{A^{(v)},P^{(v)},E^{(v)}\}_{v=1}^{V},\\ C,G,F}} \quad \sum_{v=1}^{V}\Big[\left\|X^{(v)}-A^{(v)}C\right\|_F^2+\alpha\left\|A^{(v)}-P^{(v)}\big(Y+E^{(v)}\big)\right\|_F^2 + \beta\left\|E^{(v)}\right\|_F^2\Big]+\lambda\left\|C-GF\right\|_F^2. \tag{3}$$

The constraints are

$$\begin{aligned} &P^{(v)\top}P^{(v)}=I_k, \qquad v=1,\dots,V,\\ &G^\top G=I_k, \qquad F_{ij}\in\{0,1\}, \qquad \sum_{i=1}^{k}F_{ij}=1, \quad j=1,\dots,n. \end{aligned} \tag{4}$$

Here G contains the cluster prototypes, and F is the discrete indicator matrix. Each column of F has exactly one entry equal to 1, indicating the cluster assigned to the corresponding sample.The parameter $\lambda > 0$ controls agreement between the shared representation and the discrete clustering result. By jointly optimizing C, G, and F, cluster assignments can influence representation learning. Once optimization is complete, sample labels are read directly from F, without additional spectral clustering.

The objective terms have complementary roles. Data reconstruction preserves information from the observed features; prior alignment and correction constrain the structural organization of the anchors and the magnitude of their adjustments; and discrete factorization feeds the current cluster partition back into the shared representation. Prior correction is therefore not a separate post-processing step. Instead, it affects the final clustering result through C during joint representation learning.

The roles of the structural prior Y and the discrete indicator matrix F should be distinguished. The matrix $Y \in \mathbb{R}^{k\times m}$ prespecifies only the grouping relationship between latent categories and anchor positions;it contains no ground-truth sample labels. In contrast, $F \in \{0,1\}^{k\times n}$ determines sample assignments from the learned representation during optimization. They act on anchor structure and sample partitioning, respectively, and are connected indirectly through the anchors, shared representation, and cluster prototypes. Thus, using the number of clusters k to construct the prior does not amount to prespecifying the clustering result.

4 Optimization and Complexity Analysis

4.1 Alternating Optimization

The objective contains coupled variables,orthogonality constraints,and discrete constraints. PCAL fixes the other variables and sequentially updates $A^{(v)}$, $P^{(v)}$, C, $E^{(v)}$, G, and F. The hyperparameters α, β, and λ remain fixed during optimization. In terms of variable dependence,once the shared representation and clustering variables are fixed, the anchors, mappings, and corrections of different views can be updated separately using the data and variables of their respective views. Updating the shared representation then aggregates information from all views.Each step uses the latest available values of the other variables.

Updating the anchor matrices. Let $B^{(v)} = Y + E^{(v)}$. Fixing the other variables and setting the gradient with respect to $A^{(v)}$ to zero gives

$$A^{(v)}=\left[X^{(v)}C^\top+\alpha P^{(v)}B^{(v)}\right]\left(CC^\top+\alpha I_m\right)^{-1}. \tag{5}$$

Because $\alpha > 0$, the coefficient matrix is positive definite. The views can reuse its factorization to solve their linear systems, reducing computational cost.

Updating the mapping matrices. With the other variables fixed, updating $P^{(v)}$ involves only the prior-alignment term. Under the column-orthogonality constraint, the subproblem is equivalent to

$$\max_{P^{(v)\top}P^{(v)}=I_k}\operatorname{Tr}\left(P^{(v)\top}A^{(v)}B^{(v)\top}\right). \tag{6}$$

This is an orthogonal Procrustes problem.Taking a thin singular value decomposition (SVD) gives

$$A^{(v)}B^{(v)\top} = U_P^{(v)}\Sigma_P^{(v)}V_P^{(v)\top}, \qquad P^{(v)} = U_P^{(v)}V_P^{(v)\top}. \tag{7}$$

Updating the shared representation. The reconstruction terms of all views and the discrete clustering term jointly determine C. Setting its gradient to zero yields the linear system

$$\left(\sum_{v=1}^{V} A^{(v)\top}A^{(v)} + \lambda I_m\right) C = \sum_{v=1}^{V} A^{(v)\top}X^{(v)} + \lambda GF. \tag{8}$$

Solving this system updates the representations of all samples at once. The updates of the anchor matrices and shared representation involve the $m \times m$ coefficient matrices $CC^\top + \alpha I_m$ and $\sum_v A^{(v)\top}A^{(v)} + \lambda I_m$, respectively. Since $\alpha, \lambda > 0$, both are positive definite. The former is shared across the view-specific anchor updates, while the latter is solved with right-hand sides corresponding to all samples. Neither update requires constructing or factorizing an $n \times n$ matrix.

Updating the prior-correction matrices. The subproblem for $E^{(v)}$ contains only the prior-alignment and correction-regularization terms. Using $P^{(v)\top}P^{(v)} = I_k$, the first-order condition is

$$(\alpha + \beta)E^{(v)} = \alpha\left(P^{(v)\top}A^{(v)} - Y\right). \tag{9}$$

Therefore, the correction has the closed-form update

$$E^{(v)} = \frac{\alpha}{\alpha + \beta}\left(P^{(v)\top}A^{(v)} - Y\right). \tag{10}$$

This update shrinks the discrepancy between the projected anchors and the shared prior, allowing adaptation to view-specific differences while controlling the correction magnitude.

Updating the cluster prototypes. Under $G^\top G = I_k$, minimizing $\|C - GF\|_F^2$ is equivalent to maximizing

$$\max_{G^\top G = I_k} \operatorname{Tr}\left(G^\top CF^\top\right). \tag{11}$$

A thin SVD of $CF^\top$ gives

$$CF^\top = U_G\Sigma_G V_G^\top, \qquad G = U_G V_G^\top. \tag{12}$$

Updating the discrete indicator matrix. With C and G fixed, the cluster assignment of each sample can be solved independently. Let c_j denote the jth column of C and g_i the ith cluster prototype. The cluster index of sample j is

$$r_j = \arg\min_{1 \le i \le k} \|c_j - g_i\|_2^2. \tag{13}$$

The jth column of F is then updated as

$$F_{ij} = \begin{cases} 1, & i = r_j, \\ 0, & i \neq r_j, \end{cases} \qquad j = 1, \ldots, n. \tag{14}$$

Since $G^\top G = I_k$, all prototypes have the same norm. The assignment can therefore also be obtained by maximizing $g_i^\top c_j$.

Starting from feasible initial values, PCAL repeats these updates until the relative change in the objective falls below a threshold or the maximum number of iterations is reached. When each subproblem is solved exactly, the nonnegative objective values form a nonincreasing sequence and therefore converge. This property does not guarantee a global optimum. The final cluster labels are obtained directly from F.

4.2 Complexity Analysis

Let $D = \sum_{v=1}^{V} d_v$ denote the total feature dimension across views, and let T be the number of iterations. The following analysis assumes $k \le m \ll n$ and $d_v \ge k$.

Time complexity. Computing the data–representation products across views requires $\mathcal{O}(Dnm)$ time. Constructing the relevant matrix of the shared representation and solving the linear system with all samples as right-hand sides costs $\mathcal{O}(nm^2)$. Small-matrix operations in the view-specific anchor updates require $\mathcal{O}(Dm^2)$, and factorizing an $m \times m$ coefficient matrix costs $\mathcal{O}(m^3)$. The mapping and prototype updates use thin SVDs, while the corrections have closed-form updates. Computing representation–prototype products for discrete assignment costs $\mathcal{O}(mnk)$. Under $k \leq m$ and reuse of the shared coefficient-matrix factorization, the total optimization time complexity is

$$\mathcal{O}\big(T\left[Dnm + nm^2 + Dm^2 + m^3 + mnk\right]\big). \tag{15}$$

When m, k, V, the feature dimensions, and T are fixed, the optimization cost grows linearly with n. This expression excludes the truncated SVD and clustering computations used for initialization. Final labels follow directly from F, without spectral-clustering post-processing.

Space complexity. Excluding the input data and temporary reconstruction buffers, the main stored variables are the shared representation C, the anchors and mapping matrices of all views, the prior-correction matrices, the clustering variables, and work matrices for the linear systems. Using $k \leq m$, the principal space complexity is

$$\mathcal{O}\big(mn + Dm + Vkm + m^2\big). \tag{16}$$

PCAL does not explicitly construct an $n \times n$ sample-affinity matrix. Its main sample-level variable is the $m \times n$ shared representation, and the view-specific corrections occupy only $\mathcal{O}(Vkm)$ space, independent of the sample count. The stated space bound excludes input data and temporary buffers, so it should not be interpreted as the total memory consumption of an entire implementation. In particular, the storage advantage specific to the correction variables does not imply that the whole model is independent of n.

5 Experiments

5.1 Experimental Setup

We evaluate the clustering quality and computational efficiency of PCAL on six public multi-view datasets: MSRC, BBCSport, Caltech101-20, CCV, NUS-WIDE, and NUSWIDEOBJ, with sample counts ranging from 210 to 30,000. The compared methods are LMVSC (Kang et al., 2020), SMVSC (Sun et al., 2021), FPMVS-CAG (Wang et al., 2022b), UDBGL (Fang et al., 2024), OMSC (Chen et al., 2022), OPMC (Liu et al., 2021), and FastMICE (Huang et al., 2023). We primarily report clustering accuracy (ACC), normalized mutual information (NMI), and the running time required to obtain labels. Purity, F-score, and complete experimental settings are provided in the appendix.

After selecting the parameters, we repeat each method with seeds 11–20 and report the mean and standard deviation. The parameter search uses clustering measures computed from ground-truth labels; it should therefore not be interpreted as parameter selection on an independent test set. running time is measured from the actual method call until labels are produced. Data loading, parameter search, and metric computation are excluded. The current OPMC interface standardizes its input before timing; the preprocessing procedures and timing scopes of all methods are detailed in Appendix B.3.

5.2 Clustering Performance

Table 1 reports mean ACC and NMI. PCAL ranks second on both measures on MSRC and BBCSport and first on CCV, but does not lead on Caltech101-20 or the two NUS datasets. Its mean ACC on MSRC (0.7771) is close to FastMICE (0.7819), while its BBCSport ACC varies substantially across runs (standard deviation 0.1128). These results support competitive, rather than uniformly superior, clustering quality.

5.3 Running time and Scalability

Figure 2 compares ACC and mean running-time on the six datasets. PCAL is the fastest method on CCV and the second fastest on BBCSport, Caltech101-20, NUS-WIDE, and NUSWIDEOBJ.

Table 1: Clustering results on six public multi-view datasets. Each cell reports mean ACC/NMI.

Method	MSRC	BBCSport	Caltech101-20	CCV	NUS-WIDE	NUSWIDEOBJ
LMVSC	0.698/0.612	0.579/.464	0.387/.534	0.205/0.165	0.145/0.178	0.129/0.120
SMVSC	0.672/0.536	0.379/0.140	0.618/0.603	0.215/0.169	0.203/0.188	0.189/0.131
FPMVS-CAG	0.732/0.614	0.406/0.106	0.664/0.639	0.237/0.182	0.207/0.192	0.193/0.137
UDBGL	0.594/0.562	0.397/0.106	0.404/0.515	0.153/0.133	0.142/0.099	0.140/0.078
OMSC	0.762/0.653	0.406/0.106	0.661/0.642	0.240/0.186	0.216/0.189	0.199/0.127
OPMC	0.640/0.563	0.262/0.021	0.456/0.625	0.202/0.178	0.162/0.203	0.145/0.146
FastMICE	0.782/0.698	0.970/0.902	0.433/0.607	0.201/0.156	0.170/0.206	0.149/0.144
PCAL	0.777/0.671	0.644/0.476	0.568/0.609	0.246/0.197	0.195/0.172	0.166/0.126

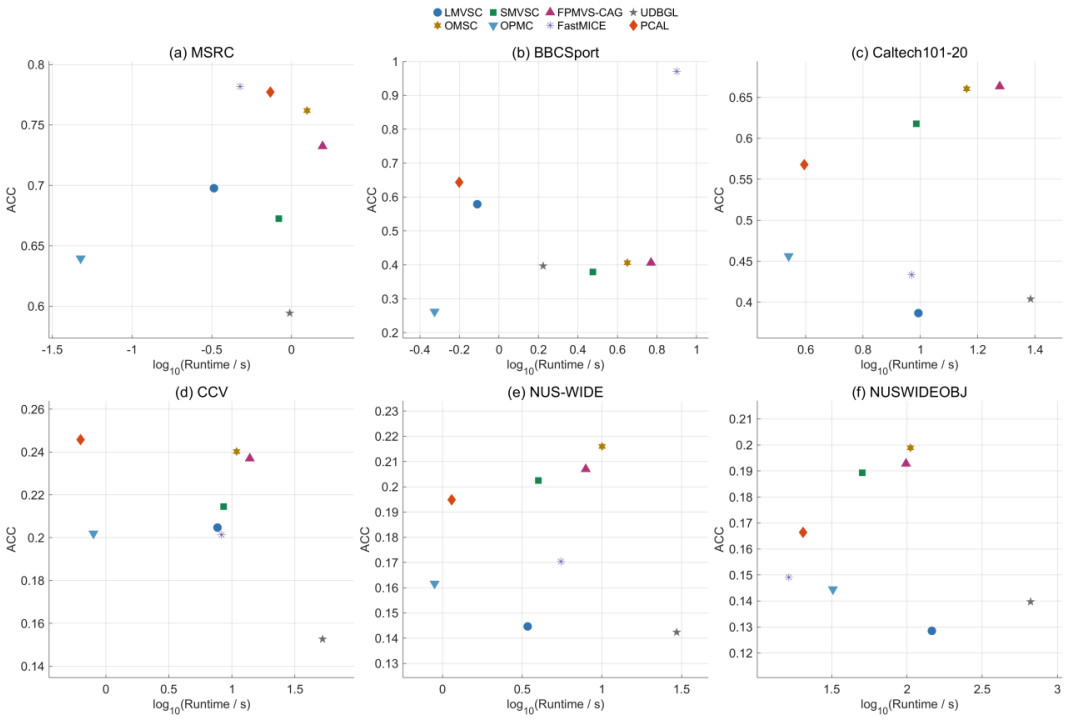

Figure 2: ACC versus running time for different methods on six datasets. The horizontal axis shows running time on a base-10 logarithmic scale.

On CCV, it achieves the highest mean ACC and NMI in approximately 0.63 seconds. On NUSWIDEOBJ, which contains 30,000 samples, its mean running-time is 20.41 seconds, shorter than those of SMVSC, FPMVS-CAG, and OMSC, although these three methods achieve higher ACC. PCAL lies on the ACC–running time Pareto frontier on five of the six datasets; under NMI and running time, it lies on the frontier on two. Complete running time and scalability results are given in Appendix D.

On CCV, PCAL takes 0.631 seconds on average, approximately one 17th of OMSC's running-time, while achieving higher ACC and NMI. On NUSWIDEOBJ, PCAL is faster than SMVSC, FPMVS-CAG, and OMSC, but these methods obtain higher ACC. FastMICE is faster than PCAL and also has higher NMI on this dataset. Thus, running time advantages should be interpreted together with the chosen clustering measure. Complete running time results and comparisons between NMI and running time are presented in Table 9 and Figure 7, respectively.

To examine how PCAL's computational cost changes with data size, we perform stratified subsampling on NUSWIDEOBJ. As the sample count increases from approximately 6,000 to 30,000, mean solver running time increases from 8.95 to 20.51 seconds, showing gradual growth over the tested range. The sampling procedure, detailed timings, and scaling curve are given in Table 6 and Figure 6. This experiment uses a fixed solver-level timing scope to compare different sample sizes.

Table 2: Fixed-parameter component ablation results for PCAL. Each cell reports mean ACC/NMI.

Dataset	Base	CorrectionOnly	DiscreteOnly	Full PCAL
MSRC	0.7452/0.6470	0.7757/0.6639	0.7633/0.6802	0.7771/0.6713
BBCSport	0.3682/0.0924	0.3686/0.0930	0.6432/0.4751	0.6438/0.4755
Caltech101-20	0.2145/0.1780	0.2123/0.1705	0.5811/0.6050	0.5679/0.6092
CCV	0.0824/0.0206	0.0840/0.0208	0.2458/0.1965	0.2458/0.1965
NUS-WIDE	0.1503/0.1955	0.1485/0.1958	0.1954/0.1694	0.1948/0.1723
NUSWIDEOBJ	0.1502/0.1494	0.1523/0.1523	0.1664/0.1262	0.1664/0.1262

5.4 Ablation Study

We construct four variants around prior correction and joint discrete partitioning. BASE disables both components; CORRECTIONONLY retains only prior correction; DISCRETEONLY retains only joint discrete partitioning; and FULL PCAL includes both. All variants retain data reconstruction and prior alignment. When correction is disabled, $E^{(v)}$ is fixed to zero. When joint discrete partitioning is disabled, the corresponding factorization term is removed and k-means is applied to the final shared representation. All variants use the full model's parameter configuration and the same repetition seeds, allowing us to examine component removal under a fixed configuration.

Joint discrete partitioning substantially improves both ACC and NMI on BBCSport, Caltech101-20, and CCV. On NUS-WIDE and NUSWIDEOBJ, it increases ACC but decreases NMI, so its contribution depends on the evaluation measure.

The effect of prior correction is dataset dependent. On MSRC, CORRECTIONONLY improves both ACC and NMI relative to BASE. On Caltech101-20, FULL PCAL improves NMI but reduces ACC relative to DISCRETEONLY. On CCV and NUSWIDEOBJ, these two variants have identical displayed clustering scores. On NUSWIDEOBJ, the mean running-times of DISCRETEONLY and FULL PCAL are 6.87 and 20.49 seconds, respectively, indicating that correction-related computation and changes in the optimization process can increase running time. Prior correction must therefore be assessed using both clustering quality and computational cost; the current results do not show an additional benefit on every dataset.

Comparing CORRECTIONONLY with BASE, and FULL PCAL with DISCRETEONLY, also shows that the effect of correction changes depending on whether joint discrete partitioning participates in optimization. The contributions of these components cannot simply be added independently. Moreover, all variants use parameters selected for the full model. The observed differences characterize component removal under a fixed configuration, not the best results obtainable after separately retuning each reduced model.

Appendix A provides supplementary model derivations. Appendix B documents the datasets, implementation settings, and timing protocols. Appendix C reports parameter sensitivity and convergence, while Appendix D contains the complete clustering and running time results, including the sample-size study.

6 Conclusion

We proposed PCAL, which introduces view-specific prior correction in the category–anchor space within a framework for shared anchor representation learning and direct discrete partitioning. This design allows different views to make constrained adjustments around a common structural reference while keeping correction storage at $\mathcal{O}(Vkm)$. The alternating optimization procedure uses linear-system solves, singular value decompositions, and closed-form updates, and obtains cluster labels directly from the discrete indicator matrix. Results on six datasets show that PCAL achieves competitive clustering quality with low running time in several settings. However, the benefits of prior correction and the trade-offs among evaluation measures depend on the dataset. Future work will investigate which forms of view heterogeneity benefit most from prior correction and assess computational efficiency under more closely matched preprocessing and parameter-search protocols.

Statement on the Use of AI Tools

We used AI tools to assist with English language editing and LaTeX formatting. All AI-assisted content was reviewed and verified by the authors, who take full responsibility for the accuracy, originality, and integrity of the manuscript.

References

Man-Sheng Chen, Chang-Dong Wang, Dong Huang, Jian-Huang Lai, and Philip S. Yu. Efficient orthogonal multi-view subspace clustering. In *Proceedings of the 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pp. 127–135, 2022.

Zhe Chen, Fanhui Meng, Tianyang Xu, and Xiao-Jun Wu. Cluster-aware anchor learning for multi-view clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 17714–17723, 2026.

Chris Ding, Xiaofeng He, and Horst D. Simon. On the equivalence of nonnegative matrix factorization and spectral clustering. In *Proceedings of the 2005 SIAM International Conference on Data Mining*, pp. 606–610, 2005.

Theodoros Evgeniou and Massimiliano Pontil. Regularized multi-task learning. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pp. 109–117, 2004.

Si-Guo Fang, Dong Huang, Xiao-Sha Cai, Chang-Dong Wang, Chaobo He, and Yong Tang. Efficient multi-view clustering via unified and discrete bipartite graph learning. *IEEE Transactions on Neural Networks and Learning Systems*, 35(8):11436–11447, 2024.

Dong Huang, Chang-Dong Wang, and Jian-Huang Lai. Fast multi-view clustering via ensembles: Towards scalability, superiority, and simplicity. *IEEE Transactions on Knowledge and Data Engineering*, 35:11388–11402, 2023.

Zhao Kang, Wangtao Zhou, Zhitong Zhao, Junming Shao, Meng Han, and Zenglin Xu. Large-scale multi-view subspace clustering in linear time. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 4412–4419, 2020.

Abhishek Kumar, Piyush Rai, and Hal Daumé. Co-regularized multi-view spectral clustering. In *Advances in Neural Information Processing Systems*, volume 24, 2011.

Jialu Liu, Chi Wang, Jing Gao, and Jiawei Han. Multi-view clustering via joint nonnegative matrix factorization. In *Proceedings of the 2013 SIAM International Conference on Data Mining*, pp. 252–260, 2013.

Jiyuan Liu, Xinwang Liu, Yuexiang Yang, Li Liu, Siqi Wang, Weixuan Liang, and Jiangyong Shi. One-pass multi-view clustering for large-scale data. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 12344–12353, 2021. doi: 10.1109/ICCV48922.2021.01212.

Suyuan Liu, Ke Liang, Zhibin Dong, Siwei Wang, Xihong Yang, Sihang Zhou, En Zhu, and Xinwang Liu. Learn from view correlation: An anchor enhancement strategy for multi-view clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 26151–26161, 2024.

Shirui Luo, Changqing Zhang, Wei Zhang, and Xiaochun Cao. Consistent and specific multi-view subspace clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, pp. 3730–3737, 2018.

Jinjia Peng, Tianhang Cheng, Guangqi Jiang, and Huibing Wang. Prior-oriented anchor learning with coalesced semantics for multi-view clustering. In *Proceedings of the 33rd ACM International Conference on Multimedia*, pp. 1141–1150, 2025.

Xi Peng, Zhenyu Huang, Jiancheng Lv, Hongyuan Zhu, and Joey Tianyi Zhou. COMIC: Multi-view clustering without parameter selection. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 5092–5101, 2019.

Yalan Qin, Nan Pu, Guorui Feng, and Nicu Sebe. Robust consensus anchor learning for efficient multi-view subspace clustering. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 50327–50337, 2025.

Peter H. Sch"onemann. A generalized solution of the orthogonal procrustes problem. *Psychometrika*, 31:1–10, 1966.

Mengjing Sun, Pei Zhang, Siwei Wang, Sihang Zhou, Wenxuan Tu, Xinwang Liu, En Zhu, and Changjian Wang. Scalable multi-view subspace clustering with unified anchors. In *Proceedings of the 29th ACM International Conference on Multimedia*, pp. 3528–3536, 2021.

Yuze Tan, Yixi Liu, Shudong Huang, Wentao Feng, and Jiancheng Lv. Sample-level multi-view graph clustering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 23966–23975, 2023.

Hong Tao, Chenping Hou, Jubo Zhu, and Dongyun Yi. Multi-view clustering with adaptively learned graph. In *Proceedings of the Ninth Asian Conference on Machine Learning*, volume 77 of *Proceedings of Machine Learning Research*, pp. 113–128, 2017.

Siwei Wang, Xinwang Liu, Suyuan Liu, Jiaqi Jin, Wenxuan Tu, Xinzhong Zhu, and En Zhu. Align then fusion: Generalized large-scale multi-view clustering with anchor matching correspondences. In *Advances in Neural Information Processing Systems*, volume 35, 2022a.

Siwei Wang, Xinwang Liu, Xinzhong Zhu, Pei Zhang, Yi Zhang, Feng Gao, and En Zhu. Fast parameter-free multi-view subspace clustering with consensus anchor guidance. *IEEE Transactions on Image Processing*, 31:556–568, 2022b.

Jianlong Wu, Xingyu Xie, Liqiang Nie, Zhouchen Lin, and Hongbin Zha. Unified graph and low-rank tensor learning for multi-view clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pp. 6388–6395, 2020.

Ben Yang, Xuetao Zhang, Feiping Nie, Fei Wang, Weizhong Yu, and Rong Wang. Fast multi-view clustering via nonnegative and orthogonal factorization. *IEEE Transactions on Image Processing*, 30:2575–2586, 2021.

Shengju Yu, Zhibin Dong, Siwei Wang, Suyuan Liu, Ke Liang, Xinwang Liu, Yue Liu, and Yi Zhang. From spectrum-free towards baseline-view-free: Double-track proximity driven multi-view clustering. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 72895–72924, 2025.

Chao Zhang, Xiuyi Jia, Zechao Li, Chunlin Chen, and Huaxiong Li. Learning cluster-wise anchors for multi-view clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 16696–16704, 2024.

Pei Zhang, Siwei Wang, Liang Li, Changwang Zhang, Xinwang Liu, En Zhu, Zhe Liu, Lu Zhou, and Lei Luo. Let the data choose: Flexible and diverse anchor graph fusion for scalable multi-view clustering. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 11262–11269, 2023.

Handong Zhao, Zhengming Ding, and Yun Fu. Multi-view clustering via deep matrix factorization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 31, 2017.

A Supplementary Model Construction and Map Update

A.1 From shared anchors to prior correction

For view v, let $X^{(v)} \in \mathbb{R}^{d_v \times n}$, $A^{(v)} \in \mathbb{R}^{d_v \times m}$, and $C \in \mathbb{R}^{m \times n}$ be the data, view-specific anchors, and shared representation, respectively. Reconstruction through $A^{(v)}C$ alone gives

$$\min_{\{A^{(v)}\}_{v=1}^{V}, C} \sum_{v=1}^{V} \|X^{(v)} - A^{(v)}C\|_F^2. \tag{17}$$

This objective shares sample representations but imposes no explicit anchor-group structure. Let $Y \in \mathbb{R}^{k \times m}$ be the prior in Equation (2). Mapping it to view v with $P^{(v)} \in \mathbb{R}^{d_v \times k}$ leads to

$$\begin{aligned} \min_{\{A^{(v)}, P^{(v)}\}_{v=1}^{V}, C} \quad & \sum_{v=1}^{V} \Big[\|X^{(v)} - A^{(v)}C\|_F^2 + \alpha \|A^{(v)} - P^{(v)}Y\|_F^2 \Big], \\ \text{s.t.} \quad & P^{(v)\top} P^{(v)} = I_k, \quad v = 1, \dots, V. \end{aligned} \tag{18}$$

Here $\alpha > 0$ weights prior alignment relative to reconstruction. The orthogonality constraint controls the map's scale.

The same fixed Y may not fit every view equally well. PCAL therefore introduces $E^{(v)} \in \mathbb{R}^{k \times m}$ and regularizes its magnitude:

$$\begin{aligned} \min_{\{A^{(v)}, P^{(v)}, E^{(v)}\}_{v=1}^{V}, C} \quad & \sum_{v=1}^{V} \Big[\|X^{(v)} - A^{(v)}C\|_F^2 + \alpha \|A^{(v)} - P^{(v)}(Y + E^{(v)})\|_F^2 \\ & \qquad + \beta \|E^{(v)}\|_F^2 \Big], \\ \text{s.t.} \quad & P^{(v)\top} P^{(v)} = I_k, \quad v = 1, \dots, V. \end{aligned} \tag{19}$$

The coefficient $\beta > 0$ penalizes departure from the shared prior. Since $E^{(v)}$ is real-valued, $Y + E^{(v)}$ is a continuous structural code, not a probability matrix. Equation (19) does not yet include a discrete partition; adding the shared-representation factorization gives Equation (3).

A.2 Orthogonal Procrustes update of the maps

Fixing the other variables and writing $B^{(v)} = Y + E^{(v)}$, the map subproblem is

$$\min_{P^{(v)\top} P^{(v)} = I_k} \|A^{(v)} - P^{(v)} B^{(v)}\|_F^2. \tag{20}$$

Expansion under $P^{(v)\top} P^{(v)} = I_k$ yields

$$\|A^{(v)} - P^{(v)} B^{(v)}\|_F^2 = \|A^{(v)}\|_F^2 + \|B^{(v)}\|_F^2 - 2 \operatorname{Tr}\Big(P^{(v)\top} A^{(v)} B^{(v)\top} \Big). \tag{21}$$

The first two terms are constant for this subproblem, so Equation (20) is equivalent to

$$\max_{P^{(v)\top} P^{(v)} = I_k} \operatorname{Tr}\Big(P^{(v)\top} A^{(v)} B^{(v)\top} \Big). \tag{22}$$

If $A^{(v)} B^{(v)\top} = U_P^{(v)} \Sigma_P^{(v)} V_P^{(v)\top}$ is a thin SVD, one optimum is $P^{(v)} = U_P^{(v)} V_P^{(v)\top}$, as used in Equation (7).

B Datasets and Implementation Details

B.1 Datasets and evaluation protocol

Table 3 lists sample counts, views, clusters, and feature dimensions. All methods use the corresponding dataset versions and evaluation procedures, and the number of clusters is set to the dataset's ground-truth cluster count. ACC, NMI, Purity, and F-score are used for evaluation. Ground-truth

Table 3: Dataset statistics.

Dataset	Samples	Views	Clusters	Feature dimensions
MSRC	210	5	7	24, 576, 512, 256, 254
BBCSport	544	2	5	3183, 3203
Caltech101-20	2386	6	20	48, 40, 254, 1984, 512, 928
CCV	6773	3	20	20, 20, 20
NUS-WIDE	2000	5	31	65, 226, 145, 74, 129
NUSWIDEOBJ	30000	5	31	65, 226, 145, 74, 129

labels are not used in PCAL's objective or prior construction, but metrics computed from them are used during hyperparameter search.

PCAL searches three candidate anchor counts and eight values for each of three weights, forming $3 \times 8^3 = 1{,}536$ configurations. After selecting a configuration, parameters remain fixed for ten runs with seeds 11–20. Independent seeds measure variation due to random initialization; they do not constitute validation on an independent test dataset.

B.2 PCAL PARAMETERS AND IMPLEMENTATION

Table 4 gives the final PCAL settings; α, β, and λ weight prior alignment, correction regularization, and discrete factorization, respectively.

Table 4: Final PCAL parameters.

Dataset	m	α	β	λ
MSRC	7	10^{4}	10^{3}	1
BBCSport	10	10^{-2}	10^{-3}	10^{-3}
Caltech101-20	60	10^{4}	10^{4}	10
CCV	20	10^{-1}	10^{-3}	10^{3}
NUS-WIDE	31	10^{4}	10^{4}	10^{-1}
NUSWIDEOBJ	31	10^{4}	10^{-3}	10^{4}

PCAL standardizes each view and initializes the shared representation using truncated SVD and k-means. The initialization k-means uses at most 100 iterations and five restarts. Alternating optimization runs for at most 100 iterations and stops when the relative change in consecutive objective values is below 10^{-3} or the objective value falls below 10^{-10}. For variants without joint discrete partitioning, k-means is applied to the shared representation to obtain labels, with at most 200 iterations and 20 restarts; its running time is included.

All experiments were performed on an AMD Ryzen 7 5800U machine with 16 GB RAM, 64-bit Windows 10, and MATLAB R2024b. Parallel implementations used a local pool of eight workers. The pool was started before timing, and its startup time was excluded.

Table 5: Experimental environment.

Item	Configuration
CPU	AMD Ryzen 7 5800U
Memory	16 GB
Operating system	Windows 10 Enterprise 22H2, 64-bit
MATLAB	R2024b (24.2.0.2712019)
Parallel workers	8

B.3 BASELINE PARAMETERS AND IMPLEMENTATION

The seven baselines use public implementations with a common interface for data loading, result saving, and metric computation. Their cluster counts equal the dataset ground-truth counts. For

methods requiring parameter search, candidate settings are evaluated with seed 99; among successful runs, the highest ACC is selected, with NMI and then FSC breaking ties. The chosen settings are then fixed for runs with seeds 11–20. Ground-truth labels are used for configuration selection and evaluation, so these repeated runs measure variability rather than independent-test generalization. Identically named parameters in different methods need not have the same meaning.

LMVSC. Anchors are selected from $\{k, 2k, 3k\}$, and its regularization parameter from $\{10^{-2}, 1, 10^{2}\}$, for nine configurations. Each view uses k-means anchors, with at most 100 iterations and ten internal restarts. The learned anchor graphs are then fused to obtain a clustering embedding.

SMVSC. Anchors are selected from $\{k, 2k, 3k\}$ and the latent dimension is k. Its implementation parameter λ is fixed to 1, yielding three configurations. We retain the implementation's standardization and stopping rules: at least ten iterations, followed by stopping when relative objective change is below 10^{-3}, objective value is below 10^{-10}, or the iteration count exceeds 50.

FPMVS-CAG. Anchors are selected from $\{k, 2k, 3k\}$ and the latent dimension is k. The interface parameter λ is set to 0, yielding three configurations. Optimization uses the authors' released P-code and internal settings, without imposing a common external iteration limit.

UDBGL. Anchors are selected from $\{k, 2k, 3k\}$; its parameters α and β are fixed to 10^{-3} and 10^{-5}, respectively, yielding three configurations. We retain feature normalization and anchor initialization. The outer iteration limit is 20 and the internal bipartite-graph-update limit is 50. Labels are taken directly from the method's output. For repeatability, an internally fixed seed was replaced with the current experimental seed, and the graph connected-components call was updated for compatibility.

OMSC. Anchors are selected from $\{k, 2k, 3k\}$, the latent dimension is k, and $\lambda \in \{10^{-3}, 10^{-2}, 10^{-1}, 1\}$, for 12 configurations. We retain the implementation's data standardization and fixed 15 iterations. Labels are obtained by taking the maximum entry in each column of its output indicator matrix.

OPMC. The cluster count and latent dimension both equal k. Each view is standardized by feature, without an additional hyperparameter grid. The implementation uses random initialization and equal view weights. After more than two iterations, it stops when relative objective change is below 10^{-5}, and labels are taken from its output.

FastMICE. The ensemble size is 20, the number of anchors is $\min(1000, n)$, and the neighbor count is five. BBCSport uses cosine distance; the other datasets use Euclidean distance. The random feature-sampling ratio ranges from 0.2 to 0.8, and base-clustering cluster counts range from k to $2k$. We retain ensemble and consensus clustering, while removing a system-clock-based seed reset so that the external experiment seed controls randomness.

For LMVSC, SMVSC, and FPMVS-CAG, which output continuous embeddings, each embedding row is normalized to unit ℓ_2 norm and then processed by the same k-means post-processing, with at most 200 iterations and 20 internal restarts. These restarts differ from the ten independent experimental runs, and post-processing time is included.

Running time is measured from the selected method call until final labels are available. It excludes file loading, parameter search, metric computation, and parallel-pool startup. Internal preprocessing is included, except that the present OPMC interface standardizes its input before timing. Therefore, the reported running time should not be interpreted as fully standardized end-to-end times. The methods also use different parameter search spaces and their own stopping criteria; results reflect the specified configurations rather than an equal tuning budget.

C Parameter Sensitivity and Convergence

C.1 Parameters and anchor count

On MSRC, CCV, and NUSWIDEOBJ, we vary α, β, λ, and m/k while fixing other parameters. The curves are drawn from parameter-search records and show trends under those search settings; they do not represent ten independent evaluations at each point. Figures 3 and 4 show ACC and NMI, respectively.

Within the displayed ranges, the model is relatively insensitive to β, whereas responses to α, λ, and the anchor count differ by dataset. Some weights affect MSRC more strongly; several CCV and NUSWIDEOBJ curves are comparatively flat. None of the three datasets shows consistent improvement as anchors are added, so anchor count should be chosen with the data and computational budget in mind.

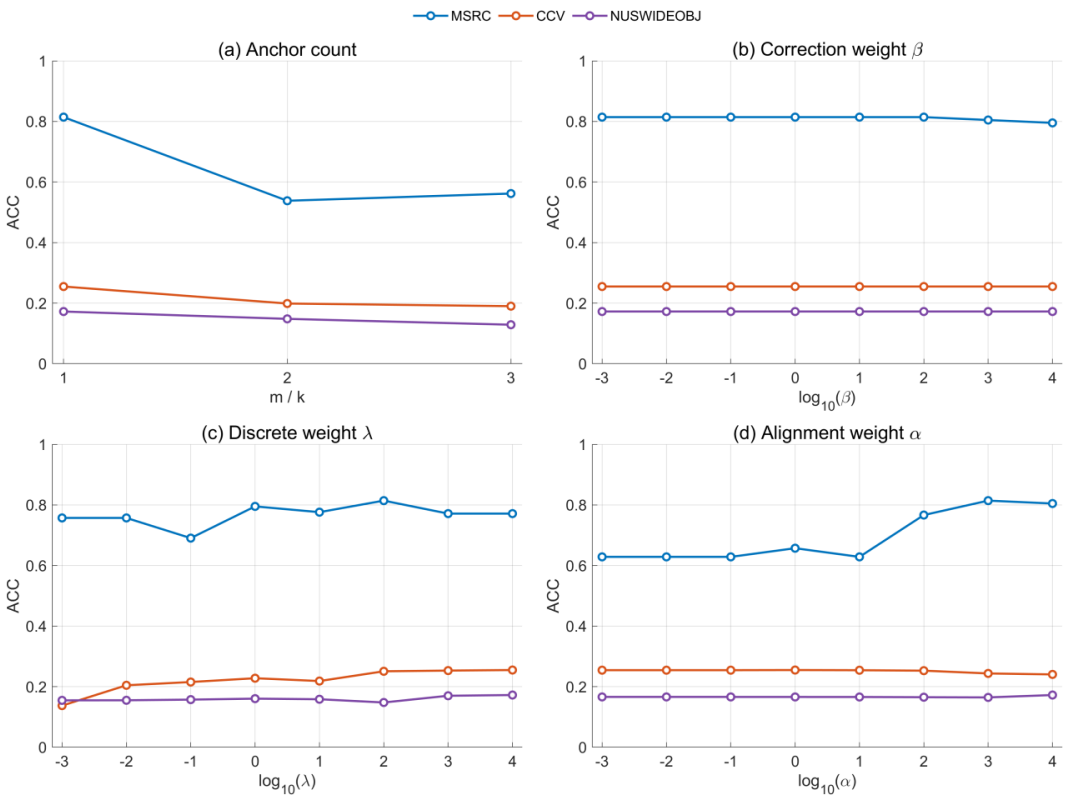

Figure 3: ACC sensitivity to anchor count and the PCAL weights.

C.2 Convergence behavior

We record objective values on MSRC, Caltech101-20, and NUSWIDEOBJ and normalize each curve by its first-iteration value. The displayed curves decrease early and then stabilize, showing numerical descent under these settings. They do not prove global optimality or identical behavior for all initializations.

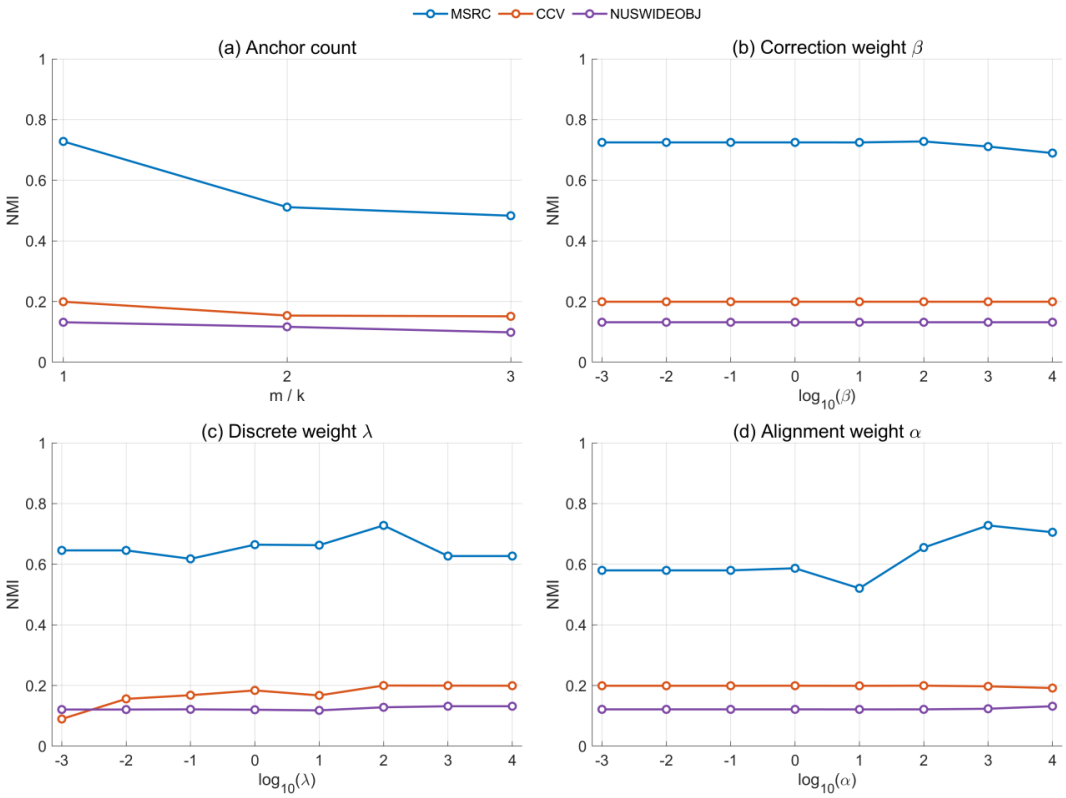

Figure 4: NMI sensitivity to anchor count and the PCAL weights.

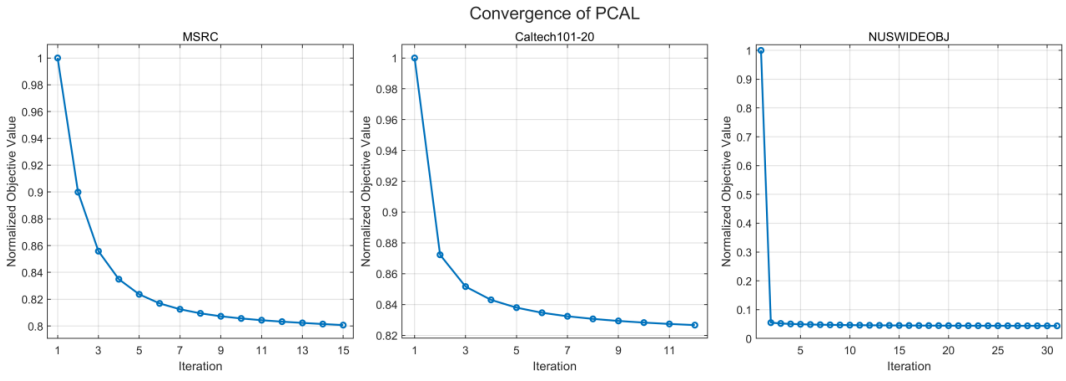

Figure 5: Normalized objective values during PCAL optimization.

D Scalability and Supplementary Efficiency Results

D.1 Running time at different sample sizes

We stratify NUSWIDEOBJ by cluster and sample approximately 20%, 40%, 60%, 80%, and 100% of the data. The same model parameters are used at every size, with three runs using seeds 11–13. Independent rounding within clusters produces slight deviations from ideal sample proportions.

Table 6: Actual sample sizes and solver running time on NUSWIDEOBJ (seconds).

Samples	Running time, mean $\pm$ standard deviation
6002	8.95 ± 0.23
12001	12.51 ± 0.37
17999	15.16 ± 0.24
23998	17.79 ± 0.49
30000	20.51 ± 0.09

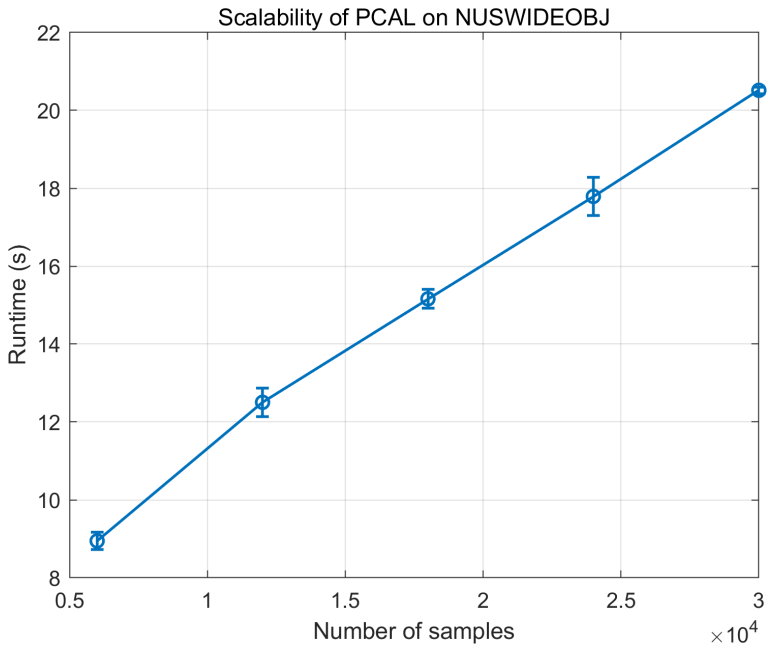

Figure 6: PCAL running time versus the number of NUSWIDEOBJ samples.

This experiment times the PCAL solver call, including its internal initialization and optimization, but excluding label-index extraction from the returned indicator matrix and metric computation. Its endpoint therefore differs slightly from the main comparison protocol. Running time grows gradually over the tested range; initialization, iteration counts, and numerical overhead also affect elapsed time, so this finite-range curve is not a proof of linear time complexity.

D.2 Complete quality and efficiency comparison

Table 7 reports ACC and NMI means and standard deviations for all methods. Table 9 gives mean running-time, Figure 7 plots NMI against running-time, and Table 8 gives Purity and F-Score. PCAL is on the ACC–running-time Pareto frontier for five datasets and on the NMI–running-time frontier for two; efficiency gains must therefore be interpreted with respect to the particular clustering measure.

On MSRC and BBCSport,PCAL ranks second on both Purity and F-score;on CCV it has the highest means,0.2742 and 0.1403,exceeding the second-ranked OMSC by approximately 0.53 and 0.19 per-

Table 7: ACC and NMI on six datasets (mean ± standard deviation).

Method	MSRC	BBCSport	Caltech101-20	CCV	NUS-WIDE	NUSWIDEOBJ
			ACC			
LMVSC	0.6976±0.0197	0.5789±0.0775	0.3865±0.0310	0.2047±0.0075	0.1447±0.0056	0.1285±0.0037
SMVSC	0.6724±0.0233	0.3787±0.0000	0.6178±0.0000	0.2145±0.0013	0.2025±0.0003	0.1893±0.0003
FPMVS-CAG	0.7324±0.0020	0.4062±0.0000	0.6635±0.0000	0.2370±0.0000	0.2070±0.0000	0.1928±0.0000
UDBGL	0.5943±0.0702	0.3967±0.0354	0.4038±0.0834	0.1527±0.0097	0.1424±0.0075	0.1397±0.0092
OMSC	0.7619±0.0000	0.4062±0.0000	0.6605±0.0000	0.2402±0.0000	0.2160±0.0000	0.1989±0.0000
OPMC	0.6395±0.0769	0.2619±0.0194	0.4561±0.0297	0.2019±0.0053	0.1616±0.0038	0.1445±0.0032
FastMICE	0.7819±0.0475	0.9704±0.0057	0.4334±0.0311	0.2014±0.0121	0.1704±0.0072	0.1491±0.0070
PCAL	0.7771±0.0313	0.6438±0.1128	0.5679±0.0410	0.2458±0.0063	0.1948±0.0080	0.1664±0.0040
			NMI			
LMVSC	0.6115±0.0162	0.4643±0.0689	0.5340±0.0139	0.1653±0.0071	0.1775±0.0029	0.1200±0.0021
SMVSC	0.5355±0.0207	0.1399±0.0000	0.6030±0.0000	0.1685±0.0009	0.1877±0.0004	0.1308±0.0003
FPMVS-CAG	0.6137±0.0018	0.1058±0.0000	0.6387±0.0000	0.1818±0.0000	0.1920±0.0000	0.1373±0.0000
UDBGL	0.5615±0.0557	0.1060±0.0739	0.5152±0.0562	0.1330±0.0089	0.0994±0.0101	0.0782±0.0086
OMSC	0.6534±0.0000	0.1058±0.0000	0.6420±0.0000	0.1855±0.0000	0.1893±0.0000	0.1274±0.0000
OPMC	0.5628±0.0431	0.0208±0.0138	0.6247±0.0150	0.1775±0.0054	0.2026±0.0044	0.1463±0.0016
FastMICE	0.6984±0.0220	0.9022±0.0176	0.6070±0.0191	0.1555±0.0152	0.2061±0.0068	0.1437±0.0076
PCAL	0.6713±0.0094	0.4755±0.0999	0.6092±0.0178	0.1965±0.0025	0.1723±0.0038	0.1262±0.0026

Table 8: Purity and F-Score on six datasets (mean ± standard deviation).

Method	MSRC	BBCSport	Caltech101-20	CCV	NUS-WIDE	NUSWIDEOBJ
			Purity			
LMVSC	0.7295 ± 0.0165	0.6688 ± 0.0496	0.7093 ± 0.0141	0.2390 ± 0.0068	0.2329 ± 0.0045	0.2335 ± 0.0047
SMVSC	0.6762 ± 0.0219	0.5074 ± 0.0000	0.7125 ± 0.0000	0.2429 ± 0.0019	0.2368 ± 0.0004	0.2333 ± 0.0003
FPMVS-CAG	0.7324 ± 0.0020	0.4816 ± 0.0000	0.7460 ± 0.0000	0.2639 ± 0.0001	0.2410 ± 0.0000	0.2391 ± 0.0000
UDBGL	0.6290 ± 0.0664	0.4057 ± 0.0419	0.6614 ± 0.0833	0.1843 ± 0.0069	0.1634 ± 0.0076	0.1670 ± 0.0087
OMSC	0.7619 ± 0.0000	0.4816 ± 0.0000	0.7452 ± 0.0000	0.2689 ± 0.0000	0.2445 ± 0.0000	0.2342 ± 0.0000
OPMC	0.6686 ± 0.0595	0.3601 ± 0.0106	0.7690 ± 0.0110	0.2352 ± 0.0045	0.2637 ± 0.0058	0.2590 ± 0.0026
FastMICE	0.8071 ± 0.0232	0.9704 ± 0.0057	0.7593 ± 0.0173	0.2400 ± 0.0076	0.2623 ± 0.0111	0.2529 ± 0.0109
PCAL	0.7871 ± 0.0110	0.6693 ± 0.0995	0.7337 ± 0.0138	0.2742 ± 0.0053	0.2325 ± 0.0038	0.2365 ± 0.0047
			F-Score			
LMVSC	0.5917 ± 0.0189	0.5147 ± 0.0748	0.3298 ± 0.0230	0.1161 ± 0.0038	0.0806 ± 0.0025	0.0814 ± 0.0019
SMVSC	0.5285 ± 0.0182	0.3070 ± 0.0000	0.6684 ± 0.0000	0.1312 ± 0.0006	0.1235 ± 0.0002	0.1305 ± 0.0003
FPMVS-CAG	0.6019 ± 0.0026	0.3043 ± 0.0000	0.6917 ± 0.0000	0.1368 ± 0.0000	0.1268 ± 0.0000	0.1343 ± 0.0000
UDBGL	0.4557 ± 0.0663	0.3932 ± 0.0111	0.3446 ± 0.0845	0.1054 ± 0.0034	0.1108 ± 0.0020	0.1111 ± 0.0022
OMSC	0.6413 ± 0.0000	0.3043 ± 0.0000	0.6987 ± 0.0000	0.1384 ± 0.0000	0.1326 ± 0.0000	0.1320 ± 0.0000
OPMC	0.5232 ± 0.0596	0.2239 ± 0.0079	0.4497 ± 0.0504	0.1216 ± 0.0032	0.0950 ± 0.0030	0.0960 ± 0.0018
FastMICE	0.6853 ± 0.0303	0.9403 ± 0.0115	0.3774 ± 0.0288	0.1187 ± 0.0062	0.0998 ± 0.0067	0.0959 ± 0.0045
PCAL	0.6636 ± 0.0149	0.5721 ± 0.0885	0.5907 ± 0.0594	0.1403 ± 0.0019	0.1179 ± 0.0041	0.1069 ± 0.0020

Table 9: Mean running time in seconds for eight methods on six datasets.

Method	MSRC	BBCSport	Caltech101-20	CCV	NUS-WIDE	NUSWIDEOBJ
LMVSC	0.326	0.777	9.840	7.690	3.421	146.446
SMVSC	0.832	2.985	9.682	8.621	3.994	50.468
FPMVS-CAG	1.567	5.889	18.920	13.879	7.920	98.619
UDBGL	0.975	1.674	24.246	52.509	29.467	663.391
OMSC	1.252	4.467	14.502	10.909	10.027	105.523
OPMC	0.048	0.472	3.471	0.797	0.892	32.218
FastMICE	0.476	7.942	9.321	8.269	5.534	16.383
PCAL	0.737	0.629	3.932	0.631	1.141	20.408

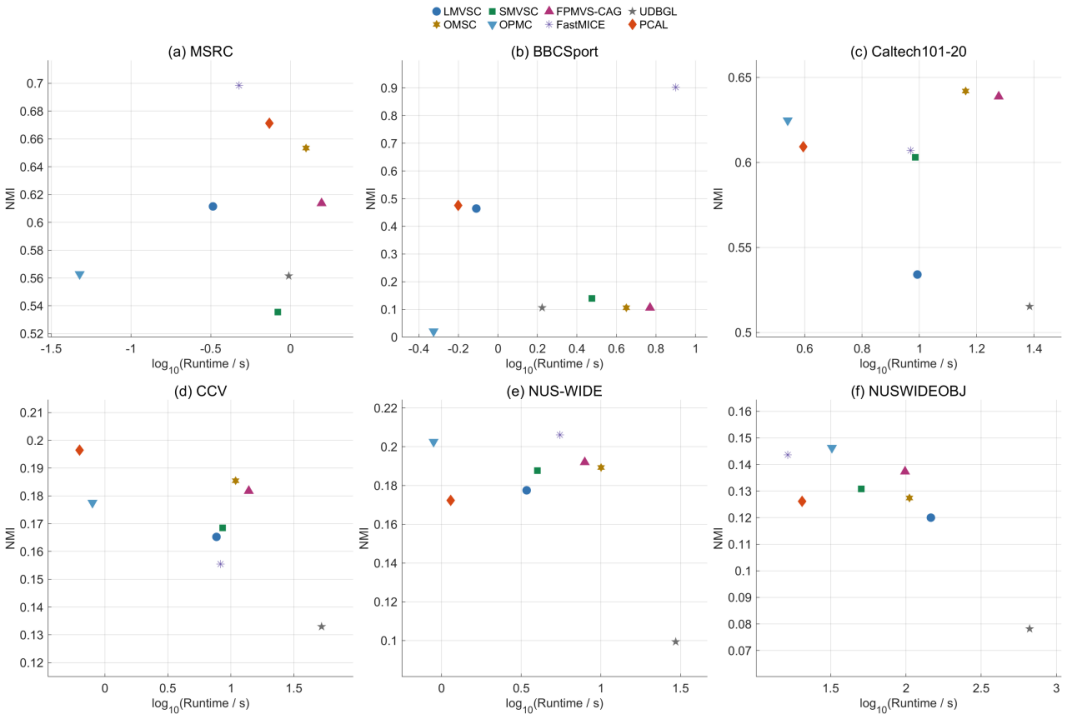

Figure 7: Mean NMI versus running time on six datasets. The horizontal axis uses a base-10 logarithmic scale.

centage points,respectively.PCAL is not best on Caltech101-20,NUS-WIDE,or NUSWIDEOBJ,and its BBCSport variation remains substantial.Taken together with running time,the results support a lower-cost,competitive-quality interpretation rather than universal superiority across datasets and metrics.